MERVA for ESA

Macro Reference

Version 4 Release 1

SH12-6377-01

MERVA for ESA

Macro Reference

Version 4 Release 1

SH12-6377-01

Note!

Before using this information and the product it supports, be sure to read the general information under "Appendix B. Notices" on page 327.

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 provides a reference to all macros available with the IBM licensed program Message Entry and Routing with Interfaces to Various Applications for ESA Version 4 Release 1 (abbreviated to MERVA ESA). It contains all of the macro formats, syntax rules, and operand descriptions.

When 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.

Code the MERVA ESA macros according to the rules described in the *High Level Assembler Language Reference*. The output produced is the control block defined by the input macros. MERVA ESA provides sample source for all required definitions. If errors are detected in the input, the control block produced is incomplete and should be discarded; error messages indicating the cause of the errors are printed in the output listing.

For the SWIFT Link it is assumed that you are familiar with the contents of the *S.W.I.F.T. User Handbook*; for the Telex Link you should be familiar with the telex terminology as defined by your local PTT. [1]

Note: 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®.

At the back of the book you find a list of abbreviations, a bibliography, and an index.

1. National Post and Telecommunication Authority (post, telegraph, telephone).

Summary of Changes

This edition of this manual reflects the following changes:

- The following DSLKPROC parameters have been added:
  - AUTHENT (for TYPE=SEND and TYPE=RECEIVE)
  - CCSIDP (for TYPE=SEND and TYPE=RECEIVE)
  - ISNCTLQ (for TYPE=SEND and TYPE=RECEIVE)
  - JIDRCVD (for TYPE=SEND and TYPE=RECEIVE)
  - JIDSENT (for TYPE=SEND and TYPE=RECEIVE)
  - OSNCTLQ (for TYPE=RECEIVE)
  - SECURE (for TYPE=SEND and TYPE=RECEIVE)
- The descriptions of the following DSLKPROC parameters have been modified:
  - ACKWQ
  - CCSID
  - COAWQ
  - CODWQ
  - EXIT
  - MRVCTLQ
  - PASCMID
  - TRACMFS
  - TRACTOF

Chapter 1. Introduction to MERVA ESA Macros

The first 3 characters of each macro identify the MERVA ESA component:

DSL	Base Functions
DWS	SWIFT Link
EKA	MERVA Link and FMT/ESA with MERVA Link
ENL	Telex Link

When using macros you should be aware of:
- The distinction between direct and central services
- The MERVA ESA buffer format
- The use of special characters in literals

Each of these points are described in the following.

Distinguishing between Direct and Central Services

When using MERVA ESA macros to call MERVA ESA services, you must make the distinction between calling the service as a direct service or as a central service (for details refer to *MERVA for ESA Concepts and Components*). Generally, the following rules apply:
- If a macro does not have an EP parameter, it is always called as a direct service.
- If a macro has an EP parameter, programs linked to DSLNUC (via DSLNPTT, DSLNTRT, or DSLNCMT) must use this parameter with the name of the called program. All other programs must not use the EP parameter but must use the appropriate DSLNIC macro later.

To find out how to call MERVA ESA service programs, refer to the *MERVA for ESA Customization Guide*.

MERVA ESA Buffer Format

Unless otherwise stated, all MERVA ESA buffers have a standard format which is described in the *MERVA for ESA Customization Guide* in the chapter on buffer standard.

Note: In MERVA ESA the buffer standard allows buffer sizes larger than 32KB (1KB equals 1024 bytes).

Using Special Characters in Literals

Literals are character strings of variable length, enclosed in single quotes and can consist of any valid EBCDIC characters. However, both characters, quote (’) and ampersand (&), must be specified twice in a literal. They will appear only once in the field generated in the routing table. For example, SMITH’S BANK must be coded 'SMITH''S BANK'.

Defining a Field Definition Table (FDT)

The following macros are used to define fields and to create the Field Definition Table (FDT):

DSLLFDT	Identifies the beginning of the FDT.
DSLLFLD	Defines a field entry.
DSLLSUBF	Defines a subfield entry.
DSLLGEN	Identifies the end of the FDT and completes the FDT generation.

Defining a Message Control Block (MCB)

Macros can be grouped together to define messages for one or more devices in one Message Control Block. The macros used to define messages are described in the following list:

DSLLDEV	Identifies the device type and operational options.
DSLLGRP	Defines the start of a logical group of fields.
DSLLUNIT	Defines the beginning of a unit of message fields.
DSLLMFLD, DSLLDFLD, DSLLNFLD	Define a message field, device field, or net field, respectively. When all three types of field definition macros are referred to, the notation DSLLxFLD is used in this book.
DSLLUEND	Defines the end of a unit.
DSLLGEN	Shows the end of an MCB and completes the MCB generation.

The hierarchy within an MCB is as follows:

- Device level, referring to all definitions starting with a DSLLDEV macro and ending with another DSLLDEV macro or a DSLLGEN macro.
- Unit level, referring to all definitions starting with a DSLLUNIT macro and ending with a DSLLUEND, DSLLDEV, or DSLLGEN macro. If no DSLLUNIT macro is defined for a device, the unit level is the same as the device level.
- Field level, referring to all DSLLxFLD definitions.

The following macros are used to provide conditions under which the fields in the message are to be processed, and to ease coding of the MCB by allowing embedding:

DSLLCOND	The conditions under which a part of the MCB is to be skipped or when a new page is to begin.
DSLLEXIT	Can be used to embed, dynamically, other message descriptions in an MCB.

In addition, assembler macros are used to provide screen and printer definitions for the field descriptions in MCBs. The use of such macros simplifies the MCB definitions and allows for a more flexible change of definitions if required. An example, DWSSW71A for SWIFT field 71, is shown in the *MERVA for ESA Customization Guide*.

Chapter 2. Notation Conventions

The following notations are used to define macros.

Symbols you must not enter in your specifications:
The following symbols indicate how a tag or command can be written:

[ ] Brackets indicate optional parameters:

[A | B | C]

This indicates that you can specify A, B, or C, or you can omit the parameter.

{ } Braces indicate that you must choose one from among several parameters:

{A | B | C}

This indicates that you must specify one of A, B, or C.

| A vertical bar indicates that you must choose one from among the parameters that are separated by this symbol.

Stacked parameters
Stacking parameters vertically indicates that you must choose one from among the parameters, exactly as if the parameters were separated by a vertical bar:

A

B

C

is equivalent to

A | B | C

Italics Italic letters and words contained in the format description represent variables. You must replace these variables with an actual value:

```
[label]
```

You need not enter the label name.

Bold characters
Bold characters contained in the format description must be entered as shown.

UPPERCASE and Symbols
Uppercase letters and the following symbols must be entered as shown here:

,	Comma
( )	Parenthesis
=	Equal sign

Note: Where you find the word *blank* in this book, leave a blank space: " ".

Lowercase
: Lowercase letters or words in italic represent variables for which specific information must be substituted when the parameter is entered.

Underlined
: Underlined characters or words indicate that MERVA ESA uses these parameters as default values, if nothing is specified.

Chapter 3. Base Functions: Macros

DSLCOM: Defining the MERVA ESA Communication Area

The DSLCOM macro maps the MERVA ESA service communication area DSLCMO. The service communication area contains address pointers to significant modules and storage areas needed to fulfill MERVA ESA service requests.

You must ensure that the fields of DSLCOM are filled appropriately before issuing a service request. DSLCOM field requirements are described in the *MERVA for ESA Customization Guide*.

Name	Operator	Operands

	`DSLCOM`	`[DSECT={NO|YES}]`
		`[,NUC={NO|YES}]`

Programming Notes:

DSECT
Whether DSLCOM is to be mapped as a DSECT:

NO Use a DSLCOM service communication area without a DSECT. This is the default.

YES Map DSLCOM as a DSECT.

NUC
Whether to map for a program link-edited to DSLNUC:

NO Use a DSLCOM service communication area without the parts needed for a program link-edited to DSLNUC. This is the default.

YES Map for a program link-edited to DSLNUC.

Note: The MERVA ESA service communication area contains the definition of the parameter lists for DSLTRAP at label COMTRAPL, and for DSLSRVP at label COMSRVPL.

DSLCWA: Defining Storage in the CICS CSA

The DSLCWA macro maps the DSLCWA DSECT. It is used only with MERVA ESA running in CICS®. MERVA ESA requires 112 (X'70') bytes in the common work area (DFHCWA) of CICS. The address of DFHCWA must be obtained from CICS. DSLCWA starts from there at an offset that you can specify with the CWAOFF parameter of the DSLPARM macro.

DSLCWA is required for MERVA ESA intraregion communication and, under VSE, for MERVA ESA operator command processing from the VSE system console.

The label CWALEN contains the length of the CWA area.

Name	Operator	Operands
	`DSLCWA`	

Programming Notes: This macro has no parameters.

Note: User-written applications must not change the MERVA ESA area of the CWA.

DSLDPTFx: Defining the Installation Macros

The DSLPTFx macros are system-generation macro and are not part of a programming interface.

MERVA ESA provides the following installation macros:

DSLDPTFP
: Definitions of the global JCL cards, including POWER extensions if required.

  Prerequisite to the macros DSLDPTFA, DSLDPTFL, or both.

DSLDPTFA
: Punches the JCL to assemble source members, or to pre-edit macros.

  Requires the macro DSLDPTFP as prerequisite.

DSLDPTFL
: Punches the JCL for link-editing of modules.

  Requires the macro DSLDPTFP as prerequisite.

For more information refer to the *MERVA for ESA Installation Guide*.

In addition the following macros are used by the installation macro DSLGEN:
- DSLDPTFJ
- DSLDLINK
- DSLDJCDA

DSLDSFDT: Mapping the FDT Data Structures

The DSLDSFDT macro is used to map the following data structures of the MERVA ESA Field Definition Table:

1. The Field Definition Table header
2. The Field Definition Table entry of a main field
3. The Field Definition Table entry of a subfield

Name	**Operator**	**Operands**
[*label*]	**DSLDSFDT**	

Programming Notes:

label
A unique statement label according to assembler language conventions.

The DSLDSFDT macro has no parameters.

DSLDSMCB: Mapping the MCB Data Structures

The DSLDSMCB macro is used to map the following data structures of a Message Control Block (MCB):

1. The data reference
2. The literal reference
3. The buffer layout of a DSLLCOND O1 buffer specification
4. The Message Control Block header layout
5. The layout of an MCB device description
6. The layout of an MCB unit description
7. The layout of an MCB field entry
8. The layout of an MCB display field (for screen, HARDCOPY, and system printers)
9. The layout of an MCB net field (for communication lines, NOPROMPT, and sequential files)
10. The layout of an MCB condition
11. The layout of an MCB exit

Name	Operator	Operands
[*label*]	**DSLDSMCB**	

Programming Notes:

label
A unique statement label according to assembler language conventions.

The DSLDSMCB macro has no parameters.

DSLECOFN: Defining the End-User Driver Interface Area

The macro DSLECOFN is used to describe the interface area between the End-User Driver and programs written by the user for functions or End-User Driver exits.

Name	Operator	Operands
	DSLECOFN	[**PREFIX=**{<u>**FN**</u>|*cc*}]

Programming Notes:

PREFIX
One to 2 characters of the field names. The default is FN.

DSLEISPA: Defining the Interface between DSLEUD and Its SPA Subroutine

The macro DSLEISPA is used to describe the interface between the End-User Driver program DSLEUD and its SPA subroutine DSLEOSPA. This macro is available for IMS only.

Name	Operator	Operands
[*label*]	**DSLEISPA**	[**PREFIX=**{<u>**TX**</u>|*cc*}]

Programming Notes:

label
: A unique statement label according to assembler language conventions.

PREFIX
: One or 2 characters that are to be the first characters of the labels generated in the storage description. The default is TX.

This macro defines the following:
- The pointer and fields necessary for connection to IMS
- The first 30 bytes of the IMS system SPA
- The debugging fields
- The addresses and the length of the buffers to be saved on internal SPA file
- The work fields of DSLEOSPA
- The register contents at time of entry and exit of DSLEOSPA
- The fields for informing DSLEUD of any erroneous case

Note: Any error arising after return to DSLEUD overlays the error message from DSLEOSPA.

DSLEPT: Defining the End-User Driver Program Table

The macro can be used to:

- Get the entry description (DSECT)
- Define a function program for the End-User Driver

Getting the DSECT

Name	Operator	Operands

	`DSLEPT`	`TYPE=DSECT`
		`[,PREFIX={`<u>`DSLE`</u>`|`*`cccc`*`}]`

Programming Notes:

TYPE
Gets the entry description (DSECT).

PREFIX
The 1 to 4 characters of the field names in the dummy section of the DSLEPT entry. The default is DSLE.

Defining a Function Program for the End-User Driver

In this table, all function programs must be specified.

Name	Operator	Operands

[*label*]	`DSLEPT`	`[TYPE={`<u>`ENTRY`</u>`|INITIAL|FINAL}]`
		`,NAME={`*`cccccccc`*`}`
		`[,CMD={`*`cccccccc`*`}]`
		`[,ERR={`<u>`NO`</u>`|YES}]`
		`[,EXLIST={`<u>`NO`</u>`|YES}]`

Programming Notes:

label
An optional assembler label for the entry. This label is ignored if TYPE=INITIAL or TYPE=FINAL is specified.

TYPE
The type of table entry to be generated:

INITIAL	The table definition begins and generates the table header description. The other parameters are ignored.
ENTRY	A table entry is to be generated. This parameter is optional. This is the default.
FINAL	The table definition ends and generates the table trailer description. The other parameters are ignored.

NAME
The name of the function program. The name can be 1 to 8 characters long.

CMD
The name of the command table that defines the function commands for the users. The name can be 1 to 8 characters long.

ERR
Whether the function program will get control even if the command is in error. In some special cases it may be necessary to see the invalid command. NO is the default.

EXLIST
If the function program has an external command table, and if DSLEUD does not find the command in the known command tables, this parameter specifies whether DSLEUD gives control to the function program. NO is the default.

DSLFLT: Defining the File Table

The DSLFLT macro generates the MERVA ESA File Table.

Generating File Table Entries

The file table is required for the following MERVA ESA programs:

- File service program DSLFLVP
- File utility program DSLFLUT
- Online general file maintenance program DSLEFLM
- SWIFT Correspondents File utility DWSCORUT
- SWIFT Currency Code File utility DWSCURUT
- SWIFT address expansion routine DWSMX001
- Telex correspondents expansion ENLMU301 and ENLME094
- Currency Code checking DWSMU141

Each file you want to process using these services must be defined in the file table.

The name of the file table is defined in the MERVA ESA customizing parameters. The default is DSLFLTT.

Standard MERVA ESA Files

MERVA ESA provides the following standard files:

- Nicknames File
- SWIFT Correspondents File
- SWIFT Currency Code File
- Telex Correspondents File

These files are defined in the sample file table that is provided with MERVA ESA. Most of these definitions must not be modified. The descriptions of the DSLFLT macro show which values in the sample file table must not be changed.

The following list is an overview of those parts of your installation that are related with the file table and must be consistent. The descriptions of the DSLFLT macro give the details.

- Jobs that start programs
- Jobs that define VSAM KSDS clusters
- CICS file control table (CICS only)
- DBDs of DL/I HISAM DBs (IMS only)
- PSBs (IMS only)
- Function table
- Message type table
- Field definition table (FDT)
- MCBs that map panels and listings

Global Rules for the File Table Structure

The following rules are related with the global structure of the file table:

- There must be only one DSLFLT TYPE=INITIAL. It must be the first instruction.
- There must be only one DSLFLT TYPE=FINAL. It must be the last instruction.
- There must be at least one DSLFLT TYPE=DAT instruction. The maximum number of these instructions is 127.

- Each TYPE=DAT must be followed by a TYPE=FLD. One TYPE=DAT and the following TYPE=FLD define one file. In both macros, the values specified by the DAT operand and the values specified by the FLD operand must be the same.

Note: There may be several entries TYPE=FLD where the values specified by the FLD operand are the same, that is, search fields of different files may have the same name. This is even recommended when search fields of different files have the same meaning.

The order of TYPE=DAT instructions determines the order of files in the file selection menu during the online file maintenance.

The File Table Entry with TYPE=INITIAL

The first DSLFLT macro in a file table must be TYPE=INITIAL. There must be only one DSLFLT macro with TYPE=INITIAL.

The generated entry starts with a CSECT statement. If a label is specified, the CSECT name is the label. Otherwise, the CSECT name is DSLFLTT.

Name	Operator	Operands
[*label*]	**DSLFLT**	**TYPE=INITIAL**

The File Table Entry with TYPE=FINAL

The last DSLFLT macro in a file table must be TYPE=FINAL. There must be only one DSLFLT macro with TYPE=FINAL.

Name	Operator	Operands
	DSLFLT	**TYPE=FINAL**

File Table Entries with TYPE=DAT

For each file, you must define one DSLFLT macro with TYPE=DAT. This instruction is for definitions relating to the total data of the file.

Name	Operator	Operands
	DSLFLT	**TYPE=DAT** **,DAT=***cccccccc* **,FIELDS=**(*cccccccc*,...,*cccccccc*) **,FLD=***cccccccc* **,LENGTH=***nnnnn* **,MSGID=***cccccccc* **,NAME=***ccccccc* [**,COLS80=**(*cccccccc*,...,*cccccccc*)] [**,COLS132=**(*cccccccc*,...,*cccccccc*)] [**,DESCR=**([*quoted literal*[,*quoted literal*]])] [**,JOURNAL=**{<u>**NO**</u>|**YES**}] [**,MAINT=**{<u>**NO**</u>|**YES**|**PRIVATE**|**COMMON**}] [**,ROWS24=***nn*] [**,ROWS27=***nn*] [**,ROWS32=***nn*]

Name	Operator	Operands
		[,**ROWS43=***nn*] [,**RSEG=***cccccccc*] [,**SELECT=**([*cccccccc*[,*cccccccc*]])] [,**SHARED=**{<u>**NO**</u>|**YES**}]

Programming Notes:

TYPE=DAT
The file table entry for a file.

DAT
The unique name of the file. The name can be up to 8 characters long. This operand is mandatory.

In IMS installations, the specified name is also the root segment name of the related DL/I DB unless the RSEG parameter specifies a different name.

Programs use this name to refer to the file when calling the file service program DSLFLVP.

For a standard MERVA ESA file, you must not change the name specified in the sample file table.

FIELDS
The names of the record fields. This operand is mandatory. You can define a list of up to 28 names. Each name can consist of up to 8 characters. The sequence has no effect. Duplicates are not allowed.

The set of names must be the set of field names specified in the net section of the MCB of the file (see operand MSGID).

The DSLEFLM program uses these names to refer to the record fields when calling the TOF supervisor service program DSLTOFSV. This requires that every name is a field name defined in the Field Definition Table (see the macro DSLLFLD).

You can omit the name of the search field and the names DSLFLCUP (Creation Stamp), and DSLFLUP (Last Update Stamp), as they are always generated.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

FLD
The name of the search field. The name can be up to 8 characters long. This operand is mandatory.

In IMS installations, the specified name must be the sequence field name of the related DL/I DB. Programs use the specified name to refer to the search field when calling the TOF supervisor service program DSLTOFSV. This requires that the name is defined as a field name in the MERVA ESA Field Definition Table (see the macro DSLLFLD).

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

LENGTH
The record length. This operand is mandatory. The value must be a decimal number in the range from 5 to 32761. Note that only files of fixed record length may be defined.

In CICS installations, the specified number must be the record length of the related VSAM cluster.

In IMS installations, the specified number must be the root segment length of the related DL/I DB. The record length of the VSAM cluster is different, an appropriate value is recommended in the listing returned from the DBD generation step.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

MSGID
A unique message ID of the MCB of the file. This operand is mandatory. The name can be up to 8 characters long. It must be the message ID defined in the message type table for the MCB of the file. Each file defined in the file table must use a different MCB.

The MCB maps panels or listings of records of the file for screen or printer devices:

- The net section describes the record layout starting from offset 4. This description is used to map records between the file buffers and the TOF area.
- The system printer section defines the layout of the listing produced by the file batch utility program DSLFLUT.
- The screen section defines layouts for panels and printouts for the online file maintenance. The panels are displayed when the file is selected from the file-selection menu.

NAME
The unique logical file name from the view of the basic DB system (VSAM and CICS file control in CICS installations, DL/I in IMS installations). This operand is mandatory. The name can be up to 7 characters long.

In IMS installations, the specified name must be the DBD name of the related DL/I DB. It is defined in the DBD. Note that the DBD name is also specified in the DB PCBs for the file, and DB PCBs for the file must be included in the PSBs of all programs that access the file.

In CICS installations, the specified name must be the DD name (MVS™) or DLBL name (VSE) that is used for allocating the related VSAM cluster. Note that the DD/DLBL name is also specified in the CICS file control table.

COLS80
The names of the record fields that are shown on the list panels of the online file maintenance if the display device is an 80-column screen. This operand is optional. If omitted, only the search field is listed.

If the operand is specified, the set of names must be a subset of the names specified in the FIELDS operand.

You can omit the search field as it is always generated. Related operands for TYPE=DAT are FIELDS, MSGID, and MAINT.

COLS132
The names of the record fields that are shown on the list panels of the online file maintenance if the display device is a 132-column screen in the same way as COLS80 for an 80-column screen.

Note: The file MCBs provided with MERVA ESA do not map different list panels depending on the number of screen columns. You must modify these MCBs if you want to have different list panels for 80 columns and 132 columns.

DESCR

A unique description of the file. The description is shown in the file selection menu of the online file maintenance. This operand is mandatory unless MAINT=NO is specified. For MAINT=NO it is optional.

For a file that is not shared, or for a shared file with MAINT=PRIVATE or MAINT=COMMON, you must specify one descriptor of up to 64 characters enclosed in quotes.

For a shared file with MAINT=YES, you must specify two descriptors of up to 64 characters enclosed in quotes. The first one is for private data, the second one for common data. For more information on shared and non-shared files, see the description of the SHARED operand. If the length of a descriptor exceeds 64 characters, it is truncated.

JOURNAL

Whether updates to a file are recorded in the MERVA ESA journal.

NO

The updates are not recorded in the journal. JOURNAL=NO is the default.

YES

The updates are recorded in the journal.

The layout of these journal records is described in Appendix A of the manual *MERVA for ESA Concepts and Components*.

MAINT

Whether the file is available in the online file maintenance. For a shared file, the operand shows if private data, common data, or both are available.

Related operands for TYPE=DAT are DESCR, SELECT, and SHARED. The only related operand for TYPE=FLD is DESCR.

NO

The file is not available in the online file maintenance. The file selection menu does not include any entry for the file. This is the default.

If MAINT=NO is specified for all files in the file table, terminal users cannot select the online file maintenance.

YES

The file is available in the online file maintenance:

- For a non-shared file, the file selection menu includes one entry showing a selection ID and a description of the file.
- For a shared file, both private and common data are available, and the file selection menu includes two entries showing selection IDs and descriptions of both private and common data, respectively.

PRIVATE

For a shared file, only the private data owned by the user who is signed on are available in the online file maintenance. The file selection menu includes one entry showing a selection ID and a description of the private data.

MAINT=PRIVATE is not allowed unless SHARED=YES is specified.

COMMON
For a shared file, only the common data are available in the online file maintenance. The file selection menu includes one entry showing a selection ID and a description of the common data. MAINT=COMMON is not allowed unless SHARED=YES is specified.

ROWS24
The number of record entries shown on the online file maintenance, if the display device is a 24-line screen.

This operand is optional. The value must be in the range from 1 to 22. The default is 1.

The ROWS24 value must not exceed the maximum number of list entries that has been specified in the MCB of the file (see the MSGID operand).

Related operands for TYPE=DAT are MSGID and MAINT.

ROWS27
This operand is similar to ROWS24 except that it is for display devices with 27 lines, and the range of values that can be specified is from 1 to 25.

ROWS32
This operand is similar to ROWS24 except that it is for display devices with 32 lines, and the range of values that can be specified is from 1 to 30.

ROWS43
This operand is similar to ROWS24 except that it is for display devices with 43 lines, and the range of values that can be specified is from 1 to 41.

RSEG
For IMS installations, the name of the root segment name of the related DL/I DB. If the RSEG parameter is not specified, the name specified in the DAT parameter is used as default.

SELECT
The unique selection ID of the file.

The selection ID is shown in the file selection menu of the online file maintenance.

This operand is mandatory unless MAINT=NO is specified. For MAINT=NO it is optional.

For a non-shared file, or a shared file with MAINT=PRIVATE or MAINT=COMMON, you must specify one selection ID of up to 8 characters.

For a shared file with MAINT=YES, you must specify two different selection IDs of up to 8 characters each. The first is for private data, the second for common data.

SHARED
Whether a file is shared.

NO
The file is not shared. An example is the SWIFT Correspondents File. SHARED=NO is the default.

YES
The file is shared. An example is the Nicknames File. Processing of a shared file appears as if there is data common to all users and other data owned by the user who is signed on. Two examples related with the Nicknames File are:

- In the online file maintenance, records of common data (common nicknames) are visible to any user regardless of who created the records. Records of private data (private correspondents names) are visible only to the user who created the records.
- In a message-processing function where expansion of common names has been defined, common names are expanded regardless of who is signed on. In a message-processing function where expansion of private names has been defined, private names are expanded only if they have been created by the user who is signed on.

For a shared file, the search field must be extended by 8 bytes. This part is the *owner prefix*. The normal record key then starts at offset 8. The owner prefix contains either the literal '*   ' (showing common ownership) or a user ID (showing private ownership). The owner prefix is not displayed during the online file maintenance.

Related operands for TYPE=DAT are DESCR, MAINT, and SELECT. Related operands for TYPE=FLD are INFLEN and LENGTH.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

File Table Entries with TYPE=FLD

For each file you must define one DSLFLT macro with TYPE=FLD. This instruction is for definitions related with the search field.

Name	Operator	Operands
	DSLFLT	**TYPE=FLD** **,DAT=***cccccccc* **,FLD=***cccccccc* **,LENGTH=***nnn* **,OFFSET=***nnnnn* [**,CHECK=**{**BASIC** }] {**ALPHA** } {**ALPHANUM**} {**DBCS** } {**NUM** } {**SWIFT** } [**,DESCR=***quoted literal*] [**,INFLEN=***nnn*]

Programming Notes:

TYPE=FLD
The file table entry for the search field of a file.

DAT
The unique name of the file to which the search field belongs. This operand is mandatory. The value must be the same as the value specified by the DAT operand in the related file table entry TYPE=DAT.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

FLD
The name of the search field. This operand is mandatory. The value must be the same as the value specified by the FLD operand in the related file table entry TYPE=DAT.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

LENGTH
The length of the search field. This operand is mandatory. The value must be a decimal number in the range from 1 to 255. The LENGTH value must be equal to or greater than the INFLEN value. For a shared file, the number must be greater than or equal to 9. If CHECK=SWIFT is specified, the number must be greater than or equal to 11. If CHECK=SWIFT is specified and the file is shared, the number must be greater than or equal to 19.

The specified number must always be the key length of the related VSAM cluster. It must also be the maximum length defined for the search field in the Field Definition Table.

In IMS installations, the specified number must be the sequence field length of the related DL/I DB. The LENGTH parameter can be chosen to be greater than the actual length of the search field (for example, to allow for increasing the length of the search field information in future changes without having a new VSAM, CICS, or IMS definition of the file).

Related operands for TYPE=FLD are CHECK and INFLEN. The only related operand for TYPE=DAT is SHARED.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

OFFSET
The offset of the search field. This operand is mandatory. It must be a decimal number in the range from 4 to record length minus the value of the LENGTH parameter. Note that the search field must not overlap with the first 4 bytes of the record.

In CICS installations, the specified number must be the key offset of the related VSAM cluster.

In IMS installations, the specified number must be the sequence field offset from the beginning of the root segment. The key offset of the related VSAM cluster is different; the required value is given by the listing returned from the DBD generation step.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

CHECK
The type of data the search field must contain. For all data types, the search field must never start with a blank. The file service program DSLFLVP checks the type of data when the request type is ADD. The ADD request is rejected if the search field of the record provided in the record buffer does not contain the correct type of data. The check is not made on the owner prefix of shared files. Note that both lowercase and uppercase input is allowed on MERVA ESA panels, and MERVA ESA programs translate lowercase input into uppercase.

Related operands for TYPE=FLD are INFLEN and LENGTH. The only related operand for TYPE=DAT is SHARED.

The CHECK parameter can have the following values:

BASIC
Any character is allowed except blanks, commas (,), and apostrophes ('). Trailing blanks are allowed.

CHECK=BASIC is the default. It is used for the Telex Correspondents File.

ALPHA
Only uppercase letters are allowed for the search fields. Trailing blanks are allowed.

ALPHANUM
Only uppercase alphanumeric characters are allowed for the search fields. Trailing blanks are allowed.

DBCS
Double-byte character set (DBCS) data are allowed for the search fields.

NUM
Only digits are allowed for the search fields. Trailing blanks are allowed.

SWIFT
The search fields must be in SWIFT address format, that is, characters 1-8 are *AAAAAAXX* and characters 9-11 are blank, or characters 1-11 are *AAAAAAXXXXX*, where *A* means uppercase letters and *X* means uppercase alphanumeric characters.

CHECK=SWIFT is recommended for the SWIFT Correspondents File.

DESCR
A description of the search field. The description is used in the online file maintenance. It is used in MERVA ESA error messages that refer to the search field.

This operand is mandatory unless MAINT=NO is specified. For MAINT=NO it is optional. The descriptor can be up to 24 characters long. If this length is exceeded, it is truncated.

The specified descriptor is also used for the search field on the record panels and list panels in the online file maintenance. It should also be used as a literal for the search field in the screen section of the MCB of the file (see operand MSGID).

INFLEN
The information length, that is, the number of leading search field characters that are actually used as search information by MERVA ESA.

This operand is optional. If you omit this operand, the default is the value specified by the LENGTH operand unless CHECK=SWIFT is specified. If CHECK=SWIFT is specified, the default is 11 for non-shared files or 19 for shared files.

If a value is specified, the value must be a decimal number in the range from 1 to 255. It must not exceed the value specified by the LENGTH operand. For a shared file, the number must be greater than or equal to 9 as the search information must include the owner prefix.

If the INFLEN value is smaller than the LENGTH value, the bytes in positions greater than the INFLEN value must be blanks in the search fields of all records.

Record and list panels should show the search argument without the owner prefix and without the blanks in positions greater than the INFLEN value.

Related operands for TYPE=FLD are CHECK and LENGTH. Related operands for TYPE=DAT are MSGID and SHARED.

For a standard MERVA ESA file, you must not change the value specified in the sample file table.

DSLFLV: Calling the MERVA ESA General File Service

The DSLFLV macro has two functions, to:

- Call the general file service program DSLFLVP
- Map the parameter list and the request control block of DSLFLVP

In this macro the addresses of storage areas may be specified in one of the following ways:

addr	An assembler instruction label of the storage area
(r)	For a register that contains the address.

Calling the General File Service

You can use the DSLFLV macro to read and write general MERVA ESA files. All files referenced with the DSLFLV macro must be included in the file table.

The file table is generated using the DSLFLT macro.

Name	Operator	Operands

[*label*]	**DSLFLV**	**TYPE=**{**ADD** }
		{**CLOSE** }
		{**DELETE** }
		{**GET** }
		{**GETNEXT** }
		{**OPEN** }
		{**REPLACE** }
		
		,DAT={'*cccccccc*'|*addr*|(*r*)}
		
		,TS={*addr*|(*r*)}
		
		[**,ARG=**{*addr*|(*r*)}]
		
		[**,BUF=**{*addr*|(*r*)}]
		
		[**,MF=**<u>**E**</u>]
		
		[**,OPT=**(*opt*,...,*opt*)]
		
		[**,PREFIX=**{<u>**FLV**</u>|*ccc*}]
		
		[**,TBUF=**{*addr*|(*r*)}]

Programming Notes: DSLCOM must be addressable and the field COMFLVPA must contain the entry address of the general file service program DSLFLVP. The field COMFLTTA must contain the address of the MERVA ESA file table.

The DSLFLVP parameter list and the request control block must be addressable.

label
A unique label according to assembler language conventions.

TYPE
The file service function to be performed.

ADD
Adds a record to the file. The search field of the new record must not be a duplicate of the search field of a record that already exists in the file.

The search field must contain the type of data that has been specified in the file table with the CHECK parameter of the DSLFLT macro.

Related operands are BUF and DAT. No options are allowed.

CLOSE
In a CICS-related batch environment only, VSAM closes the VSAM cluster defined for the file. For other environments, the request is ignored. The only related operand is DAT. No options are allowed.

DELETE
Deletes a record from the file. The file record is deleted if the search argument is found. Related operands are ARG, DAT, and TBUF. No options are allowed.

GET
Gets a record from the file. This request carries out a direct file access. Specify the search argument in the operand ARG.

Related operands are ARG, BUF, DAT, and OPT.

GETNEXT
Gets the next record in the file. This request carries out a sequential file access, depending on the previous request. A current position must have already been established in the file (using TYPE=GET and OPT=GETNEXT). A reason code is generated when the end of file has been reached. Related operands are BUF, DAT, and OPT.

OPEN
In a CICS-related batch environment only, VSAM opens the VSAM cluster that has been defined for the file. The cluster is always opened for direct and sequential access. It is opened for input and, unless OPT=GET is specified, for output. In other environments, this request is ignored.

Related operands are DAT and OPT.

REPLACE
Replaces a record in the file having the same search field. If the record to be replaced does not exist, the request is not executed. Related operands are BUF, DAT, and TBUF. No options are allowed.

DAT
The search argument and the name of the file. The file name must be defined in the file table. DAT is a literal up to 8 characters long enclosed in single quotes, or the address of an 8-byte field containing the file name, or a general register containing the address of that field. The name must be left-adjusted and padded with blanks.

This operand is mandatory with all requests.

TS
A temporary storage area used by DSLFLVP.

It may be the label of the storage, or a general register containing the storage address. This operand is mandatory.

The storage area must be aligned on a fullword boundary. The length must be at least 3072 bytes.

ARG
The search argument. ARG must specify the label of a field that contains the search argument, or a general register containing the address of that field. The

length of this field must be equal to the length of the search field defined in the file table (LENGTH parameter of the DSLFLT macro).

This operand is mandatory with the TYPE=DELETE and TYPE=GET. For other request types it is rejected.

BUF

The record buffer. It may be the label of the record buffer, or a general register containing the address of the buffer. The buffer must have the standard MERVA ESA format which is described in the *MERVA for ESA Customization Guide*, in the chapter dealing with the buffer standard of MERVA ESA.

For TYPE=ADD and TYPE=REPLACE, this buffer contains the new record. The record length must be equal to the length defined in the file table (LENGTH parameter of the DSLFLT TYPE=DAT macro).

For TYPE=GET and TYPE=GETNEXT, this buffer will receive the retrieved record. The buffer length must be at least 4 greater than the record length defined in the file table.

This operand is mandatory with the request types ADD, GET, GETNEXT, and REPLACE. For other request types, it is rejected.

MF

The format of the macro. It must be E for execute form. This is the default.

OPT

An option for the file service request. One or two options can be specified.

This operand is optional for TYPE=GET, GETNEXT, and OPEN. It is not allowed for all other request types.

EQ

For TYPE=GET only, shows the search condition that gets the record from the file where the search field content is equal to the search argument.

This option is mutually exclusive with the options GT and GTEQ. OPT=EQ is the default when neither EQ, GT, nor GTEQ is specified.

GET

For TYPE=OPEN in a CICS-related batch environment only, the VSAM cluster defined for the file is opened for input but not for output.

This implies that the next request types must be GET or GETNEXT.

GETNEXT

For TYPE=GET only, specifies that TYPE=GETNEXT requests will follow.

This option ensures that, after the GET is complete, a current position is available for the TYPE=GETNEXT requests.

FINISH

For TYPE=GETNEXT only, do not get another record from the file but ensure that a new current position may be established subsequently.

For a CICS transaction, the current position is released and no more requests with the request type GETNEXT are allowed until a new current position has been established (see option GETNEXT).

GT

For TYPE=GET only, shows the search condition that gets the first record from the file where the search field content is greater than the search argument.

This option is mutually exclusive with the options EQ and GTEQ.

GTEQ
For TYPE=GET only, shows the search condition that gets the first record from the file where the search field content is equal to or greater than the search argument.

This option is mutually exclusive with the options EQ and GT.

Note: When starting to read a file sequentially from the beginning, the search argument for the options GT and GETNEXT can be set to binary zeros. Then the first record of the file whose search argument does not contain binary zeros is taken.

PREFIX
One to 3 characters of the first characters of the labels in the DSLFVLP list. The list storage area must be mapped using the same prefix (see DSLFLV MF=L). The default is FLV.

TBUF
A temporary record buffer used by DSLFLVP. It may be the label of the buffer, or a general register containing the address of the buffer.

TBUF may be specified for request types DELETE or REPLACE only. This operand is optional, but it is recommended for improved system performance.

The buffer length must be at least 4 greater than the record length defined in the file table. The buffer must be aligned on a fullword boundary. The first halfword must contain the buffer length.

Compatibility of Request Types and Macro Operands

The following table shows which operands are required or optional for a given request type.

TYPE	ARG	BUF	DAT	OPT	MF	PREFIX	TBUF	TS
ADD	-	X	X	-	O	O	-	X
CLOSE	-	-	X	-	O	O	-	X
DELETE	X	-	X	-	O	O	O	X
GET	X	X	X	O	O	O	-	X
GETNEXT	-	X	X	O	O	O	-	X
OPEN	-	-	X	O	O	O	-	X
REPLACE	-	X	X	-	O	O	O	X

Legend:

X	Required
O	Optional
–	Not allowed

Mapping the Parameter List and the Request Control Block of DSLFLVP

The *list* form of the DSLFLV macro may be used to generate a storage definition of the DSLFLVP lists. The storage area includes a parameter list and a request control block. The parameter list contains the required storage addresses. The request control block contains the request type, the options, and the file name.

The request control block also contains feedback fields which are filled on completion of the request. These include a return code, a reason code, and a diagnostic message that can be printed or displayed.

Name	Operator	Operands
[*label*]	**DSLFLV**	**MF=L** [**,PREFIX=**{<u>**FLV**</u>|*ccc*}]

Programming Notes:

label
: A unique label according to assembler language conventions. Regardless of the label specification, the label *xxx*L is always generated at the beginning of the storage definition where *xxx* is the value specified by the PREFIX operand.

MF
: The format of the macro. It must be L for list form.

PREFIX
: One to 3 characters that are to be the first characters of the labels generated in the storage description. The default is FLV.

DSLFNT: Defining the Function Table Entry

The DSLFNT macro is used to:

- Generate a function table
- Map the function table entry

The function table definition consists of a sequence of DSLFNT macros. The first instruction should have a label; if it does not, the default label DSLFNTT is used. The last statement of the table must be the END statement.

If errors are detected in a specific DSLFNT macro, appropriate MNOTEs are provided. For severe errors, the macro does not generate a table entry.

MERVA ESA provides a sample function table DSLFNTT. The name of the function table to be used in the system has to be specified in the MERVA ESA customizing parameter definition DSLPRM.

Generating a Function Table

Name	Operator	Operands
[*label*]	**DSLFNT**	**TYPE=**{**INITIAL**|**FINAL**}

Programming Notes:

label
: The name of the CSECT generated. It must be a unique label according to the assembler language naming conventions. The label must be specified in the first function table definition.

TYPE
: TYPE=INITIAL must be the first function table definition. If the first statement does not contain a label, DSLFNTT is used. In the second and following calls of DSLFNT, the label is disregarded. TYPE=FINAL must be the last function table definition statement.

Generating a Function Table Entry

Name	Operator	Operands
[*label*]	**DSLFNT**	[**TYPE=ENTRY**]
		[**,CHECK=**{**YES**|**NO**}]
		[**,CHECKAUT=**{**NO**|**YES**}]
		[**,COPY=***cccccccc*]
		[**,DE=**{**NO**|**YES**}]
		[**,DESCR=***quoted literal*]
		[**,DQFILL=**{**YES**|**NO**}]
		[**,EXPAND=**{**NO**|**CLEAR**|**UNCOND**}]
		{**(COND,COND)** }
		{**(COND,UNCOND)** }
		[**,EXPNAM=**{**NO**|**COMMON**|**PRIVATE**}]
		{**(PRIVATE,COMMON)** }
		{**(COMMON,PRIVATE)** }
		[**,FRAME=**{*tttttttt* }]
		{([*tttttttt*],*bbbbbbbb*)}
		{**(0TOP,0BOT)** }

Name	Operator	Operands
		[,FOUREYE={OFF }]
		{(PREVIOUS,STORE)}
		{(PREVIOUS,ANY) }
		{(ALL,STORE) }
		{(ALL,ANY) }
		[,INTQUE=*cccccccc*]
		[,KEEPMSG={NO|YES}]
		[,KEY1=(*cccccccc*,*nn*,*p*[,NOMOD])]
		[,KEY2=(*cccccccc*,*nn*,*p*[,NOMOD])]
		[,LTERM=*cccccccc*]
		[,MODE={SW2|SW1}]
		[,MQI={NO|YES}]
		[,MSGID=*cccccccc*]
		[,MSGLIM=(*nnnnn*[,CHKP|NOCHKP])]
		[,MSGSEL=LIST]
		[,NAME=*cccccccc*]
		[,NEXT=*cccccccc*]
		[,NOPR={DISPLAY|YES|NO}]
		[,NOTIFY={NO|YES}]
		[,PARTID=*cc*]
		[,PFGROUP=*nn*]
		[,PFKSET=*cccccccc*]
		[,PRFORM=({A|B|...|Z|0|1|...|9},{0|1|2|3|4})]
		[,PRINT=*cccccccc*]
		[,PROGRAM={DSLEMSG|*cccccccc*}]
		[,PROT={NO|YES}]
		[,QUEUE={NO|YES|DUMMY}]
		[,RELATED=(*cccccccc*,*cccccccc*)]
		[,RETYPE={NO|YES}]
		[,ROUTE=*cccccccc*]
		[,SPCMND={DEL|OK|ROU|AUT }]
		{([DEL][,OK][,ROU][,AUT])}
		[,STATUS={NOHOLD|HOLD|AUTO }
		{(NOHOLD[,IGNACT])}
		[,STORE=({SMALL|LARGE}[,*nnnnn*])]
		[,THRESH={0|*n*}]
		[,TOF={YES|NO}]
		[,TRAN=*cccccccc*]
		[,UAPL=*cccc*]
		[,XKEYS={NO|YES}]

Programming Notes:

cccccccc
Shows a character string of 1 to 8 characters according to the assembler language naming conventions.

label
For TYPE=ENTRY calls of DSLFNT, the label is disregarded.

TYPE
TYPE=ENTRY is specified for a function table entry definition. The default is ENTRY.

CHECK
The CHECK parameter specifies whether a message checking operation is carried out by the DSLCXT transaction or by the batch programs DSLSDI, DSLSDO, and DSLSDY, and by the **eom** command during the processing by end users. The default for the CHECK parameter is YES.

CHECKAUT
The CHECKAUT parameter specifies an additional authentication checking of SWIFT input messages during data entry. With CHECKAUT=YES it is checked whether there is a valid sending key in the authenticator-key file for the actual date. The default for the CHECKAUT parameter is NO.

COPY
The name of a copy queue for forms.

DE
Whether the respective function allows data entry. The default is NO.

DESCR
A description for the function table entry. The description is displayed to the user in the Function Selection menu when working with MERVA ESA at a display station.

The literal can be up to 50 characters long. If this length is exceeded, only 50 characters are used in the Function Selection menu.

DQFILL
Whether a queue is displayed to a MERVA ESA operator who issues the **dq filled** command. The default is YES.

EXPAND
Which type of field expansion is requested for this function; this parameter is used with the SWIFT address expansion of the SWIFT Link. Address expansion is only performed by DSLEUD and DSLCXT. The meaning of the specification is:

NO
No field expansion is carried out in this function. The default for the EXPAND parameter is NO.

CLEAR
The expansion fields are deleted before a message is displayed in this function.

COND
The field expansion is carried out only when the address field does not contain data, or when it was modified. The second subparameter shows the processing for NOPROMPT.

COND,UNCOND in NOPROMPT specifies an UNCOND expansion.

UNCOND
The expansion is always carried out for each address field.

This specification is interpreted by the MERVA ESA checking and expansion transaction DSLCXT and by the expansion exit programs defined within MERVA ESA Message Format Service.

EXPNAM
Which type of nicknames for field expansion is requested for this function; this parameter is used with the SWIFT address expansion of the SWIFT Link. You can specify:

NO
No nicknames are allowed for field expansion. The default for the EXPNAM parameter is NO.

COMMON
Nicknames defined in a file available to all users (COMMON) may be used. For example, in SWIFT address fields you can enter a nickname which is, during expansion, replaced by the appropriate SWIFT address.

PRIVATE
Nicknames defined in a file available only to the specific user (PRIVATE) may be used.

If PRIVATE,COMMON or COMMON,PRIVATE is specified, PRIVATE and COMMON nicknames may be used.

This specification is interpreted by the expansion exit programs defined within the MERVA ESA Message Format Service, for example by DWSMX001 of the SWIFT Link.

FRAME
The name of the message identifications for the panel frames:

- The first subparameter specifies the message identification for the top frame. If this subparameter is omitted, no top frame is used.
- The second subparameter specifies the message identification for the bottom frame. If this subparameter is omitted, no bottom frame is used.

The frames are displayed on top or bottom of each screen, or print display page. The message identification must also be specified in the message type table where an MCB name is connected to the message identification.

The default value is (0TOP,0BOT). The MERVA ESA standard frame MCBs DSL0TOP and DSL0BOT are defined for message identifications 0TOP and 0BOT, respectively.

FOUREYE
Whether the *four eyes principle* (that is, that two persons must authorize an action) is enforced when users process messages.

OFF	The four eyes principle is not enforced. This is the default.
PREVIOUS	A user who processed as the last user a message in the previous queue is not allowed to store (if the second subparameter is STORE) or to get (if the second subparameter is ANY) that message in the current queue.
ALL	A user who at any time previously processed a message is not allowed to store (if the second subparameter is STORE) or get (if the second subparameter is ANY) that message in any queue for which this value is specified.

The second subparameter applies only for FOUREYE=PREVIOUS or FOUREYE=ALL. It shows which kind of access is allowed:

STORE The user is allowed to get the message, but is not allowed to store it to any queue. That is, the user is allowed to look at the message. Then the only allowed command is ESCAPE. This is the default.

ANY The user is neither allowed to store the message nor to look at it.

Note: The MERVA ESA queue test commands (COPY, DELETE, ...) do not check the FOUREYE parameter setting.

INTQUE
The name of the intermediate queue that can be used by transaction programs to provide for a restart if processing could not be completed. If the value is longer than 8 characters, the name is truncated.

KEEPMSG
Whether messages are kept in this function after printing the message queue. The default is NO.

KEY1
A description of the first key to be used for the queue of the respective function. This parameter is evaluated only if QUEUE=YES is specified:

- The first element of the argument list specifies the name of a TOF field that contains the key. If this parameter is specified, queue management retrieves key1 from the TOF during all PUT requests. If this parameter is omitted or if the value is too long, the name is set to blanks and the key must be supplied by the program calling DSLQMGT.
- The second element of the argument list specifies the length of the key. A value from 1 to 24 can be specified. If this parameter is specified, the queue is processed with key1. The default is 0; that is, key1 is not used for this queue.
- The third element of the argument list specifies the number of the character in the TOF field which is to be used as the first character of the key (offset *p* in data of the TOF field). The default is 1; that is, the first character of the TOF field is also the first character of the key.
- The fourth element of the argument list specifies that the key must be used without any modification. If NOMOD is omitted, queue management can modify the key, for example remove trailing blanks.

If a MERVA ESA API program requires unmodified keys for storing and retrieving of messages, it must enable the usage of unmodified keys. It must set the character '1' to the field APICQBIN using the API function FLDP. For details refer to the *MERVA for ESA Application Programming Interface Guide*.

KEY2
Describes the second key to be used for the queue of the respective function:

- The first element of the argument list specifies the name of a TOF field that contains the key. If this parameter is specified, queue management retrieves key2 from the TOF during all PUT requests. If this parameter is omitted or if the value is too long, the name is set to blanks and the key must be supplied by the program calling DSLQMGT.

- The second element of the argument list specifies the length of the key. A value from 1 to 24 can be specified. If this parameter is specified, this queue is processed with key2. The default is 0; that is, key2 is not used for this queue.
- The third element of the argument list specifies the number of the character in the TOF field which is to be used as the first character of the key (offset *p* in data of the TOF field). The default is 1; that is, the first character of the TOF field is also the first character of the key.
- The fourth element of the argument list specifies that the key must be used without any modification. If NOMOD is omitted, queue management can modify the key, for example remove trailing blanks.

This parameter is evaluated only if QUEUE=YES is specified.

If a MERVA ESA API program requires unmodified keys for storing and retrieving of messages, it must enable the usage of unmodified keys. It must set the character '1' to the field APICQBIN using the API function FLDP. For details refer to the *MERVA for ESA Application Programming Interface Guide*.

LTERM
The logical terminal name. If LTERM is specified, TRAN must also be specified. If the value is longer than 8 characters, the name is truncated.

Note: When MERVA ESA is running in a CICS environment, the name can only be 4 characters long.

MODE
The display format of a message for screen or printer.

SW2	The header and trailer of a message is to be shown in SWIFT II format. This is the default.
SW1	The header and trailer of a message is to be shown in SWIFT I format.

MQI
The usage of a function associated with a transaction.

NO	The function is not used as a MERVA-MQI Attachment function. This is the default.
YES	The function is used as a MERVA-MQI Attachment function. For such functions, the operator commands CF, HF, and SF become restricted commands; that is, they are accepted only when issued either by a MERVA ESA master user or a MERVA-MQI Attachment user (see the *MERVA for ESA User's Guide* for details).

MSGID
The message identification for the cover MCB used in the specified function. For more information on cover MCBs refer to the *MERVA for ESA Customization Guide*. The default value is 0COV.

MSGLIM
For IMS only, the first subparameter specifies how many messages a MERVA ESA transaction program can process in a single scheduling before control is returned to IMS. This parameter is used with the MERVA ESA expansion and checking transaction DSLCXT, the MERVA ESA printing program DSLHCP, and the MERVA-MQI Attachment transactions DSLKQR and DSLKQS.

This value overwrites the value specified by the MSGLIM parameter in DSLPARM.

A number between 1 and 65535 can be specified for *nnnnn*. Code a value of 65535 if no limit is to be placed upon the number of messages processed at a single scheduling. If the parameter is omitted, the MSGLIM parameter defined in DSLPARM is referred to by the transaction programs DSLCXT, DSLHCP, DSLKQR, and DSLKQS.

For IMS only, the second subparameter specifies whether checkpoints are to be written by the MERVA ESA hardcopy printing program DSLHCP. Checkpoint writing is necessary in order not to loose messages in case of an abnormal end of DSLHCP.

CHKP is the default value.

This parameter has a similar meaning to the PROCLIM parameter for the IMS system definition. It is the best if you use this technique also for user transactions to avoid performance problems.

MSGSEL
MSGSEL=LIST specifies for message processing functions, that after message selection the Queue Key List panel is displayed instead of the Message Selection panel.

NAME
The name of the message-processing function. If the value is longer than 8 characters, the name is truncated. DSLQMGT uses this name as the queue name.

NEXT
The name of the next function. If the value is longer than 8 characters, the name is truncated.

The function defined by the respective entry passes a message to the next function if there was no routing table named or if the evaluation of the routing table resulted in an error.

NOPR
The type of NOPROMPT message processing.

DISPLAY	With this type, message display in NOPROMPT mode is allowed but editing is not allowed. This is the default.
YES	The NOPROMPT command including editing is allowed in this function.
NO	The NOPROMPT command is not supported.

Note: Editing in NOPROMPT mode is very dangerous and should be allowed only for verifying or error-correcting functions.

NOTIFY
Whether a function name should receive a notify indicator on a queue display. YES defines that character * is displayed in the Operator Command Processing panel. The default is NO.

PARTID
A two-character partition ID for a queue (only for queue management using DB2®; see the QIO parameter of DSLPARM).

The columns PARTID, MSGTABLENO, and SEQNO form the key of the DB2 table DSLTQMSG. The partition ID can be used to define a partitioned index on DB2 table DSLTQMSG. If not specified, the two leftmost characters from the queue name will be used as partition ID.

PFGROUP
The program function key group number for this function. This number can be referred to in the program function key definition macro DSLMPFK.

PFGROUP has the following defaults depending on the specification or default of the PROGRAM parameter (do not use any other PFGROUP numbers for the specified programs):

8	For PROGRAM=DSLECMD
12	For PROGRAM=DSLEUSR
16	For PROGRAM=DSLEMSG
20	For PROGRAM=DSLEFLM
24	For PROGRAM=DWSEAUT
0	For all other values of the PROGRAM parameter

PFKSET
The name of a program function key table used for this function, if the user profile record of the user does not contain a program function key table specification.

A program function key table is generated with the DSLMPFK macro.

PRFORM
Two printer format indicators:

- The first element of the argument list specifies the 1-byte format identification. The format identification corresponds to the ID parameter (language ID) in the DSLLDEV macro of the MCB. If the specified value is not in the MCB, the ID in the first DSLLDEV macro of the corresponding TYPE is assumed. An alphanumeric value from A to Z or 0 to 9 can be specified. If the value is omitted or outside this range, a default value is assumed depending on the value specified for second element (the compression format):
  - For compression format 0, 1, 2, or 3, E is the default.
  - For compression format 4, X is the default.
  - If the compression format is omitted or outside the range 0 through 4, 0 is the default.
- The second element of the argument list specifies the 1-byte compression format:

  0 Prints the message without compression; it is printed according to the layout defined in the hardcopy printer section of the MCB. That means that all lines and their literals are printed even if certain lines do not contain data (PROMPT mode).

  1 Prints the message with field compression; only those fields are printed that contain data. However, empty data areas of a field are printed as blank lines (PROMPT UNIT mode).

  2 Prints the message with line compression; those lines of the message layout are printed that either contain data or a literal. Empty data areas are not printed, however, the literals of empty lines or fields are printed (PROMPT LINE mode).

  3 Prints the message with field and line compression; only those fields are printed that contain data. Empty data areas of a field are not printed (PROMPT UNIT LINE mode).

4 Prints the message in NOPROMPT format.

PRINT
The name of the hardcopy print function. If the value is longer than 8 characters, the name is truncated. This name is used for the **hco** command.

PROGRAM
The name of the MERVA ESA processing program used for this function. The following standard program names may be used for this parameter:

DSLEMSG	For all message-processing functions. This is the default.
DSLECMD	For MERVA ESA operator command processing functions.
DSLEUSR	For MERVA ESA user file maintenance functions.
DSLEFLM	For MERVA ESA general file maintenance functions.
DWSEAUT	For the authenticator-key file maintenance functions.

This parameter lets you specify your own names for the MERVA ESA standard functions. For example, specifying

```
DSLFNT NAME=COMMAND,PROGRAM=DSLECMD
```

assigns the function name COMMAND to the operator command processing function.

PROT
All fields of the end user panels are protected, a user processing a message under such a function cannot change the message. The default is NO. In the online maintenance of the user file and the authenticator-key file, PROT=NO means that the user is allowed to make updates to the file, PROT=YES means that the user can only display records.

QUEUE
Whether a function has a queue (QUEUE=YES or QUEUE=NO). QUEUE=DUMMY defines a dummy queue for a function. A dummy queue can be used to get rid of messages during routing as no queue element is created. Yet associated transactions are started and ECBs are posted (see "DSLQMG: Defining Queue Management Services" on page 199). QUEUE=NO is the default.

RELATED
One or two function names that have the same TRAN parameter and that are to be processed as a set by this transaction. Related functions are always set to the same function status (ACTIVATED, HOLD, NOHOLD). The one or two functions specified must also use the RELATED parameter with the other one or two functions.

RETYPE
Whether the respective function is a retype verification function. The default is NO.

ROUTE
The name of the routing table. The specified routing table is used to determine where a message is to be routed when the function defined by the respective entry is ready to pass on the message. If the value is longer than 8 characters, the name is truncated.

The routing tables are loaded during MERVA ESA startup and must be contained in the library and in the CICS program definition.

SPCMND
Used to restrict user commands. Restriction means that users are allowed to enter these commands only if the message-processing function under which they are working allows the command in the SPCMND parameter. For descriptions of the user commands, see the *MERVA for ESA User's Guide*. The following list shows the values you must specify for the SPCMND parameter to enable using the corresponding commands:

DEL For the **delete** command, which is used to delete messages in the input queue.

OK For the **ok** command, which is used to show that the message is correct or, for the user file and the authenticator-key file online maintenance, to enable updated records to be authorized.

ROU For the **route** command, which is used to route the message to the specified function. If ROU is specified, the ROUTE parameter must specify a routing table that evaluates the MSGOK field.

AUT For the **authent** command, which is used to authenticate an output message (SWIFT Link only).

For example:

```
SPCMND=(AUT,DEL,OK,ROU)
```

STATUS
The initial status of functions for which a transaction name is specified (TRAN parameter):

NOHOLD
Causes a transaction associated with the function to be started automatically when a message is written to the queue, and the status is not ACTIVATED. This is the default.

IGNACT
For IMS and STATUS=NOHOLD only, IGNACT can be used to ignore the ACTIVATED status, that is, the transaction is started each time a message is written to the queue.

HOLD
Prevents a transaction associated with that function from being started automatically when a message is written to the queue. The function is held until released by the operator via the MERVA ESA SF (start function) command.

AUTO
Causes a transaction associated with the function to be started automatically when MERVA ESA is started.

STORE
The handling of messages depending on their size when they are stored in a queue. If the STORE parameter is not used, the handling of messages depends on the specification of the LRGMSG parameter of the DSLPARM macro.

SMALL
Indicates to DSLQMGT that large messages are not stored in this queue even if LRGMSG=YES is specified in DSLPRM.

If the length value *nnnnn* is not specified, the length of a small message is limited to 31900 characters (this is the space available in a queue data set block).

If the length value *nnnnn* is specified, the length of a small message is limited to this length.

The SMALL parameter helps to avoid routing of messages into a queue that is processed by a program that does not want to handle large messages or messages in the MERVA ESA internal format (TOF format). The storing of too long messages is rejected by DSLQMGT with the return code "data too long".

LARGE
Indicates to DSLQMGT that large messages can be stored in this queue if LRGMSG=YES is specified in DSLPRM. The length value *nnnnn* can specify an individual limit for a queue depending on which DSLQMGT stores the message in the queue data set (QDS) or in the large message cluster (LMC).

When routing messages into several queues with different STORE parameters, STORE=SMALL usually decides about accepting or rejecting the request, or STORE=(LARGE,*nnnnn*) decides about storing the message in the QDS or LMC.

Note: For queue management using DB2 only (see the QIO parameter of DSLPARM), there is no QDS or LMC. Messages are stored if their length is not greater than specified in the STORE parameter.

THRESH
The threshold number of queue elements.

If this number is reached, a message is issued to the MERVA ESA operator. The queue display indicates that the threshold condition was reached, thus indicating that an operator or user action is required. The queue accepts messages if there is room in the queue data set.

A value from 0 to 32700 can be specified. If a greater value is specified, 32700 is used. If 0 (the default) is specified, no threshold processing takes place.

TOF
Whether messages in a queue are stored in the MERVA ESA internal format. The default is YES.

Queues with the TOF=NO parameter:

- Cannot be processed by the normal function programs of MERVA ESA
- Are used by the SWIFT Link to store pregenerated session keys when the SWIFT secure login/select is used, and to store information such as login, select, input and output sequence numbers between a SWIFT Link termination and the next startup

TRAN
The name of a transaction to be started when a message is inserted into the queue of the respective function if the function is in NOHOLD status.

This name can refer to an entry in the MERVA ESA transaction table DSLTXTT. The type of transaction connected to this name is defined in the entry of the DSLTXTT.

When the name is not defined in the DSLTXTT the transaction is started in the local DC system.

Transaction names to be used in a CICS environment can only be 4 characters long.

If the value is longer than 8 characters, the name is truncated.

UAPL
Defines a string of up to 4 characters to be used as parameter for a user application started when the queue is filled.

This parameter is used by the MERVA ESA checking and expansion transaction program DSLCXT.

When a numeric value is specified for UAPL, it is used as the number of an MFS user exit to be called for each message in addition to the standard MFS user exit 23 (DSLMU023). The interface for the specified user exit is identical to the interface for MFS user exit 23.

The program DSLCXT is also able to convert messages to the external line format according to the specification in the UAPL parameter.

The following variants are supported:

- Convert tokenized format to external format and replace the tokenized format (UAPL=ELF)
- Convert tokenized format to external format and add it to the tokenized format (UAPL=ELF+)
- Convert external format to tokenized format and replace the external format (UAPL=TOF)
- Convert external format to tokenized format and add it to the external format (UAPL=TOF+)

XKEYS
For queue management using DB2 only (see the QIO parameter of DSLPARM), specifies that extra keys are stored for this function. Extra keys are defined in DB2 table DSLTQXDEF, the actual values are stored in DB2 table DSLTQXKEY. The default is NO.

Mapping a Function Table Entry

Name	Operator	Operands
[*label*]	**DSLFNT**	**TYPE=MAP** [**,PREFIX=**{<u>**FNT**</u>|*ccc*}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The map of the function table entry is to be generated. If TYPE=MAP is specified, all other parameters except PREFIX are ignored.

PREFIX
The 3-character prefix of the variable names of the function-table entry map. The default is FNT.

DSLGEN: Generating the MERVA ESA System

This macro is a system-generation macro and is not part of a programming interface.

The DSLGEN macro is used to generate the tables that control the MERVA ESA installation environment. DSLGEN also allows for adding user definitions.

There are four types of DSLGEN macros:

TYPE=INITIAL	The start of DSLGEN processing and which MERVA ESA components are used. This must be the first statement.
TYPE=USER	Additions to the MERVA ESA tables.
TYPE=FINAL	The end of DSLGEN processing. This macro may be followed by a DSLGEN TYPE=CLINK macro.
TYPE=CLINK	VSE only, specifies that DSLGEN is to generate link-edit jobs for CICS dependent modules. This type can be used without other DSLGEN macros (it then must contain the same parameters as the DSLGEN TYPE=INITIAL) or after the DSLGEN TYPE=FINAL (then no other parameters must be specified).

For details refer to the *MERVA for ESA Installation Guide*. The following macros are used by the DSLGEN process:
- DSLGENS
- DSLGENC

DSLGRP: Defining the Group Table Entry

The group table can be used to categorize users in the user file. Whether a group table should be used is controlled by the parameter USGRP in the MERVA ESA customizing parameter module DSLPRM.

The DSLGRP macro has two functions:

- Generate a group table
- Map the group table entry

The group table definition consists of a sequence of DSLGRP macros. MERVA ESA provides a sample group table DSLGRPT. The last statement of the table must be the END statement.

If errors are detected in a specific DSLGRP macro, appropriate MNOTEs are provided. For severe errors, the macro does not generate a table entry.

Generating a Group Table

Name	Operator	Operands
[*label*]	**DSLGRP**	**TYPE=**{**INITIAL**|**FINAL**}

Programming Notes:

label
: A unique statement label according to assembler language conventions.

TYPE
: TYPE=INITIAL must be the first group table definition statement.
TYPE=FINAL must be the last group table definition statement.

Generating a Group Table Entry

The definitions for one group consist of a sequence of one TYPE=ENTRY and up to 1000 TYPE=FUNCTION statements.

Name	Operator	Operands
[*label*]	**DSLGRP**	**TYPE=ENTRY** **,GROUPID=***cccccccc* **,ORIGINID=***cccccccccccc* [**,DESCR=***quoted literal*] [**,USERPREF=***cccccccc*]

Programming Notes:

label
: For TYPE=ENTRY calls of DSLGRP, the label is disregarded.

TYPE
: The type of MERVA ESA group table entry to be generated. TYPE=ENTRY indicates that a table entry is to be generated.

GROUPID
The name of the group. If the value is longer than 8 characters, the name is truncated.

ORIGINID
The origin ID. If the value is longer than 12 characters, the name is truncated. The origin ID of each user belonging to the group must start with these characters. This will be checked in the user file maintenance.

DESCR
A description for the group table entry. The literal can be up to 50 characters long. If this length is exceeded, no description will be stored.

USERPREF
A user ID prefix. If a value is longer than 8 characters, the prefix is truncated. If specified, the user ID of each user belonging to the group must start with these characters. This will be checked in the user file maintenance.

Name	Operator	Operands
[*label*]	**DSLGRP**	[**TYPE=FUNCTION**] **,FUNCTION=**cccccccc

For each TYPE=ENTRY, you can specify up to 1000 TYPE=FUNCTION statements.

Programming Notes:

label
For TYPE=FUNCTION calls of DSLGRP, the label is disregarded.

TYPE
The type of MERVA ESA group table entry to be generated. TYPE=FUNCTION indicates that a function for a group is to be generated. This parameter is optional.

FUNCTION
A function name. If a function name is longer than 8 characters, the value is truncated. The function names can contain the following wildcards:

%	Matches any single character
*	Matches any number of characters, including no characters

In the user file maintenance you can specify that a user should belong to a group. You can then assign only those functions to a user that have been allowed for that group in DSLGRPT. For example, the function names L1DE0 and L1VE0 in the user file maintenance match with L1%%0 defined in the group table.

Mapping a Group Table Entry

Name	Operator	Operands
[*label*]	**DSLGRP**	**TYPE=DSECT**

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
Generates DSECTs of the header and entry sections. With TYPE=DSECT all other parameters are ignored.

DSLJRN: Defining the Journal Service

The DSLJRN macro serves the following purposes:

- Prepare the parameter list to read and write journal records using the MERVA ESA journal program DSLJRNP (the EP operand determines if the request will be processed as a direct or central service)
- Map the parameter list of DSLJRNP

Calling the Journal Program

DSLJRNP reads and writes records to the MERVA ESA journal file.

Read requests are carried out by a VSAM key, using a search argument key equal, key greater, or key equal or greater. The VSAM key is established in the DSLJRNP parameter list.

The journal record content is described in *MERVA for ESA Concepts and Components*.

The parameter list is also used for internal requests by the MERVA ESA system program DSLNUC to initialize and terminate the journal program and by the command routine performing the switching of journal data sets.

Name	Operator	Operands
[*label*]	**DSLJRN**	**TYPE=**{**PUT**|**GET**}
		,DATA={*addr*|(*r*)}
		[**,EP=DSLJRNP**]
		[**,GETOPT=**{**KGT**|**KEQ**|**KGE**}]
		[**,JID=**{*nnn*|*addr*|(*r*)}]
		[**,KEYARG2=**{*addr*|(*r*)}]
		[**,KEYDATE=**{*addr*|(*r*)}]
		[**,KEYEXT2=**{*addr*|(*r*)}]
		[**,KEYSEGC=**{*addr*|(*r*)}]
		[**,KEYTIME=**{*addr*|(*r*)}]
		[**,KEYTIMEX=**{*addr*|(*r*)}]
		[**,MF=(E,**{*addr*|(*r*)})]
		[**,UKEY=**{*addr*|(*r*)}]

Programming Notes: For the address parameters in this macro, the following notations are possible:

addr	stands for a symbolic label
(r)	stands for a general register containing the address.

label
A unique statement label according to assembler language conventions.

TYPE
The MERVA ESA journal function to be performed:

PUT Writes a record to the journal file.

GET Reads a record from the journal file.

DATA
The label of the data buffer or a general register containing the address of the buffer.

For TYPE=PUT, the address shows the location where DSLJRNP finds the data to be written to the data portion of the journal record.

For TYPE=GET, the address shows the buffer where the data portion of the retrieved record is to be moved for use by the calling program.

The data buffer must have the standard MERVA ESA format which is described in the *MERVA for ESA Customization Guide*, in the chapter dealing with the buffer standards of MERVA ESA. Note that the first 4 bytes of this buffer are not moved to the journal record.

The header portion of the journal record, including the VSAM key and user extension, is found in the DSLJRNP parameter list.

EP
Must specify DSLJRNP only. It defines that the request will be executed as a direct service.

The EP parameter must be specified by all programs that are linked to DSLNUC. Before issuing the DSLJRN macro with EP=DSLJRNP, the program must ensure that general register 12 contains the address of the MERVA ESA communication area DSLCOM and that general register 13 points to a usable save area.

Use the parameter EP=DSLJRNP only for programs that are linked to DSLNUC. After using the DSLJRN macro to prepare the parameter list, a DSLNIC macro with TYPE=REQ and NAME=DSLJRNP is required to request execution of a central service.

If the EP parameter is omitted or invalid, a central service request is assumed.

GETOPT
For TYPE=GET only, specifies the key search argument.

KGT The key that is retrieved is the next-higher key in the file from the key provided by the caller. This is the default.

KEQ The key that is retrieved must be equal to the key provided by the caller.

KGE The key that is retrieved must be equal to the key provided by the caller or the next-higher key in the file.

JID
A 1-byte journal record identifier. It may be a decimal value, or the label of an equate statement, or a general register containing a hexadecimal value. The maximum value of JID is 254 (X'FE'). Journal record identifiers used by MERVA ESA can be found in the DSLJRNP parameter list.

KEYARG2
For TYPE=GET only, specifies the label of a field or a general register containing the address of the search argument used for accessing the journal file.

The length of the search argument is 20 bytes. The format of the search argument for a journal file using a 4-digit year format is:

```
yyyymmddhhmmssfffggg
```

where:

yyyy	The 4-digit year
mm	The 2-digit month (01 to 12)
dd	The 2-digit day of the month (01 to 31)
hhmmss	The time of the day (hours, minutes, seconds)
fff	The fractional part (between 000 and 999) which is used to make the keys unique if more than one journal record was created during the same second.
ggg	Either blanks, or for segmented records the segment count (001 to 999)

KEYDATE
For TYPE=GET only, specifies the label of a 6-byte field containing the date for the record key, in the form YYMMDD, or a general register containing the address of the date field. This format should no longer be used. When using the 4-digit year format in the journal file the parameter KEYARG2 should be used instead.

KEYEXT2
The label of a field that contains user extension data for the journal record VSAM key, or a general register containing the address of the key extension field.

The maximum length of the user extension is 25 bytes. If a field label is used, the data moved has the same length as the field that is referenced. If register notation is used, 25 bytes are moved. This data is used to extend the significant characters of the VSAM search key or to provide data for a user-defined alternate index.

If this parameter is omitted, the user extension data in the DSLJRNP parameter list remains unchanged.

KEYSEGC
For TYPE=GET only, specifies the label of a 3-byte field containing the segment number of the record key (three numeric digits) or a general register containing the address of the segment number field. This parameter can be used when segmented records are processed. This format should no longer be used. When using the 4-digit year format in the journal file the parameter KEYARG2 should be used instead.

KEYTIME
For TYPE=GET only, specifies the label of an 8-byte field containing the time for the record key, in the form HHMMSSCC or a general register containing the address of the time field. This format should no longer be used. When using the 4-digit year format in the journal file the parameter KEYARG2 should be used instead.

KEYTIMEX
For TYPE=GET only, specifies the label of a 10-byte field containing the time for the record key, in the form HHMMSSCCCC or a general register containing the address of the time field. This parameter is used when the extended time stamp with segmented records is used. KEYTIMEX and KEYTIME are mutually exclusive. This format should no longer be used. When using the 4-digit year format in the journal file the parameter KEYARG2 should be used instead.

MF
The format of the macro.

The first parameter must be E for execute form. The second parameter specifies the label of the DSLJRNP parameter list or a general register containing the address of the parameter list. The default is (E,(1)).

UKEY
The label of a field that contains user extension data for the journal record VSAM key, or a general register containing the address of the key extension field.

The maximum length of the user extension is 25 bytes. If a field label is used, the data moved has the same length as the field that is referenced. If register notation is used, 25 bytes are moved. This data is used to extend the significant characters of the VSAM search key or to provide data for a user-defined alternate index.

If this parameter is omitted, the user extension data in the DSLJRNP parameter list remains unchanged. This parameter should no longer be used. When using the 4-digit year format in the journal file the parameter KEYEXT2 should be used instead.

Mapping the Parameter List of DSLJRNP

The list form of the DSLJRN macro is used to generate a storage description of the DSLJRNP parameter list. The parameter list includes a map of the journal record header.

Name	Operator	Operands
[*label*]	**DSLJRN**	**MF=L**

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

DSLKPROC: Defining the MERVA-MQI Attachment Process Table

The DSLKPROC macro has two functions:

- To generate the customization information for the MERVA-MQI Attachment process table
- To map the customization information

The name of the MERVA-MQI Attachment process table is always DSLKPROC. It is defined by coding DSLKPROC statements. A DSLKPROC statement can be of one of the following types:

INITIAL	Defines the beginning of the table
SEND	Defines the MERVA-to-MQI send process
RECEIVE	Defines the MQI-to-MERVA receive process
FINAL	Defines the end of the table
DSECT	Generates an assembler DSECT of the attachment process table

Note: When you assemble the DSLKPROC table, the correct SYSPARM setting according to your operating system is required. See "SYSPARM Requirements" on page 225 for details.

Beginning the MERVA-MQI Attachment Process Table

The DSLKPROC TYPE=INITIAL statement defines the begin of the attachment process table. It is a mandatory statement.

Name	Operator	Operands
	`DSLKPROC`	**TYPE=INITIAL** [,**CCSID=**{<u>**QMGR**</u>|*nnnnn*}] [,**MQMGR=***cccccccc*]

Programming Notes:

TYPE

TYPE=INITIAL must be specified for this form of the macro.

CCSID

The coded character set identifier (CCSID) that identifies a code page. The value can be a number of up to 5 digits, or the string QMGR (the default).

- For a MERVA-to-MQI send process, this parameter identifies the code page of the message data. The value QMGR indicates that the message data uses the code page that corresponds to the CCSID of the MQSeries queue manager.
- For an MQI-to-MERVA receive process, this parameter identifies the code page that is to be used to convert the message data. The meaning of the value QMGR depends on the operating system and the value of the CONVERT parameter of DSLKPROC TYPE=RECEIVE:
  - If the operating system is MVS/ESA™ and CONVERT=YES, the CCSID of the MQSeries queue manager is to be used.
  - If the operating system is CICS/VSE and CONVERT=YES, the CCSID 500 is to be used.
  - If CONVERT=NO, the CCSID of each received message is used.

| The CONVERT parameter is described in "Defining the MQI-to-MERVA
| Receive Process" on page 62.

MQMGR
For IMS and CICS/VSE only, the name of the MQSeries queue manager. The name consists of up to 48 characters.

If this parameter is omitted, the name of the default queue manager is used. For IMS, the names of the available queue managers including the default queue manager are defined in the MQSeries subsystem definition table CSQQDEFV using macro CSQQDEFX (see the *MQSeries for MVS/ESA System Management Guide* for details). For CICS/VSE, the default queue manager is the local queue manager in the CICS partition.

Defining the MERVA-to-MQI Send Process

The DSLKPROC TYPE=SEND statement describes the information used for the message transfer from MERVA ESA to the MQSeries.

Name	Operator	Operands
	DSLKPROC	**TYPE=SEND**
		,ALLSNDQ=([(*mrvsq1*,*mqisq1*)][,...][,(*mrvsq10*,*mqisq10*)])
		,MQICTLQ=*cccccccc*
		,MRVCTLQ=(*cccccccc*[,{**STOP**|**CONTINUE**}])
		,NAME=*cccccccc*
		[**,ACKWQ=***cccccccc*]
		[**,ALTUID=***cccccccccccc*]
		| [**,AUTHENT=**{**YES**|**NO**}]
		| [**,CCSIDP=***nnnnn*]
		[**,CHECK=**{**YES**|**NO**}]
		[**,CNVDEST=***cccccccc*]
		[**,COAWQ=***cccccccc*]
		| [**,CODWQ=**{*cccccccc*|#}]
		[**,COMMIT=**{**10**|*nnnnn*}]
		[**,EXCEPT=**{**NO**|**YES**}]
		[**,EXIT=***nnnnn*]
		[**,FORMAT=**{([{**KCOV**|*cccccccc*}][,{**W**|**P**|**U**|*c*}])|**QUEUE**}]
		| [**,ISNCTLQ=***cccccccc*]
		| [**,JIDRCVD=**{**61**|*cc*}]
		| [**,JIDSENT=**{**60**|*cc*}]
		[**,JRNDGRM=**{**NO**|**YES**}]
		[**,JRNRPLY=**{**NO**|**YES**}]
		[**,JRNRQST=**{**NO**|**YES**}]
		[**,MQFMT=**{(**MQSTR**[,altfmt])|**BLANK**|*fmtname*}]
		[**,MRVSTAQ=**(*cccccccc*[,{**NOHOLD**|**ALL**}])]
		[**,NEXT=**{**STANDARD**|**NEW**|**FORWARD**}]
		[**,OPMSDM=**{**NONE**|**SUBSET**|**FULL**}]
		[**,OPMSJRN=**{**NONE**|**SUBSET**|**FULL**}]
		[**,PASCMID=**{**YES**|**NO**}]
		[**,REPLYTQ=***cccccccc*]

Name	Operator	Operands
		[,SECURE={NO|AUTHENCR}] [,TRACMFS={NO|YES}] [,TRACTOF={NO|YES}]

Programming Notes:

TYPE=SEND
The definition of a MERVA-to-MQI send process begins and a MERVA-to-MQI send process entry must be generated. You can define up to 1000 send processes.

ALLSNDQ
The list of MERVA ESA and MQI send queue pairs.

Each MERVA ESA send queue is associated to one MQI send queue. Several MERVA ESA send queues can be associated to the same MQI send queue. The pair of associated send queues is enclosed in parentheses.

The send queue pairs are specified according to the priority of the respective MQI send queue. Send queue pairs containing a queue with higher priority are specified before pairs containing a queue with lower priority. When a send process is initiated by the MERVA ESA start queue specified in parameter MRVSTAQ, the send queue pairs are processed in the order of their appearance in the list. The list of send queue pairs is enclosed in parentheses.

Up to 10 send queue pairs can be specified. They contain the MERVA ESA send queues *mrvsq1* to *mrvsq10* and the MQI send queues *mqisq1* to *mqisq10*.

The MERVA ESA queue names consist of up to 8 characters, the MQI queue names consist of up to 48 characters. The total length of the send queue pairs is limited to 255 characters.

The MERVA ESA send queue names must be unique within a send process and across all send processes. The MQI send queue names can be the same within a send process and across all send processes.

When a MERVA ESA send queue is defined in the MERVA ESA function table DSLFNTT, the transaction code of the MERVA-to-MQI send process must be specified in parameter TRAN. The default transaction code is DSLS. In addition, MQI=YES must be specified to prevent an unauthorized operator from starting a send process.

At least one send queue pair must be specified.

MQICTLQ
The name of the MQI control queue. The queue must be defined in MQSeries. Its name consists of up to 48 characters.

The name of the MQI control queue can be shared by send processes when the MERVA-MQI Attachment runs under IMS or CICS/ESA. MQI control queues cannot be shared when the attachment runs under CICS/VSE. The name must be unique among the names of the MQI send queues of all send processes and the names of the MQI receive queues of all receive processes.

MRVCTLQ
The name of the MERVA ESA control queue. The queue must be defined with KEY1 and KEY2 in the MERVA ESA function table DSLFNTT. The name can be up to 8 characters.

The MERVA ESA control queue can be shared by send processes. The name of the send process specified in parameter NAME followed by the name of the currently processed MERVA ESA send queue specified in parameter ALLSNDQ are used as KEY1. The MQI MsgId is used as KEY2 to provide for a unique message key. The subparameter NOMOD is required for KEY1 and KEY2.

Except for the shareable MERVA ESA control queues, the name of the control queue must be unique among the names of the MERVA ESA queues of all send processes and receive processes.

The following options apply to the recovery of the current send process:

STOP
This indicates that a message is kept in the MERVA ESA control queue in case of an error during the recovery of the current send process. The send process stops due to the message causing the error. This is the default.

CONTINUE
This indicates that the messages are removed from the MERVA ESA control queue in case of an error during the recovery of the current send process. The messages are routed to an error queue. The send process continues with the next message in the MERVA ESA send queue.

Note: During normal processing the current send process always stops when an error occurs.

The control queue name and the value for the option must be separated by a comma and enclosed in parentheses.

NAME
The name of the current send process. It consists of up to 8 characters, and must be unique for all send processes.

ACKWQ
The name of the MERVA ESA wait queue for messages waiting for an acknowledgment from the receiving application. The acknowledgment is a reply message sent by the receiving application in response to a request message from the current send process. The queue must be defined with KEY1 in the MERVA ESA function table DSLFNTT. The name of the wait queue can be the same as the name of the wait queue specified in parameters COAWQ and CODWQ. The name can be up to 8 characters.

The value of the MQI MsgId is used as KEY1. The subparameter NOMOD is required for KEY1.

The ACK wait queue can be shared by send processes. The reply message can contain:

- The same MQI MsgId as the waiting message
- The name of the current send process as MQI CorrelId

Using this information, the reply message can be correlated with the appropriate message of the respective send process in the ACK wait queue.

If this parameter is omitted, an acknowledgment is not requested by the messages in the current send process. That is, MQI request messages will not be used in this send process.

ALTUID
For MVS/ESA only, the alternate user identifier that is to be used to validate the opening of the MQI send queues of the current send process.

It consists of up to 12 characters. The first 8 characters are used to check the authorization for the open, in place of the user identifier that the send process is currently running under. The current user identifier may not even be authorized to do the open. Context checks, however, are still carried out with the current user identifier.

The current user identifier must also be authorized to specify this particular alternate user identifier. All 12 characters of the alternate user identifier are used for this check. If this parameter is omitted, it is assumed that an alternate user identifier is not needed.

AUTHENT
Whether the FMT/ESA authenticates SWIFT input messages.

YES SWIFT input messages are authenticated. The SWIFT authentication must have been started either automatically or by the command **start**, which starts the authentication initialization program defined in the MERVA ESA nucleus program table DSLNPTT. This is the default.

For CICS/VSE, note that you can also request general message authentication by setting the parameter SECURE=AUTHENCR.

NO SWIFT input messages are not authenticated.

CCSIDP
The coded character set identifier (CCSID) that identifies the code page that describes the message data. The value can be up to 5 digits.

The value of CCSIDP overrides the CCSID specified in DSLKPROC TYPE=INITIAL, but if CCSIDP is omitted, that CCSID is used.

CHECK
The MERVA ESA MFS message checking can be enabled or disabled before a message is mapped from the TOF to an external network format.

YES The MFS message checking is enabled. This is the default.

NO The MFS message checking is disabled.

This allows you to prevent duplicate message checking if the messages to be sent have already been checked in a previous processing step. Furthermore, this option reduces the overhead in the mapping process.

CNVDEST
For CICS/VSE only, the name of the destination platform for messages to be converted before they are sent. It can be up to 8 characters.

The sample attachment conversion exit DSLKCVSE uses this name to determine the conversion parameters for the appropriate destination platform. The exit DSLKCVSE supports the following destination platform names:

- AIX
- AIXBIN
- OS2
- OS2BIN
- WINNT
- WINNTBIN

See the *MERVA for ESA Customization Guide* for details.

COAWQ

The name of the MERVA ESA wait queue for messages waiting for a confirm-on-arrival (COA) report sent by the local or remote MQSeries. The queue must be defined with KEY1 in the MERVA ESA function table DSLFNTT. The name of the wait queue can be the same as the name of the wait queue specified in parameters ACKWQ and CODWQ. The name can be up to 8 characters.

The value of the MQI MsgId is used as KEY1. The subparameter NOMOD is required for KEY1.

The COA wait queue can be shared by send processes. The COA report can contain:

- The same MQI MsgId as the waiting message
- The name of the current send process as MQI CorrelId

Using this information, the COA report can be correlated with the appropriate message of the respective send process in the COA wait queue.

If this parameter is omitted, a COA report is not requested by the messages in the current send process.

CODWQ

The name of the MERVA ESA wait queue for messages waiting for a Confirm-on-delivery (COD) report sent by the local or remote MQSeries. The queue must be defined with KEY1 in the MERVA ESA function table DSLFNTT. The name of the wait queue can be the same as the name of the wait queue specified in parameters ACKWQ and COAWQ. The name can be up to 8 characters.

For FMT/ESA only, you can specify the character # to indicate that the receiving FMT/ESA is to generate a delivery notification (MT 011), and that a COD report is not to be generated.

The value of the MQI MsgId is used as KEY1. The subparameter NOMOD is required for KEY1.

The COD wait queue can be shared by send processes. The COD report can contain:

- The same MQI MsgId as the waiting message
- The name of the current send process as MQI CorrelId

Using this information, the COD report can be correlated with the appropriate message of the respective send process in the COD wait queue.

If this parameter is omitted, or if the character # is specified, a COD report is not requested by the messages in the current send process.

COMMIT

The frequency of commits for changes on the MQI queues. Committing indicates that the changes on the MQI queues can be made permanent.

The value must be between 1 and 32767. A commit frequency of 1 means that a commit is made after each message contained in a MERVA ESA send queue is processed. The default is 10.

EXCEPT

For MVS/ESA only, specifies whether exception reports are requested from messages of the current send process.

An exception report is generated by a message channel agent when a message is sent to the remote MQSeries and the message cannot be delivered to the specified destination queue. The report can only be generated if the undeliverable message has been put on the remote MQI dead-letter queue.

Exception reports are received in the Reply-to queue specified in parameter REPLYTQ. An exception report can only be correlated with the appropriate message when the message is waiting either for a reply or for a report to come. That is, either a reply message containing an acknowledgment, a Confirm-on-arrival report, or a Confirm-on-delivery report must have been requested for the message in addition. If only an exception report was requested, the correlation with the appropriate message must be done outside the MERVA-MQI Attachment.

NO Exception reports are not requested by the messages of the current send process. This is the default value.

If the messages are sent to MQSeries for VSE/ESA, specifying NO is required.

YES Exception reports are requested by the messages of the current send process.

EXIT

The number of an MFS user exit that can be used for the following purposes:

- Prepare an MQI reply message (the reply message will contain the data provided)
- Provide an MQI datagram or request message with data that is not contained in the message itself (this data will be sent together with the message)

The exit number must be specified in the MERVA ESA MFS program table DSLMPTT. The DSLMPTT must be link-edited to the module DSLMMFS.

The value must be between 1 and 32767 (inclusive), and cannot be a number that is reserved for MFS exits of other MERVA ESA components. If this parameter is omitted, it is assumed that a user exit is not needed.

FORMAT

The MERVA ESA format control information for the messages in the current send process.

The messages can be formatted in MERVA ESA external line format or in queue format:

- When **external line format** is requested, the first subparameter specifies the message identifier of a MERVA ESA MCB. It consists of up to 8 characters. The message identifier identifies the MCB via the message type table DSLMTTT.

  In order to process a wide range of different messages you can specify the message identifier of an appropriate MERVA ESA cover MCB. Examples for cover MCB message identifiers are:

Message Identifier	Cover MCB	Description
KCOV	DSLKCOV	Used for the MERVA-MQI Attachment (the default)
0COV	DSL0COV	Sample for SWIFT and telex messages
MCOV	EKAMCOV	Used for the MERVA Link
TCOV	ENLTCOV	Used for the Telex Link

If this subparameter is omitted, a blank message identifier is assumed, and the MERVA ESA message type determination exit is used to determine the message type.

The second subparameter specifies the format code that is to be used to format the message in the MCB identified by the first subparameter:

W	SWIFT message (the default)
P	Telex message
U	User-defined message
c	User-defined format code

This format code must be the same as that specified for the ID parameter of the DSLLDEV macro.

- When **queue format** is requested, the value **QUEUE** must be specified.

ISNCTLQ
The name of the ISN control queue used by the FMT/ESA. The queue must be defined with KEY1 and KEY2 in the MERVA ESA function table DSLFNTT.

The name of the send process is used as KEY1. The value of the MQI MsgId is used as KEY2. The subparameter NOMOD is required for KEY2.

This parameter is required if the FMT/ESA user exit is to be called, that is, if the value of the EXIT parameter is 8044.

JIDRCVD
The identifier of the MERVA ESA journal record that is written for a message received from the sending FMT/ESA. Received messages are:

- SWIFT output messages
- SWIFT input message acknowledgments

Note: Such acknowledgments are not always received from the remote FMT/ESA, but can also be generated by the local FMT/ESA; however, they are still recorded in the MERVA ESA journal so that both these acknowledgments and acknowledgments received by SWIFT Link are treated consistently.

The value is one or 2 hexadecimal characters. The default is 61.

JIDSENT
The identifier of the MERVA ESA journal record that is written for a message sent to the receiving FMT/ESA. Sent messages are:

- SWIFT input messages
- SWIFT output message acknowledgments

Note: In the FMT/ESA, these acknowledgments are not sent; however, they are still recorded in the MERVA ESA journal so that both these acknowledgments and acknowledgments sent by SWIFT Link are treated consistently.

The value is one or 2 hexadecimal characters. The default is 60.

JRNDGRM
Whether those messages which are sent as MQI datagrams are written to the MERVA ESA journal.

NO Datagrams are not written to the journal. This is the default.

YES Datagrams are written to the journal.

JRNRPLY
Whether those messages which are sent as MQI replies are written to the MERVA ESA journal.

NO Replies are not written to the journal. This is the default.

YES Replies are written to the journal.

JRNRQST
Whether those messages which are sent as MQI requests are written to the MERVA ESA journal.

NO Requests are not written to the journal. This is the default.

YES Requests are written to the journal.

MQFMT
The name of an MQI data format.

This is the name that the current send process uses to indicate to the receiver the nature of the data in the message. The names of MQI built-in formats known by the queue manager and application defined formats can both be specified. Names beginning with ''MQ'' represent built-in formats.

MQSTR indicates that the message consists of character data only. This is the value of the built-in format MQFMT_STRING.

The option BLANK indicates that the format name consists of 8 blanks. This is the value of the built-in format MQFMT_NONE. If you want to use the application-defined format name BLANK, include the option in parenthesis and specify (BLANK).

The application-defined format name *fmtname* consists of up to 8 characters.

altfmt represents an alternate format name. An alternate format name is used by the current send process instead of the format name MQSTR if the message consists of binary and character data. The alternate format name BLANK should not be specified. A format name consisting of 8 blanks inhibits the use of a data-conversion exit. *altfmt* consists of up to 8 characters.

The default is MQSTR.

Notes:

1. In some cases the format name used by the MERVA-MQI Attachment differs from the name specified in the process table. This does not occur, however, if *altfmt* is specified. For details refer to the *MERVA for ESA Customization Guide*.
2. When an MQI data-conversion exit is used either at the sending or receiving side, *fmtname* or *altfmt* must be equal to the name of the data-conversion exit.

MRVSTAQ
The name of the MERVA ESA start queue. The name can be up to 8 characters.

The start queue can be used to initiate one or more of the MERVA-to-MQI send processes. A send process can be initiated by entering the MERVA ESA operator command SF (Start Function) followed by the name of the start queue, or automatically during MERVA ESA startup. The queue must be defined in the MERVA ESA function table DSLFNTT as a dummy queue (parameter QUEUE=DUMMY) used in the MERVA-MQI Attachment (parameter MQI=YES) with the transaction code for the send process (parameter TRAN=DSLS, by default). The queue can be defined in the DSLFNTT to be automatically started during MERVA ESA startup (parameter STATUS=AUTO).

The start queue can be shared to initiate several send processes. The messages are processed in the order of the send processes in the process table DSLKPROC. That is, the messages of the first send process are processed according to the priority of the queues defined in parameter ALLSNDQ. Then the messages of the second send process are processed, and so on, until the messages of the last send process have been processed.

Except for the shareable MERVA ESA start queues, the name of the start queue must be unique among the names of the MERVA ESA queues of all send processes and receive processes.

You can specify the following options:

NOHOLD
This indicates that in each send process only those MERVA ESA queues are processed that are in status NOHOLD. Queues with another status are skipped. This is the default.

ALL
This indicates that in each send process all affected MERVA ESA queues are processed. The queues can have any possible status except ACTIVATED. Queues in status HOLD are also processed.

The start queue name and the value for the option must be separated by a comma and enclosed in parentheses.

If this parameter is omitted, it is assumed that the current send process is initiated only for specific MERVA ESA send queues specified in parameter ALLSNDQ. This occurs when a message is put into a send queue, when a send queue is started by an operator using command SF, or when a send queue is started automatically during MERVA ESA startup.

NEXT
How to handle an MQI datagram or request message in the next processing step.

The datagram or request message can have been received from an application or correlated with a reply or report message. A datagram can have been correlated with a report message. A request message can have been correlated with a reply or report message.

STANDARD
A received or correlated datagram can be sent as a datagram. A received request message will be transformed to, and sent as, a reply message. A correlated request message can be sent as a request message. This is the default.

NEW
The MQI message type can be changed. The new message type depends on the specification of an acknowledgment wait queue in the ACKWQ parameter. If the ACKWQ parameter is available, the new message type is a request message, otherwise it is a datagram. Thus a datagram can be sent as a

datagram or as a request message. A request message can be sent as a datagram or as a request message. In contrast to the STANDARD option, a received request message is not transformed to a reply message.

FORWARD The MQI message type is preserved. A datagram can be forwarded as a datagram. A request message can be forwarded as a request message. Forwarding a message keeps its attributes contained in the MQI message descriptor MQMD with the following exceptions:

- For a received message, after the message is converted, the encoding is set to the native machine encoding.
- For a received message, after the message is converted, the coded character set identifier (CCSID) is set to the CCSID specified in the CCSIDP or CCSID parameters.
- If a user exit is specified in the EXIT parameter, the format name is set depending on the value of the MQFMT parameter and on additional data provided by the user exit.

OPMSDM

The number of operator messages issued for the current send process that are to be added to the MERVA ESA display message table.

NONE	No operator messages are added to the display message table. This is the default.
SUBSET	A subset of operator messages is added to the display message table. These messages show when a MERVA ESA send queue was started for processing or ended processing.
FULL	The full set of operator messages is added to the display message table.

Error messages are always added to the display message table, independent of the level chosen for the operator messages.

Messages contained in the display message table are displayed using the MERVA ESA command **dm** (display messages).

The size of the display message table can be adjusted by the parameter DM of the MERVA ESA customization parameter table macro DSLPARM.

OPMSJRN

The number of operator messages issued for the current send process that are to be written to the MERVA ESA journal.

NONE	No operator messages are written to the journal. This is the default.
SUBSET	A subset of operator messages is written to the journal. These messages show when a MERVA ESA send queue was started for processing or ended processing.
FULL	The full set of operator messages is written to the journal.

Error messages are always written to the journal, independent of the level chosen for the operator messages.

PASCMID

Whether an MQI datagram or request message can demand that an MQI reply and report message passes the MQI message descriptor fields CorrelId and MsgId of the appropriate datagram or request message.

YES A reply and report message must pass the fields CorrelId and MsgId of the datagram and request message. This is the default.

NO A reply and report message contains a CorrelId and MsgId in its message descriptor according to the following rules:

- The CorrelId is copied from the MsgId of the datagram or request message.
- The MsgId is newly created by the MQSeries queue manager.

Note that you must choose the same specification in the MQI-to-MERVA receive process where the local reply-to queue which receives the reply or report message is named.

REPLYTQ

The name of the local MQI reply-to queue. The queue must be defined in MQSeries and specified in an MQI-to-MERVA receive process. Its name can consist of up to 48 characters.

The reply-to queue can be shared by send processes, and receives either a reply message or a report message:

- Reply containing an acknowledgment
- Confirm-on-arrival (COA) report
- Confirm-on-delivery (COD) report
- Exception report

These reply and report messages are requested using the parameters ACKWQ, COAWQ, CODWQ, and EXCEPT, respectively. Therefore, parameter REPLYTQ must be specified together with those parameters which request a reply or a report.

Note: After correlation with the waiting message, the reply and report messages are deleted from the reply-to queue.

If this parameter is omitted, it is assumed that reply and report messages are not requested.

SECURE

For CICS/VSE only, specifies whether messages are to be encrypted and an authentication checksum calculated for them before they are sent.

NO Messages are not encrypted, and no checksum is calculated. This is the default.

AUTHENCR Messages are encrypted and a checksum is calculated using proprietary algorithms. The messages must be sent over an MQSeries channel connecting two MQSeries queue managers; that is, the receiving MQSeries cannot be the client of the sending MQSeries for VSE/ESA.

Note that you can also request the FMT/ESA authentication of SWIFT input messages by specifying AUTHENT=YES.

TRACMFS

Whether a MERVA ESA MFS debugging trace is requested for the current send process.

NO MFS debugging trace is not requested. This is the default.

YES MFS debugging trace is requested. The module DSLMMFS is not reentrant when running the debugging trace. Relink-edit DSLMMFS into a separate load library using the linkage editor parameter REUS. For VSE, the link-edit parameter RMODE=24 is also required. For details refer to the *MERVA for ESA Diagnosis Guide*.

TRACTOF
Whether a MERVA ESA TOF debugging trace is requested for the current send process.

NO TOF debugging trace is not requested. This is the default.

YES TOF debugging trace is requested. The module DSLTOFSV is not reentrant when running the debugging trace. Relink-edit DSLTOFSV into a separate load library using the linkage editor parameter REUS. For VSE, the link-edit parameter RMODE=24 is also required. For details refer to the *MERVA for ESA Diagnosis Guide*.

Defining the MQI-to-MERVA Receive Process

The DSLKPROC TYPE=RECEIVE statement describes the information used for the message transfer from the MQSeries to MERVA ESA.

Name	Operator	Operands
	DSLKPROC	**TYPE=RECEIVE**
		,MQIRCVQ=([*mqirq1*][,...][,*mqirq10*])
		,MRVCTLQ=*cccccccc*
		,NAME=*cccccccc*
		[**,ALTUID=***cccccccccccc*]
		[**,AUTHENT=**{<u>**YES**</u>|**NO**}]
		[**,CCSIDP=***nnnnn*]
		[**,CHECK=**{<u>**YES**</u>|**NO**}]
		[**,COMMIT=**{<u>**10**</u>|*nnnnn*}]
		[**,CONVERT=**{<u>**YES**</u>|**NO**}]
		[**,EXIT=***nnnnn*]
		[**,FORMAT=**{([*cccccccc*][,{<u>**W**</u>|**P**|**U**|*c*}])|**QUEUE**}]
		[**,GETWAIT=**{<u>**1000**</u>|*nnnnnn*}]
		[**,ISNCTLQ=***cccccccc*]
		[**,JIDRCVD=**{<u>**61**</u>|*cc*}]
		[**,JIDSENT=**{<u>**60**</u>|*cc*}]
		[**,JRNDGRM=**{<u>**NO**</u>|**YES**}]
		[**,JRNRCOA=**{<u>**NO**</u>|**YES**}]
		[**,JRNRCOD=**{<u>**NO**</u>|**YES**}]
		[**,JRNREXC=**{<u>**NO**</u>|**YES**}]
		[**,JRNRPLY=**{<u>**NO**</u>|**YES**}]
		[**,JRNRQST=**{<u>**NO**</u>|**YES**}]
		[**,MQIERRQ=**{*|*mqieq*}]
		[**,MRVSTAQ=***cccccccc*]
		[**,OPMSDM=**{<u>**NONE**</u>|**SUBSET**|**FULL**}]
		[**,OPMSJRN=**{<u>**NONE**</u>|**SUBSET**|**FULL**}]
		[**,OSNCTLQ=***cccccccc*]

Name	Operator	Operands
		[,PASCMID={YES|NO}] [,SECURE={NO|AUTHENCR}] [,TRACMFS={NO|YES}] [,TRACTOF={NO|YES}]

Programming Notes:

TYPE=RECEIVE
The definition of an MQI-to-MERVA receive process begins and an MQI-to-MERVA receive process entry must be generated.

You can define up to 1000 receive processes.

MQIRCVQ
The list of MQI receive queues.

You are recommended to specify the queue names according to the priority of the respective receive queue. Queues with higher priority are specified before queues with lower priority. When a receive process is initiated by the MERVA ESA start queue specified in parameter MRVSTAQ, the queues are processed in the order of their appearance in the list. The list of receive queues is enclosed in parentheses.

Up to 10 queue names *mqirq1* to *mqirq10* can be specified. The queue names consist of up to 48 characters. The total length of the queue names is limited to 255 characters.

The receive queue names must be unique within a receive process and across all receive processes.

At least one receive queue name must be specified.

MRVCTLQ
The name of the MERVA ESA control queue. The queue must be defined with KEY1 and KEY2 in the MERVA ESA function table DSLFNTT. It consists of up to 8 characters.

The name of the receive process specified in parameter NAME followed by an 8-digit number is used as KEY1. The number is in the range 00000001 to 00000010 and represents the appropriate MQI receive queue specified in parameter MQIRCVQ. The MQI MsgId is used as KEY2 to provide for a unique message key. The subparameter NOMOD is required for KEY1 and KEY2.

The MERVA ESA control queue can be shared by receive processes, except when a single MQI reply message requests one or more COA, COD, or (for MVS/ESA only) exception reports. In this case, two receive processes are required. One of them receives the MQI report and reply messages associated to the previously sent MQI request message; the other receives the MQI report messages associated with the MQI reply message that was sent on behalf of the MQI request message. The MERVA ESA control queues cannot be shared by these receive processes.

Except for the shareable MERVA ESA control queues, the name of the control queue must be unique among the names of the MERVA ESA queues of all send processes and receive processes.

NAME
The name of the current receive process. It consists of up to 8 characters.

The name of the current receive process must be unique for all receive processes.

ALTUID
For MVS/ESA only, specifies the alternate user identifier that is to be used to validate the opening of the MQI receive queues of the current receive process.

The alternate user identifier consists of up to 12 characters. The first 8 characters are used to check the authorization for the open, in place of the user identifier that the receive process is currently running under. The current user identifier may not even be authorized to do the open. Context checks, however, are still carried out with the current user identifier.

The current user identifier must also be authorized to specify this particular alternate user identifier. All 12 characters of the alternate user identifier are used for this check. If this parameter is omitted, it is assumed that an alternate user identifier is not needed.

AUTHENT
Whether the FMT/ESA authenticates SWIFT output messages.

YES SWIFT output messages are authenticated. The SWIFT authentication must have been started either automatically or by the command **start**, which starts the authentication initialization program defined in the MERVA ESA nucleus program table DSLNPTT. This is the default.

For CICS/VSE, note that you can also request general message authentication by setting the parameter SECURE=AUTHENCR.

NO SWIFT output messages are not authenticated.

CCSIDP
The coded character set identifier (CCSID) that identifies the code page to which the data contained in received messages should be converted if CONVERT=YES is specified. For CONVERT=NO, the CCSID of each received message is used. The value can be up to 5 digits.

The value of CCSIDP overrides the CCSID specified in DSLKPROC TYPE=INITIAL, but if CCSIDP is omitted, that CCSID is used.

CHECK
The MERVA ESA MFS message checking can be enabled or disabled before a message is mapped from an external network format to the TOF.

YES The MFS message checking is enabled. This is the default.

NO The MFS message checking is disabled. This option reduces the overhead in the mapping process.

COMMIT
The frequency of commits for changes on the MQI queues. Committing indicates that the changes on the MQI queues can be made permanent.

The value must be between 1 and 32767. A commit frequency of 1 means that a commit is made after each message contained in a MERVA ESA send queue is processed. The default is 10.

CONVERT
Whether received messages are to be converted. The messages can contain both character and binary data.

YES Messages are converted. This is the default.

NO Messages are not converted.

When the MERVA-MQI Attachment runs under IMS or CICS/ESA and message conversion is requested, an MQI data-conversion exit can perform the conversion. This requires that the name of the data-conversion exit is equal to the format name contained in the message descriptor MQMD of the received message. If the format name is MQSTR indicating that the message consists of characters only, MQSeries converts the message without using a data-conversion exit. A format name consisting of 8 blanks inhibits the usage of a data-conversion exit. The sample exits DSLKCDCC and DSLKCDCM are data-conversion exits.

When the MERVA-MQI Attachment runs under CICS/VSE and message conversion is requested, the sample MERVA-MQI Attachment exit DSLKCVSE performs the conversion.

EXIT

The number of an MFS user exit that can be used for the following purposes:

- An MQI reply message was received and correlated with the message waiting for the reply. The reply data is to be written to one or more fields in the correlated message.
- An MQI datagram or request message was received together with additional data. This additional data is to be written to one or more fields in the received message.

The exit number must be specified in the MERVA ESA MFS program table DSLMPTT. The DSLMPTT must be link-edited to the module DSLMMFS.

The value must be between 1 and 32767 (inclusive), and cannot be a number that is reserved for MFS exits of other MERVA ESA components. If this parameter is omitted, it is assumed that a user exit is not needed.

FORMAT

The MERVA ESA format control information for the messages in the current receive process.

The messages can be formatted in MERVA ESA external line format or in queue format:

- When **external line format** is requested, the first subparameter is the message identifier of a MERVA ESA MCB. It consists of up to 8 characters.The message identifier identifies the MCB via the message type table DSLMTTT.

  In order to process a wide range of different messages, you can specify the message identifier of an appropriate MERVA ESA cover MCB. Examples for cover MCB message identifiers are:

Message Identifier	Cover MCB	Description
KCOV	DSLKCOV	Used for the MERVA-MQI Attachment
0COV	DSL0COV	Sample for SWIFT and telex messages
MCOV	EKAMCOV	Used for the MERVA Link
TCOV	ENLTCOV	Used for the Telex Link

If this subparameter is omitted, a blank message identifier is assumed, and the MERVA ESA message type determination exit is used to determine the message type.

The second subparameter specifies the format code that is to be used to format the message in the MCB identified by the first subparameter:

W	SWIFT message (the default)
P	Telex message
U	User-defined message
c	User-defined format code

This format code must be the same as that specified for the ID parameter of the DSLLDEV macro.

- When **queue format** is requested, the value **QUEUE** must be specified.

GETWAIT
The wait interval for getting a message from an MQI receive queue. The wait interval is the maximum time, in milliseconds, that the current receive process waits for a message to arrive. If no message has arrived by this time, the current receive process considers the MQI receive queue empty.

The wait interval can be from 0 and 999999. The default is 1000 milliseconds.

The wait interval applies only when the current receive process was triggered for a specific MQI receive queue specified in parameter MQIRCVQ. The wait interval is ignored when the current receive process was initiated by the MERVA ESA start queue specified in parameter MRVSTAQ.

ISNCTLQ
The name of the ISN control queue used by the FMT/ESA. The queue must be defined with KEY1 and KEY2 in the MERVA ESA function table DSLFNTT. The name can be up to 8 characters.

KEY1 is ignored for an MQI-to-MERVA receive process. The value of the MQI MsgId or a modified value is used as KEY2. The subparameter NOMOD is required for KEY2.

This parameter is required when the FMT/ESA user exit is to be called, that is, when the value of the EXIT parameter is 8044.

JIDRCVD
The identifier of the MERVA ESA journal record that is written for a message received from the sending FMT/ESA. Received messages are:

- SWIFT output messages
- SWIFT input message acknowledgments

The value is one or 2 hexadecimal characters. The default is 61.

JIDSENT
The identifier of the MERVA ESA journal record that is written for a message sent to the receiving FMT/ESA. Sent messages are:

- SWIFT input messages
- SWIFT output message acknowledgments

Note: In the FMT/ESA, these acknowledgments are not sent; however, they are still recorded in the MERVA ESA journal so that both these acknowledgments and acknowledgments sent by SWIFT Link are treated consistently.

The value is one or 2 hexadecimal characters. The default is 60.

JRNDGRM
Whether those messages which are received as MQI datagrams are written to the MERVA ESA journal.

NO	Datagrams are not written to the journal. This is the default.
YES	Datagrams are written to the journal.

JRNRCOA
Whether those messages which are received as MQI Confirm-on-arrival (COA) reports are written to the MERVA ESA journal.

NO	COA reports are not written to the journal. This is the default.
YES	COA reports are written to the journal.

JRNRCOD
Whether those messages which are received as MQI Confirm-on-delivery (COD) reports are written to the MERVA ESA journal.

NO	COD reports are not written to the journal. This is the default.
YES	COD reports are written to the journal.

JRNREXC
For MVS/ESA only, specifies whether those messages which are received as MQI exception reports are written to the MERVA ESA journal.

NO	Exception reports are not written to the journal. This is the default.
YES	Exception reports are written to the journal.

JRNRPLY
Whether those messages which are received as MQI replies are written to the MERVA ESA journal.

NO	Replies are not written to the journal. This is the default.
YES	Replies are written to the journal.

JRNRQST
Whether those messages which are received as MQI requests are written to the MERVA ESA journal.

NO	Requests are not written to the journal. This is the default.
YES	Requests are written to the journal.

MQIERRQ
Indicates or specifies the name of an MQI error queue. The error queue contains received messages that could not be processed. The queue name consists of up to 48 characters.

The asterisk (*) indicates that the MQSeries dead-letter queue is to be used as error queue. This specification is only possible when the MERVA-MQI Attachment runs under IMS or CICS/ESA.

The name of the MQI error queue can be shared by receive processes. It must be unique among the names of the MQI queues of all send and receive processes.

If this parameter is omitted, it is assumed that an MQI error queue is not requested.

MRVSTAQ

The name of the MERVA ESA start queue. The name can be up to 8 characters.

The start queue can be used to initiate one or more of the MQI-to-MERVA receive processes. A receive process can be initiated by entering the MERVA ESA operator command SF (Start Function) followed by the name of the start queue, or automatically during MERVA ESA startup. The queue must be defined in the MERVA ESA function table DSLFNTT as a dummy queue (parameter QUEUE=DUMMY) used in the MERVA-MQI Attachment (parameter MQI=YES) with the transaction code for the receive process (parameter TRAN=DSLR, by default). The queue can be defined in the DSLFNTT to be automatically started during MERVA ESA startup (parameter STATUS=AUTO).

The start queue can be shared to initiate several receive processes. The messages are processed in the order of the receive processes in the process table DSLKPROC. That is, the messages of the first receive process are processed according to the priority of the queues defined in parameter MQIRCVQ. Then the messages of the second receive process are processed, and so on, until the messages of the last receive process have been processed.

Except for the shareable MERVA ESA start queues, the name of the start queue must be unique among the names of the MERVA ESA queues of all send processes and receive processes.

If this parameter is omitted, it is assumed that the current receive process is triggered only for specific receive queues specified in parameter MQIRCVQ. This occurs when a message is put into a receive queue.

OPMSDM

The number of operator messages issued for the current receive process that are to be added to the MERVA ESA display message table.

NONE	No operator messages are added to the display message table. This is the default.
SUBSET	A subset of operator messages is added to the display message table. These messages show when an MQI receive queue was started for processing or ended processing.
FULL	The full set of operator messages is added to the display message table.

Error messages are always added to the display message table, independent of the level chosen for the operator messages.

Messages contained in the display message table are displayed using the MERVA ESA command **dm** (display messages).

The size of the display message table can be adjusted by the parameter DM of the MERVA ESA customization parameter table macro DSLPARM.

OPMSJRN

The number of operator messages issued for the current receive process that are to be written to the MERVA ESA journal.

NONE	No operator messages are written to the journal. This is the default.
SUBSET	A subset of operator messages is written to the journal. These messages show when an MQI receive queue was started for processing or ended processing.
FULL	The full set of operator messages is written to the journal.

Error messages are always written to the journal, independent of the level chosen for the operator messages.

OSNCTLQ

The name of the OSN control queue used by the FMT/ESA. The queue must be defined in the MERVA ESA function table DSLFNTT. The name can be up to 8 characters.

This parameter is required when the FMT/ESA user exit is to be called, that is, when the value of the EXIT parameter is 8044.

PASCMID

Whether a received MQI reply and report message passes the MQI message descriptor fields CorrelId and MsgId of the appropriate datagram or request message.

YES A reply and report message passes the fields CorrelId and MsgId of the datagram or request message. This is the default.

NO A reply and report message contains a CorrelId and MsgId in its message descriptor according to the following rules:

- The CorrelId has been copied from the MsgId of the datagram or request message.
- The MsgId has been newly created by the MQSeries queue manager.

Note that you must choose the same specification as in the associated MERVA-to-MQI send process where the datagram or request message was sent.

SECURE

For CICS/VSE only, specifies whether the messages are decrypted and authenticated after retrieval.

NO	Messages are neither decrypted nor authenticated. This is the default.
AUTHENCR	Messages are decrypted and authenticated using proprietary algorithms. After a message has been decrypted, its authenticity is checked by calculating a checksum and comparing it with the checksum that was sent with the message. The messages must be received from an MQSeries channel connecting two MQSeries queue managers; that is, the sending MQSeries cannot be the client of the receiving MQSeries for VSE/ESA. Note that you can also request the FMT/ESA authentication of SWIFT output messages by specifying AUTHENT=YES.

TRACMFS
Whether a MERVA ESA MFS debugging trace is requested for the current receive process.

NO MFS debugging trace is not requested. This is the default.

YES MFS debugging trace is requested. The module DSLMMFS is not reentrant when running the debugging trace. Relink-edit DSLMMFS into a separate load library using the linkage editor parameter REUS. For VSE, the link-edit parameter RMODE=24 is also required. For details refer to the *MERVA for ESA Diagnosis Guide*.

TRACTOF
Whether a MERVA ESA TOF debugging trace is requested for the current receive process.

NO TOF debugging trace is not requested. This is the default.

YES TOF debugging trace is requested. The module DSLTOFSV is not reentrant when running the debugging trace. Relink-edit DSLTOFSV into a separate load library using the linkage editor parameter REUS. For VSE, the link-edit parameter RMODE=24 is also required. For details refer to the *MERVA for ESA Diagnosis Guide*.

Ending the MERVA-MQI Attachment Process Table

The DSLKPROC TYPE=FINAL statement defines the end of the MERVA-MQI Attachment process table. It is a mandatory statement and must be followed by an assembler END statement.

Name	Operator	Operands
	`DSLKPROC`	`TYPE=FINAL`

Programming Notes:

TYPE
TYPE=FINAL must be specified for this form of the macro. All other parameters are ignored.

Generating a DSECT of the MERVA-MQI Attachment Process Table

The DSLKPROC TYPE=DSECT statement generates a DSECT of the MERVA-MQI Attachment process table in assembler language. This DSECT describes all parts of the process table, the table header, the send process entry, the receive process entry, and the table trailer. It also provides equate symbols for flags and identifiers.

Name	Operator	Operands
[*label*]	`DSLKPROC`	`TYPE=DSECT`

Programming Notes:

label
A unique statement label according to assembler language conventions. The default is DSLKPROC.

TYPE
TYPE=DSECT must be specified for this form of the macro. All other parameters are ignored.

DSLLCOND: Branching within an MCB

The DSLLCOND macro is used to specify that a conditional test is to be performed. The contents of the TOF field or buffer specified as the first operand are compared to the contents of the TOF field or literal specified as the third operand; the comparison is made as indicated by the operator specified as the second operand: equal or not equal. If the condition is satisfied, the branch specified in the GOTO operand is taken, or, if PAGE=NEW was specified, the next field is placed on a new page.

Name	Operator	Operands

	DSLLCOND	`[{`**`O1=`**`{(`*`dataref`*`)}`**`,EQ=`**`{`**`NO`**` }`**`,O2=`**`{'`*`literal`*`'}}]`
		`     {(`*`buffer`*`) }    {`**`YES`**`}    {(`*`dataref`*`)}`
		` {`**`HARDCOPY`**`                          }`
		` {`**`INPUT`**`                             }`
		` {`**`LINES=`**`*n*` }`
		` {`**`OUTPUT`**`                            }`
		` {`**`SCREEN`**`                            }`
		` {`**`SYSP`**`                              }`
		
		`[`**`,COMMENT=`**`{`**`NO`**`|`**`Y`**`}]`
		
		`[{`**`,GOTO=`**`*name*`}]`
		` {`**`,PAGE=NEW`**` }`

Programming Notes: If no operands are specified, the DSLLCOND macro has the effect of the Assembler ANOP statement and no entry is generated in the MCB.

O1
The first operand. O1 is to be compared to O2 as indicated by the relational operator EQ.

EQ
The relational operator.

YES O1 must be equal to O2 to fulfill the condition.

NO O1 must be unequal to O2 to fulfill the condition.

O2
The second operand. O2 is to be compared to O1 as indicated by the relational operator EQ.

buffer
The reference to a buffer as defined below. This parameter is only allowed for DSLLDEV TYPE=NET.

```
BUFFER,LENGTH=n,OFFSET=n
```

where:

BUFFER
The contents of the buffer being processed are to be used for testing.

LENGTH
The length of the area to be tested. Specify a value from 1 to 200. If no length is specified, the actual length of the second operand O2 is taken for the comparison.

OFFSET
The offset between the current position in the buffer and the beginning of the area to be tested. Specify a value from –32767 to 32767. The

default is 0, the first character at the current position in the buffer. When the specified data area (offset+length) exceeds the buffer, the condition is not met.

dataref

The reference to a TOF field in the form:

```
TEST=fn[{,DANUM={1|n|LAST}}][,GROUP={1|n|name}][,LENGTH=n][,NI=n][,OFFSET=n][,RSNUM=n]
        {,OPTION=YES      }
```

where:

TEST The TOF field name that is to be tested.

DANUM The number of the data area that is to be used for the test. Specify either a value from 1 to 32767, or LAST to show that the last-filled data area is to be used for the test. The default is 1.

OPTION The field option is to be used for the test.

GROUP Refers to the name or number of a DSLLGRP defined in DSLLDEV TYPE=MESSAGE. See also the description of the GROUP operand in the DSLLGRP macro. The default is 1.

LENGTH The length of the area to be tested. Specify a value from 1 to 200. The default is the length of the field data area minus the offset, if specified.

NI The nesting identifier. It can have a value of from 0 to 255, indicating the part of the nested message that is to be used for the test. If no specification is made, the current nesting identifier is used for processing.

OFFSET The offset between the beginning of the field data area and the beginning of the area to be tested. Specify a value from 0 to 200. The default is 0, the first character of the field data area. Offset plus length must be less than or equal to 200.

RSNUM The occurrence of a field in a repeatable sequence (defined by the REPSEQ operand of the DSLLUNIT macro) that is to be used for the test. Specify a value from 1 to 32767. The default is the current occurrence in a repeatable sequence.

When the field belongs to a nested repeatable sequence, a list of occurrence indices may be specified as the RSNUM parameter. The length of the list must match the nesting depth of the field within the nested repeatable sequences. Up to 10 occurrence indices may be specified. For example:

```
RSNUM=(1,2,2)
```

'literal'

A character string with which the area specified in O1 is to be compared. The length of the literal specified must be from 0 to 225 characters. An empty literal string ('') can be specified to test for empty or nonexistent TOF fields, or in the case of buffer tests, the condition is always true when the current position is still within the buffer.

HARDCOPY

If the message is being processed for a hardcopy printer, the condition is fulfilled.

INPUT
The condition is fulfilled when a message is mapped from a net buffer into the TOF. This parameter is only allowed for DSLLDEV TYPE=NET.

LINES
Action (GOTO or PAGE) is to be taken if the number of lines available on the current page is less than the value specified. Specify a value from 1 to the maximum number of lines on the screen or printed page but not exceeding 32767. LINES can only be specified if SCREEN, HARDCOPY, or SYSP is specified.

OUTPUT
The condition is fulfilled when a message is mapped from the TOF into a net buffer. This parameter is only allowed for DSLLDEV TYPE=NET.

SCREEN
If the message is being processed for a screen, the condition is fulfilled.

SYSP
If the message is being processed for a SYSOUT printer, the condition is fulfilled.

COMMENT
Y indicates that an MNOTE will be issued showing the macro parameters at the time of expansion. The default is NO.

GOTO
The label assigned to another macro. The label specified must occur later in the MCB macro definition than the DSLLCOND macro that refers to it; that is, only forward branches are allowed. If the condition is fulfilled, processing continues at the point in the MCB designated by the specified label. Up to 999 GOTO specifications may be made within an MCB. If no condition is specified, GOTO is executed unconditionally.

Note: Care should be taken when assigning GOTO labels. Only those labels are checked for uniqueness that are so indicated in the description of the MCB macros, for example, the labels of the DSLLDEV macros. The first occurrence of the label specified in the GOTO operand that is met after the GOTO specification is assumed to be the correct one.

The target label of the GOTO parameter must be defined in the same device description as the DSLLCOND macro, or it must be the label immediately after the end of the device description. When the target label is defined within a unit, then the DSLLCOND macro must be in the same or an inner unit. You cannot jump into a unit of fields.

PAGE
The next field is to be placed on a new page or screen if the condition is fulfilled. If no condition is specified, PAGE=NEW has the effect of an Assembler EJECT statement; that is, the next field is placed on a new page or screen. This operand is only allowed for DSLLDEV TYPE=SCREEN, DSLLDEV TYPE=HARDCOPY, or DSLLDEV TYPE=SYSP.

This operand is ignored in a top or bottom frame MCB, for example, in DSL0TOP or DSL0BOT.

DSLLDEV: Defining a Device

The DSLLDEV macro is the beginning of a device-oriented definition. The DSLLxFLD macros following this DSLLDEV statement are mapped to the device until the next DSLLDEV or DSLLGEN macro is encountered. The device definitions used for one message type must all be contained in one MCB (either through definition or by embedding). If a device definition for TYPE=MESSAGE is provided in the MCB, it must be the first device definition in the MCB. Up to 999 devices can be defined in one MCB. A DSLLDEV macro must immediately follow the DSLLMCB macro.

Note: The different formats of the DSLLDEV macro are shown below. The operand descriptions follow the formats.

DSLLDEV Macro for Message Description

Name	Operator	Operands
	DSLLDEV	**TYPE=MESSAGE**

DSLLDEV Macro for Screen, SYSOUT Printer, and Hardcopy Printer

Name	Operator	Operands
[*name*]	**DSLLDEV**	**TYPE=**{**SCREEN**|**SYSP**|**HARDCOPY**} [**,ID=**{**A**|**B**|**...**|**E**|**...**|**Z**|**0**|**1**|**...**|**9**}] [**,LIKE=***name*] [**,UCTRAN=**{**NO**|**YES**}]

DSLLDEV Macro for External Line Format

Name	Operator	Operands
[*name*]	**DSLLDEV**	**TYPE=NET** [**,ID=**{**A**|**B**|**...**|**S**|**...**|**Z**|**0**|**1**|**...**|**9**}] [**,LIKE=***name*] [**,RELCHAR=**{**X**'*cc*'|'*char*'}] [**,SEP=**{**X'0D25'**|**X**'*literal*'|'*literal*'}

Programming Notes:

name
Assigns a name that can be referred to by the LIKE operand of other DSLLDEV macros. The name specified is also used as the label of the device entry generated and must therefore be unique within the MCB. If no name is specified, the name generated for the individual types is shown below:

```
TYPE=MESSAGE  -- DSLMSG
TYPE=SCREEN   -- DSLSCn, where n is a number from 1 to 999
TYPE=SYSP     -- DSLSPn, where n is a number from 1 to 999
TYPE=HARDCOPY -- DSLHCn, where n is a number from 1 to 999
TYPE=NET      -- DSLNTn, where n is a number from 1 to 999
```

A DSLLDEV TYPE=MESSAGE can only be specified once in an MCB, and, if specified, it must be the first device specified. A total of 998 of the remaining device types can be specified in an MCB.

TYPE
The device type.

One of the following values must be specified:

```
MESSAGE    for message description
SCREEN     for screen layouts
SYSP       for SYSOUT printer layouts
HARDCOPY   for hardcopy printer layouts
NET        for external line formats
```

ID A unique identifier for the device type. For example, the identifier can be used to distinguish among different languages or different network lines. Only one character A through Z or 0 through 9 can be specified. The default is:

E	For TYPE=SCREEN, TYPE=SYSP, and TYPE=HARDCOPY
S	For TYPE=NET

LIKE
Causes the same fields and layout to be used for this device type as for the device type referred to. Using this parameter, it is possible to refer one device description to a previously defined device description. The name specified in the LIKE operand of a DSLLDEV TYPE=SCREEN, DSLLDEV TYPE=HARDCOPY, or DSLLDEV TYPE=SYSP macro must be the name assigned as label to any other previously defined DSLLDEV TYPE=SCREEN, DSLLDEV TYPE=HARDCOPY, or DSLLDEV TYPE=SYSP macro; for example, the LIKE operand of a DSLLDEV TYPE=HARDCOPY macro can refer to a previously defined DSLLDEV TYPE=SCREEN macro. The name specified in the LIKE operand of a DSLLDEV TYPE=NET macro must be the name assigned as a label to a previously defined DSLLDEV TYPE=NET macro. If LIKE is specified, no DSLLxFLD definitions can be specified for the device.

Note: The LIKE parameter must not refer to a DSLLDEV macro that is also defined with the LIKE parameter.

RELCHAR
A release character for a DSLLDEV TYPE=NET input stream. It is a literal character enclosed in single quotation marks (') or two hexadecimal characters enclosed in single quotation marks (') and preceded by an X. When the release character occurs in the input data stream immediately before a string which is defined as a separator, this string belongs to the data area instead of representing a separator. The application program is responsible for inserting and removing release characters at the appropriate places in the data stream. If the parameter is not specified, no release character is used to scan the input data stream.

SEP
The separator used to show the end of a data area for a DSLLDEV TYPE=NET input stream. It can be a literal character string of 0 to 8 characters enclosed in single quotation marks (') or an even number (2 to 16) of hexadecimal characters enclosed in single quotation marks (') and preceded by an X. The default is X'0D25'.

UCTRAN
Whether input data is to be translated to uppercase for the specified device. This parameter is significant only for DSLLDEV TYPE=SCREEN, and is

ignored for other types. When UCTRAN=NO (the default), uppercase translation is done only for the top and the bottom frame windows on the screen.

The UCTRAN value for the device can be overridden for a particular input field using the UCTRAN parameter of the DSLLDFLD statement for that field. Input data received from the message frame window is not translated by MERVA ESA.

DSLLDFLD: Defining a Screen or Printer Field

The DSLLDFLD macro defines a device field that is read from or written to a terminal (DSLLDEV TYPE=SCREEN) or written to a hardcopy or system printer (DSLLDEV TYPE=HARDCOPY or DSLLDEV TYPE=SYSP, respectively).

Name	Operator	Operands
	DSLLDFLD	{*dataref*|'*literal*'}
		[,**COLOR**={**DEFAULT**}]
		{**BLUE** }
		{**RED** }
		{**PINK** }
		{**GREEN** }
		{**TURQ** }
		{**YELLOW** }
		{**NEUTRAL**}
		[,**COMMENT=Y**]
		[,**CSSRC**={(*dataref*)|'*literal*'}
		[,**CSTARG**=(*dataref*)]
		[,**CURSOR**={**BEG**|**END**}]
		[,**DACNT**=({**1**|*min*},{**1**|*max*},**COMPRES**)]
		[,**DBCS**={**NO**|**YES**}]
		[,**DISP**={**NORM**|**NODISP**|**HIGH**}]
		[,**EDIT**={*n*|**AMOUNT**}]
		[,**HIL**={**REVERSE**|**BLINK**|**UL**}]
		[,**LENGTH**=*nnnnn*]
		[,**LIT**=*nnnnn*]
		[,**POS**=([{*line* } {,*column* } {,**NEWLINE** }])
		{**NEXT**[+*n*]}][{,**NEXT**[+*c*]}][{,*overflow*}]
		[,**PROT**={**NO**|**YES**}]
		[,**RETYPE**={**NO**|**YES**}]
		[,**UCTRAN**={**NO**|**YES**}]

Programming Notes:

dataref

The reference to a TOF field in the form:

```
FLD=fn[{,DANUM={n|LAST}}][,GROUP={n|name}][,NI=n][,RSNUM=n]
      {,OPTION=YES    }
```

where:

FLD

The reference to a TOF field. *fn* is the TOF field name.

DANUM
The number of the data area that is to be displayed or printed. Specify either a value from 1 to 32767, or LAST to show that the last-filled data area is to be displayed or printed. If no specification is made, the current data area number is used for processing.

Note: When DACNT is specified in this macro or in the preceding DSLLUNIT macro, the specified data area is displayed or printed as often as specified in the DACNT parameter. Therefore, the DANUM and DACNT parameters should not both be used in the same *dataref* specification.

OPTION
The option of the field is to be displayed or printed.

GROUP
Refers to the name or number of a DSLLGRP defined in DSLLDEV TYPE=MESSAGE. See also the description of the GROUP operand in the DSLLGRP macro. This specification overrides the GROUP specification in the DSLLGRP macro for this field only. The default is the specification made in the GROUP operand of the DSLLGRP macro; if no GROUP specification was made in the DSLLGRP macro, the default is 1, the first group defined for DSLLDEV TYPE=MESSAGE.

NI The nesting identifier with a value of 0 to 255, indicating the part of the nested message that is to be displayed or printed. If no specification is made, the current nesting identifier is used for processing.

RSNUM
The occurrence of a field in a repeatable sequence (defined by the REPSEQ operand of the DSLLUNIT macro) that is to be displayed or printed. Specify a value from 1 to 32767. If no specification is made, the current occurrence is used for processing.

When the field belongs to a nested repeatable sequence, a list of occurrence indices may be specified as the RSNUM parameter. The length of the list must match the nesting depth of the field within the nested repeatable sequences. Up to 10 occurrence indices may be specified.

For example:

```
RSNUM=(1,2,,3)
```

For each index, a value between 1 and 32767 can be specified. When an index is omitted, it refers to the current occurrence of the field which is on display.

'literal'
A literal character string to be presented to the device. The length of the literal specified must be from 1 to 225 characters for VSE and from 1 to 253 characters for MVS.

COLOR
The field's color, except for DEFAULT and NEUTRAL. The color displayed for NEUTRAL is device dependent. In general, NEUTRAL is white on display devices and black on printers. This operand is only significant for DSLLDEV TYPE=SCREEN and DSLLDEV TYPE=HARDCOPY and is ignored for DSLLDEV TYPE=SYSP.

The default value for literals is COLOR=&LITERAL and for TOF fields COLOR=&DATA. The assembler globals are defined in the copy book

DSLCOLOR and may be changed there. The definition in this copy book is 'BLUE' for &LITERAL and 'GREEN' for &DATA.

COMMENT
Y indicates that an MNOTE will be issued showing the macro parameters at the time of expansion.

The abbreviation C=Y can also be used.

CSSRC
Used only with DSLLDFLD TYPE=SCREEN, this parameter defines the cursor selection source field. When the cursor is placed on the field defined by the DSLLDFLD macro and a key defined for cursor selection is pressed, the source field or literal specified in CSSRC is moved to the target field specified in CSTARG.

A key can be defined for cursor selection by using the parameter CS=YES in the macro DSLMPFK of the PF-key definition table.

When CSSRC specifies a data reference, if a TOF field is defined by the DSLLDFLD macro, the default value for the CSSRC parameters DANUM, GROUP, NI, and RSNUM is the actual TOF position. If a literal is defined by the DSLLDFLD macro, the default value for the CSSRC GROUP is 1 and for DANUM, NI, and RSNUM, the actual position. The LENGTH parameter is not allowed in a CSSRC data reference.

CSTARG
Used only with DSLLDFLD TYPE=SCREEN, this parameter defines the cursor selection target field. The operand CSSRC must also be specified. When the cursor selection source data has been moved to the target field, if the command line field is not empty, the command is executed.

The default for CSTARG is the command line. If CSTARG specifies only a field name, the default values are DANUM=1, GROUP=1, NI=0, and RSNUM=1.

CURSOR
The positioning of the cursor:

BEG The cursor is to be positioned under the first character of the field.

END The cursor is to be positioned following the last character of the field.

This operand is only significant for DSLLDEV TYPE=SCREEN and is ignored for DSLLDEV TYPE=HARDCOPY and DSLLDEV TYPE=SYSP.

DACNT
The number of data areas that are to be displayed or printed. Starting with the *min* specification, one data area is displayed or printed until the *max* number specified for DACNT is attained. If no specification is made, one data area is displayed or printed. If only one specification is made, it is the maximum.

If DACNT was specified in DSLLUNIT, it must not be specified in DSLLDFLD.

min
The minimum number of times the unit can occur. Specify a value from 1 to 32767. The default is 1.

max
The maximum number of times the unit can occur. Specify a value from 1 to 32767. The default is 1.

COMPRES
Only nonempty data areas after the first data area are to be displayed or printed.

DBCS
Whether double-byte character set (DBCS) is allowed for an input field. The default is NO.

DISP
The field's display intensity as normal (NORM), high intensity (HIGH), or nondisplayable (NODISP). The default is NORM. This operand is only significant for DSLLDEV TYPE=SCREEN and DSLLDEV TYPE=HARDCOPY and is ignored for DSLLDEV TYPE=SYSP.

EDIT
The number of the routine used to edit the contents of the field before printing or displaying on the screen. Input to the field is also edited. A value from 0 to 32767 can be specified for *n*. If 0 is specified, no module is called. The default is the value specified in the FDT, if any.

The following mnemonic can be used to identify the standard routine:

```
AMOUNT    for amount editing (inserting decimals) (n=901)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided editing routines.

HIL
Extended highlighting for the field:

REVERSE	reverse video
BLINK	blink
UL	underline

This operand is only significant for DSLLDEV TYPE=SCREEN and DSLLDEV TYPE=HARDCOPY and is ignored for DSLLDEV TYPE=SYSP.

LENGTH
The display length of the field data area for fields specified with the FLD or LIT parameter. For fields specified with the FLD parameter, the *LENGTH* parameter is mandatory. Specify a number from 1 to 65535. When the actual field data area is longer than the specification of *LENGTH*, the field is truncated.

Note: LENGTH does not include the attribute character reserved for an IBM 3270 display device. The attribute character occupies the space immediately preceding the field for DSLLDEV TYPE=SCREEN and DSLLDEV TYPE=HARDCOPY; this character is not accessible to an application program. There is no attribute character for DSLLDEV TYPE=SYSP.

LIT
A reference to a literal in the literal table DSLMLITT. The reference is a number from 1 to 999999. The LENGTH parameter defines the length of the slot on the screen reserved for the literal. A possibly specified literal character string is treated as comment then. If the literal with the specified number is not found in the literal table, the first entry in the literal table is used.

POS
Defines the position in which the first data area for the field is to begin on the

page or screen. The last POS specification made remains in effect, and the position of all succeeding fields is calculated based on that specification as described below until a new POS specification is made. To avoid overlapping fields, it is recommended that relative positions be specified using the NEXT(+*n* or +*c*) parameters:

line
: A value from 1 to 32767. The default value for line is 1 for the first field and NEXT(+*n*) for the remaining fields. However, if a column specification is made or if NEXT(+*c*) is in effect, the default line value is the current line. If an overflow occurs during NEXT(+*c*) column processing, the default for line is NEXT(+*n*).

column
: A value from 1 to 255. For the first field, the default value for column is 1 for DSLLDEV TYPE=SYSP and 2 for DSLLDEV TYPE=SCREEN and DSLLDEV TYPE=HARDCOPY (to allow for the attribute character); for the remaining fields, the default value is the previous column position.

+*c* The number of blanks that are to separate the next field from the current field. The default is 0.

+*n* The number of blank lines that are to separate the next field from the current field. The default is 0.

NEXT
: The next available position is to be used for the field. If NEXT is specified for line, the previous line is increased by 1 plus the +*n* specification. If NEXT is specified for column, the previous column is increased by the length of the previous field plus the +*c* specification plus the attribute character, if any. If the field does not fit onto the current line, the line number is increased by 1 plus the +*n* specification, and either the overflow specification, if any, or the column default value plus +*c* is used for the new position.

NEWLINE
: The data areas of the current field are to be placed on the NEXT(+*n*) line that the currently defined or calculated column position is to be retained. The first data area is thus placed in the current column position of the current line with the remaining data areas aligned underneath the first. The NEWLINE specification is only in effect for the field for which it is specified.

overflow
: The column in which the field is to begin when field positioning causes an overflow onto a new line. A value from 1 to 255 can be specified.

For example, the following POS specifications for DSLLDEV TYPE=SCREEN with a line width of 80 will place the field in the following positions:

```
A1:  no specification → POS=(1,2)
A2:  no specification → POS=(2,2)
A3:  no specification → POS=(3,2)
.
.
.
B1 (LENGTH=35):  POS=(,NEXT)        → POS=(1,2)
B2 (LENGTH=35):  no specification    → POS=(1,38)
B3 (LENGTH=20):  no specification    → POS=(2,2) (overflow!)
B4 (LENGTH=10):  no specification    → POS=(2,23)
.
.
```

```
.
C1 (LENGTH=5):   POS=(NEXT+1,NEXT+7) → POS=(2,9)
C2 (LENGTH=4):   no specification    → POS=(4,22)
C3 (LENGTH=10):  POS=(,NEXT)         → POS=(4,27)
C4 (LENGTH=15):  POS=NEXT            → POS=(5,27)
C5 (LENGTH=25):  no specification    → POS=(6,27)
C6 (LENGTH=2):   POS=(NEXT,NEXT)     → POS=(7,38)
.
.
.
DSLLDFLD '2. DEST'                                → POS=(1,2)
DSLLDFLD FLD=XX,POS=(,NEXT,10),LENGTH=11,DACNT=10 → POS=(1,10)
                                                  → POS=(1,22)
                                                  → POS=(1,34)
                                                  → POS=(1,46)
                                                  → POS=(1,58)
                                                  → POS=(1,70)
                                      (overflow!) → POS=(2,10)
                                                  → POS=(2,22)
                                                  → POS=(2,34)
                                                  → POS=(2,46)
.
.
.
DSLLDFLD 'AMOUNT:'                                → POS=(1,2)
DSLLDFLD FLD=XX,POS=(,NEXT,NEWLINE),LENGTH=11,DACNT=3 → POS=(1,10)
                                                  → POS=(2,10)
                                                  → POS=(3,10)
```

PROT
Whether the field is protected against change. The default is NO. For literal fields, the field is always protected, and a specification of PROT=NO is ignored. For IBM 3270 color devices, the PROT specification also influences the basic field color. This operand is only significant for DSLLDEV TYPE=SCREEN and DSLLDEV TYPE=HARDCOPY and is ignored for DSLLDEV TYPE=SYSP.

RETYPE
Whether the field is to be typed in again when using the Retype function.

NO The field contents are displayed and need not be typed in again. This is the default.

YES Blanks are displayed rather than the field contents, and the contents must be typed in again. The retyped and original contents of the field are compared and the user is notified of discrepancies.

This operand is only significant for DSLLDEV TYPE=SCREEN.

UCTRAN
Whether input data is to be translated to uppercase for the specified input field. The default is the value specified for DSLLDEV TYPE=SCREEN for the screen to which the field belongs. Input data received from the message frame window is not translated by MERVA ESA.

DSLLEXIT: Defining Message Nesting

The DSLLEXIT macro can be used either to embed a message identifier or to exit to the next or a new nesting identifier, during processing.

Name	Operator	Operands
	`DSLLEXIT`	[*dataref*] [,**FIELDS**={<u>**MAND**</u>|**OPT**}] [,**IMBED**=*name*] [,**NEXTNI**={<u>**OPT**</u>|**MAND**}] [,**SEP**=({'*literal*' }[,**ALWAYS**])] {X'*literal*'} [,**TAG**=({'*literal*' }[,**ALWAYS**])] {X'*literal*'}

Programming Notes: A specification of DSLLEXIT without operands causes processing to return to the macro statement immediately following the original DSLLEXIT macro statement. For example, a DSLLEXIT IMBED causes processing to continue in another MCB; a DSLLEXIT macro without operands in the second MCB causes processing to return to the original MCB.

dataref

The reference to a TOF field in the form:

```
FLD=fn[,GROUP={name}][,NI=n]
              {n   }
```

where:

FLD

The TOF field name.

GROUP

Refers to the name or number of a DSLLGRP defined in DSLLDEV TYPE=MESSAGE. See also the description of the GROUP operand in the DSLLGRP macro. The default is 1 for the first group defined for DSLLDEV TYPE=MESSAGE.

NI

The nesting identifier. It can have a value of from 0 to 255, indicating the part of the nested message that is to be displayed or printed. If no specification is made, the current nesting identifier is used for processing.

NI=0 indicates that a new nesting level indicator is to be initialized when mapping a message from line to TOF.

FIELDS

Shows whether the fields contained in a nested message are to remain mandatory:

MAND

The fields contained in a nested message are to remain mandatory if they were mandatory in the message. This is the default.

OPT

The fields contained in a nested message are to be optional even if they were mandatory in the message.

IMBED
A message identification defined as an entry in the Message Type Table (MTT) or an MCB name. The named entry is not physically copied into the current MCB at the location of the DSLLEXIT IMBED; a reference to the entry is defined. During processing, the reference to the embedded material is used to get the needed information. Processing continues in the current MCB after the embedded information has been processed.

Note: The current MCB cannot be checked against the embedded MCB during assembly, for example, to determine if group numbering is correct.

NEXTNI
Shows an exit to the next nesting identifier (for example, from identifier 1 to identifier 2):

OPT
The processing of the next nesting identifier is optional and is the default. For example, processing of the next nesting identifier is optional for SWIFT message type 195.

MAND
The processing of the next nesting identifier is mandatory. For example, processing of the next nesting identifier is mandatory for SWIFT message type 192.

SEP
The separator used to show the end of the embedded data for a DSLLDEV TYPE=NET input stream. It can be a literal character string of 0 to 8 characters enclosed in single quotation marks (') or an even number (2 to 16) of hexadecimal characters enclosed in single quotation marks (') and preceded by an X. The default is the separator specified in the device definition.

ALWAYS indicates that the separator should be written to the output buffer, even if the exit is not taken, or there is no data to be mapped for these MCBs.

TAG
The character code of the embedded data. It can be a literal character string of 0 to 8 characters enclosed in single quotation marks (') or an even number (2 to 16) of hexadecimal characters enclosed in single quotation marks (') and preceded by an X. The tag is used to identify embedded input or output data and can only be specified for DSLLDEV TYPE=NET.

ALWAYS indicates that the tag is to be written to the output buffer, even if the exit is not taken, or there is no data to be mapped for these MCBs.

DSLLFDT: Starting an FDT

The DSLLFDT macro is used to generate the FDT header and must be the first macro specified for the FDT.

Name	Operator	Operands
[*name*]	**DSLLFDT**	

Programming Notes:

name
The unique name of the FDT. The default FDT name is DSLFDTT. The name of the FDT must also be specified in the FDT operand of the DSLPARM macro if it is other than DSLFDTT.

DSLLFLD: Defining a Field

The DSLLFLD macro is used to define a field and its characteristics in the FDT. The definition should reflect those attributes that are most often used for the field in all messages in which the field occurs, for example, length or editing routine. The attributes defined in the FDT are used if they are not overwritten in the message definition (DSLLMFLD macro) in the MCB. A total of 4095 fields and subfields can be defined in the FDT.

Name	Operator	Operands

name	**DSLLFLD**	[**CHECK=**{**ALPHA** }]
		{**NUMERIC** }
		{**ANUM** }
		{**HEX** }
		{*n* }
		{**0** }
		{**YYYYMMDD**}
		{**YYYYDDD** }
		{**YYMMDD** }
		{**YYDDD** }
		{**MMDD** }
		{**DD** }
		{**HHMM** }
		{**HH** }
		{**MM** }
		{**SEPR** }
		{**LABEL** }

		[**,DAMAX=**{**1**|*n*}]

		[**,DEFAULT=**{*n* }]
		{**0** }
		{**YYYYMMDD**}
		{**YYYYDDD** }
		{**YYMMDD** }
		{**YYDDD** }
		{**MMDD** }
		{**DD** }
		{**HHMM** }
		{**HH** }
		{**MM** }

		[**,EDIT=**{**0**|*n*|**AMOUNT**}]
		[**,EXPAND=**{**0**|*n*}]
		[**,FSEP=**{**NO**|**YES**}]
		[**,INIT=FIRST**]
		[**,LENGTH=**([**0**|*n*][,*n*[,{[**F**|**U**|**V**}]]])]
		[**,MAND=**{**NO**|**YES**}]
		[**,OPTION=**{**YES**|**NO**}]
		[**,OPTLIST=**(*option*,*option*,...)]
		[**,PAD=**{'*c*'|*X'xx'*}]
		[**,PERM=**{**NO**|**YES**}]
		[**,QUEUE=**{**YES**|**NO** }]
		[**,SEPR=**{**0**|*n*|**STANDARD**|**SYSTEM**}]
		[**,STRIP=**{**NO**|**YES**}]

Programming Notes: The numbers specified for CHECK, EDIT, DEFAULT, and EXPAND are used for the invocation of the module as specified in the MFS program table DSLMPTT.

name
Uniquely identifies the entry in the FDT and names the field.

CHECK
A number from 0 to 32767 indicating the routine used to check the contents of the field. If 0 is specified, no module is called. The default is 0.

The following mnemonics can be used to identify standard routines:

```
ALPHA     alphabetic character set     (n=901)
NUMERIC   character set 0-9            (n=902)
ANUM      alphanumeric character set   (n=903)
HEX       hexadecimal character set    (n=911)
YYYYMMDD  for dates as year month day  (n=914)
YYYYDDD   for dates as year day        (n=915)
YYMMDD    for dates as year month day  (n=904)
YYDDD     for dates as year day        (n=905)
MMDD      for dates as month day       (n=906)
DD        for dates as day             (n=907)
HHMM      for times as hours minutes   (n=908)
HH        for times as hours           (n=909)
MM        for times as minutes         (n=910)
SEPR      Checking is carried out by
          the separation routine
          specified in the SEPR operand.
LABEL     Assembler type label         (n=912)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided checking routines.

DAMAX
The maximum number of data areas for the field; for example, if LENGTH=35 and DAMAX=4 are specified, the field can consist of up to 4 data areas of 35 characters each. A value from 1 to 32767 can be specified. The default is 1.

DEFAULT
A number from 0 to 32767 used to set default values in the field. This routine is called during initialization or if the field is read and is empty. If 0 is specified, no module is called. The default is 0.

The following mnemonics can be used to identify standard routines:

```
YYYYMMDD  for dates as year month day  (n=914)
YYYYDDD   for dates as year day        (n=915)
YYMMDD    for dates as year month day  (n=904)
YYDDD     for dates as year day        (n=905)
MMDD      for dates as month day       (n=906)
DD        for dates as day             (n=907)
HHMM      for times as hours minutes   (n=908)
HH        for times as hours           (n=909)
MM        for times as minutes         (n=910)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided default routines.

EDIT
A number from 0 to 32767 used to edit the contents of the field before printing, displaying, sending, or receiving. Input to the field is also edited. If 0 is specified, no module is called. The default is 0.

The following mnemonic can be used to identify the standard routine:

```
AMOUNT    for amount editing (inserting decimals) (n=901)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided editing routines.

EXPAND
A number from 0 to 32767 used to expand the contents of the field. Expansion is carried out in the TOF and can be, for example, the expansion of a SWIFT address into a correspondent's name. If 0 is specified, no module is called. The default is 0.

Note: The numbers 900 to 32767 are reserved for IBM-provided expansion routines.

FSEP
Whether the separation routine is called when a data area of the field is read, written, or deleted. If FSEP=YES, the module number of the separation routine is specified with the parameter SEPR, and the interface to the routine is identical to the interface used for subfield separation. The default is FSEP=NO.

INIT
The field is to be initialized for nesting identifier 1 only. For example, the trailer of a SWIFT message can occur only once and should then be for nesting identifier 1. INIT=FIRST indicates that the field is initialized only on the first nesting identifier that is 1.

LENGTH
The length of a field data area as a minimum length and a maximum length. The values for *n* can be in the range from 0 to 65535. The default is 0. If only one value is specified, it is the maximum length of the field; the minimum length is assumed to be 0.

F	A fixed length field and is the default if no type is specified for the field length.
U	A field of unlimited length; if U is specified, no value specification is allowed for *n*.
V	A variable length field, and the values specified for *n* must not be equal.

For fixed length fields, two different values can also be specified; the field can then be either of the two lengths. If more than two fixed lengths are possible for a field, define the field as variable and check it using a checking routine.

Note: These values are only used for the validation of the field data. When writing data to a TOF field, they are ignored.

MAND
Whether the occurrence of the field is mandatory. The default is NO. MAND=YES causes a mandatory indicator (question mark '?') to be set on the screen, if no input is entered for a subfield marked as mandatory.

OPTION
Whether the options defined in the OPTLIST operand are to be used for the field. YES shows the field has an option field and that an option is to be processed for the field. NO shows the field has no option field. If OPTLIST is specified, the default for OPTION is YES; otherwise, it is NO.

OPTLIST
A list of possible options for the field. Use the OPTION operand to turn option

processing off and on. The options specified must all have the same number of characters (that is, the same length). The total number of characters specified must not exceed 253 for MVS and 225 for VSE.

PAD
A padding character, either a character enclosed in single quotation marks (for example, '0') or two hexadecimal characters enclosed in single quotation marks and preceded by an X (for example, X'0F') . When the field is read from the TOF, the field is padded with the specified character up to the minimum length of the field. If the data length of the field exceeds the minimum length specified in the LENGTH parameter, no padding is performed.

PERM
The disposition of the field. YES shows a permanent field that is not reinitialized during message initialization. NO shows a field that is always initialized during message initialization. The default is NO.

QUEUE
Whether the field is to be stored in a MERVA ESA queue. The default is YES.

SEPR
The number of the routine that separates the field into subfields. The routine is called when a subfield is referred to. A value from 0 to 32767 can be specified. If 0 (the default) is specified, no module is called. If CHECK=SEPR is specified, this operand is required.

The following mnemonics can be used to identify MERVA ESA provided separation routines:

```
STANDARD  standard separation     (n=901)
SYSTEM    system field separation (n=902)
```

Note: The numbers 900 to 32767 are reserved for separation routines provided by IBM.

STRIP
Whether blanks at the end of a data area should be truncated. This truncation is performed only for fields entered on a screen terminal. When no data is left after truncation, that is, only blanks were entered, the data area is deleted. The default is NO.

DSLLGEN: Ending an FDT or an MCB

The DSLLGEN macro:

- Ends the FDT and completes the FDT generation. It must be the last macro specified for the FDT.
- Ends the message definition and completes the MCB generation. It must be the last macro specified for the MCB.

The Assembler END statement must immediately follow the DSLLGEN macro.

Name	Operator	Operands
	`DSLLGEN`	

DSLLGRP: Defining a Field Group

The DSLLGRP macro defines a group of fields. If a field occurs more than once in a DSLLDEV TYPE=MESSAGE definition, use the DSLLGRP macro to distinguish between them; the same field name must not occur twice within a group in a DSLLDEV TYPE=MESSAGE definition.

Groups are numbered internally and can therefore be referred to either by name or by number. Up to 255 groups can be defined for one device type. A group ends when a new DSLLGRP, a DSLLDEV, or a DSLLGEN macro is defined.

If an operand (GROUP or GRPNUM) is specified and no NI specification is made in the DSLLDFLD or DSLLNFLD macros in the group, all data references within the group refer to the current nesting identifier; if no operand is specified and no NI specification is made in the DSLLxFLD macros in the group, all data references within the group refer to nesting identifier 0 (internal fields in the TOF).

Name	Operator	Operands
[*name*]	**DSLLGRP**	[**GROUP=**{*name*|*n*}] [**,GRPNUM=***n*]

Programming Notes:

name
: Assigns a name to the group; within a DSLLDEV TYPE=MESSAGE definition, the name must be unique.

GROUP
: Refers to a DSLLGRP macro named in DSLLDEV TYPE=MESSAGE. If a field occurs more than once in a message, the GROUP operand sets up a unique relationship between the fields defined for DSLLDEV TYPE=MESSAGE and those defined for other devices. This operand must not be specified for DSLLDEV TYPE=MESSAGE.

  name
  : Refers to the name of a DSLLGRP defined in the device description for the message (DSLLDEV TYPE=MESSAGE).

  n The number assigned to the group internally. The number can be in the range from 1 to 255. When embedding (see the description of the DSLLEXIT macro), a number can be used to relate to a particular group, providing independence from names. The numbers are not checked for prior definition in DSLLDEV TYPE=MESSAGE.

GRPNUM
: Assigns a number to the group. This operand can only be specified for DSLLDEV TYPE=MESSAGE. The number specified must be in the range from 1 to 255. If specified, groups must be numbered in ascending order, and the number must be unique within the device. If this operand is not specified, the groups within a device are numbered by MERVA ESA in ascending order beginning with the last GRPNUM specified or 1, for the first group in the device, and incrementing by 1.

DSLLMCB: Starting an MCB

The DSLLMCB macro shows the start of the MCB and names it. DSLLMCB must be the first macro specified in the MCB.

Name	Operator	Operands
name	**DSLLMCB**	

Programming Notes:

name
The unique name of the MCB in the library (member name). The same name must also be specified in the message type table entry for a particular message type. During processing, the name in the message type table entry is used to find the MCB.

DSLLMFLD: Defining a Message Field

In an MCB you use the DSLLMFLD macro to declare a field in a message (DSLLDEV TYPE=MESSAGE). You can use the operands to overwrite or extend the specifications of the field defined in the FDT using the DSLLFLD macro (see “DSLLFLD: Defining a Field” on page 86). Up to 4095 DSLLMFLD macros can be specified within a group.

Name	Operator	Operands

name	**DSLLMFLD**	[**CHECK=**{**ALPHA**}]
		 {**NUMERIC**}
		 {**ANUM**}
		 {**HEX**}
		 {*n*}
		 {**YYYYMMDD**}
		 {**YYYYDDD**}
		 {**YYMMDD**}
		 {**YYDDD**}
		 {**MMDD**}
		 {**DD**}
		 {**HHMM**}
		 {**HH**}
		 {**MM**}
		
		 [,**DAMAX=***n*]
		
		 [,**DEFAULT=**{*n*}]
		 {**YYYYMMDD**}
		 {**YYYYDDD**}
		 {**YYMMDD**}
		 {**YYDDD**}
		 {**MMDD**}
		 {**DD**}
		 {**HHMM**}
		 {**HH**}
		 {**MM**}
		
		 [,**EDIT=**{*n*|**AMOUNT**}]
		
		 [,**EXPAND=***n*]
		
		 [,**FSEP=**{**NO**|**YES**}]
		
		 [,**LENGTH=**([**0**|*n*][,*n*[,{[**F**|**U**|**V**}]]])]
		
		 [,**MAND=**{**YES**|**NO**}]
		
		 [,**OPTION=**{**YES**|**NO**}]
		
		 [,**OPTLIST=**(*option*,*option*,...)]
		
		 [,**QUEUE=**{**YES**|**NO**}]

Programming Notes: The numbers specified for CHECK, EDIT, and DEFAULT are used for the invocation of the module as specified in the MFS program table DSLMPTT.

name
The name defined as the label of the DSLLFLD macro for the FDT entry.

CHECK
The number of the routine used to check the contents of the field. A value

from 0 to 32767 can be specified for *n*. If 0 is specified, no module is called. The default is the value specified in the FDT, if any.

The following mnemonics can be used to identify standard routines:

```
ALPHA     alphabetic character set     (n=901)
NUMERIC   character set 0-9            (n=902)
ANUM      alphanumeric character set   (n=903)
HEX       hexadecimal character set    (n=911)
YYYYMMDD  for dates as year month day  (n=914)
YYYYDDD   for dates as year day        (n=915)
YYMMDD    for dates as year month day  (n=904)
YYDDD     for dates as year day        (n=905)
MMDD      for dates as month day       (n=906)
DD        for dates as day             (n=907)
HHMM      for times as hours minutes   (n=908)
HH        for times as hours           (n=909)
MM        for times as minutes         (n=910)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided checking routines.

DAMAX
The maximum number of data areas for the field; for example, if LENGTH=35 and DAMAX=4 are specified, the field can consist of up to four data areas of 35 characters each. A value from 1 to 32767 can be specified. The default is the value specified in the FDT.

DEFAULT
The number of the routine used to set default values in the field. This routine is called during initialization or if the field is read and is empty. A value from 0 to 32767 can be specified. If 0 is specified, no module is called. The default is the value specified in the FDT, if any.

The following mnemonics can be used to identify standard routines:

```
YYYYMMDD  for dates as year month day  (n=914)
YYYYDDD   for dates as year day        (n=915)
YYMMDD    for dates as year month day  (n=904)
YYDDD     for dates as year day        (n=905)
MMDD      for dates as month day       (n=906)
DD        for dates as day             (n=907)
HHMM      for times as hours minutes   (n=908)
HH        for times as hours           (n=909)
MM        for times as minutes         (n=910)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided default routines.

EDIT
The number of the routine used to edit the contents of the field before printing, displaying, sending, or receiving. Input to the field is also edited. A value from 0 to 32767 can be specified for *n*. If 0 is specified, no module is called. The default is the value specified in the FDT, if any.

The following mnemonic can be used to identify the standard routine:

```
AMOUNT    for amount editing (inserting decimals) (n=901)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided editing routines.

EXPAND
The number of the routine used to expand the contents of the field. Expansion is carried out in the TOF and can be, for example, the expansion of a SWIFT

address into a correspondent's name. A value from 0 to 32767 can be specified. If 0 is specified, no module is called. The default is the value specified in the FDT, if any.

Note: The numbers 900 to 32767 are reserved for IBM-provided expansion routines.

FSEP
Whether the separation routine is called when a data area of the field is read, written, or deleted. If FSEP=YES, the module number of the separation routine is specified with the parameter SEPR, and the interface to the routine is identical to the interface used for subfield separation. The default is FSEP=NO.

LENGTH
The minimum and a maximum lengths of a field data area. The values can be from 0 to 65535. If only one value is specified, it is assumed to be the maximum length of the field, and the minimum length is assumed to be 0.

F	A fixed length field and is the default if a value is specified but no type is specified.
U	A field of unlimited length; if U is specified, no value specification is allowed for *n*.
V	A variable length field, and the values specified for *n* must not be equal.

For fixed length fields, two different values can also be specified; the field can then be either of the two lengths. If more than two fixed lengths are possible for a field, define the field as variable and check it using a checking routine. The default is the value specified in the FDT, if any.

Note: These values are only used for the validation of the field data. When writing data to a TOF field, they are ignored.

MAND
Whether the occurrence of the field is mandatory. The default is the value specified in the FDT. MAND=YES causes a mandatory indicator (question mark '?') to be set on the screen if no input is entered for a field marked as mandatory.

OPTION
Whether the options defined in the OPTLIST operand are to be used for the field. YES shows the field has an option field. This parameter must be specified if an option is to be processed for the field. NO shows the field has no option field. The default is the value specified in the FDT.

OPTLIST
A list of possible options for the field. Use the OPTION operand to turn option processing off and on. The options specified must all be of the same number of characters (that is, the same length). The total number of characters specified must not exceed 253 for MVS and 225 for VSE.

QUEUE
Whether the field is to be stored in a MERVA ESA queue. The default is the value specified in the FDT.

DSLLNFLD: Defining a Network Field

The DSLLNFLD macro defines a field that is read from or transmitted to a communications line (DSLLDEV TYPE=NET).

Name	Operator	Operands
	DSLLNFLD	[*dataref*]
		[**,EDIT=***n*]
		[**,FSEP=**({'*literal*' }[**,ALWAYS**])] {X'*literal*'}
		[**,LENGTH=**{*n*|(*n*,**NOPAD**)}]
		[**,OPTLIST=**(*option*,*option*,...)]
		[**,SEP=**({'*literal*' }[,{**ALWAYS**}])] {X'*literal*'} {**EOD** }

		[**,TAG=**({'*literal*' }[**,ALWAYS**])]
		{X'*literal*' }
		{'*literal*'-...-'*literal*' }
		{X'*literal*'-...-X'*literal*'}

		[**,VFIRST=**{**NO**|**YES**}]

Programming Notes:

dataref
: The reference to a TOF field in the form:

```
FLD=fn[{,DANUM={n|LAST}}][,GROUP={n|name}][,NI=n][,RSNUM=n]
       {,OPTION=YES    }
```

where:

FLD
: The TOF field name.

DANUM
: The number of the data area that is to be written to or read from the buffer. Specify either a value from 1 to 32767, or LAST to show that the last-filled data area is to be written to or read from the buffer. If no specification is made, all data area numbers are used for processing. If the TOF field is a field defined with SEPR=SYSTEM in DSLFDTT, DANUM=1 must be specified.

OPTION
: The option of the field is to be written to or read from the buffer.

GROUP
: Refers to the name or number of a DSLLGRP defined in DSLLDEV TYPE=MESSAGE. See also the description of the GROUP operand in the DSLLGRP macro. This specification overrides the GROUP specification in the DSLLGRP macro for this field only. The default is the specification made in the GROUP operand of the DSLLGRP macro; if no GROUP specification was made in the DSLLGRP macro, the default is 1, the first group defined for DSLLDEV TYPE=MESSAGE.

NI
: The nesting identifier. It can have a a value of from 0 to 255, indicating the

part of the nested message that is to be written to or read from the buffer. If no specification is made, the current nesting identifier is used for processing.

RSNUM

The occurrence of a field in a repeatable sequence (defined by the REPSEQ operand of the DSLLUNIT macro) that is to be written to or read from the buffer. Specify a value from 1 to 32767. If no specification is made, the current occurrence is used for processing.

When the field belongs to a nested repeatable sequence, a list of occurrence indices may be specified as the RSNUM parameter. The length of the list must match the nesting depth of the field within the nested repeatable sequences. For example:

```
RSNUM=(1,2,2)
```

Specify a value between 0 and 32767, where 0 refers to the current occurrence of the field in process.

EDIT

The number of the routine used to edit the contents of the field before it is sent or received. A value from 0 to 32767 can be specified. If 0 is specified, no module is called. The default is the value specified in the FDT, if any.

Note: The numbers 900 to 32767 are reserved for IBM-provided editing routines.

FSEP

The separator used to show the end of a field for a DSLLDEV TYPE=NET input stream. This must be specified when the field separator used is different from the data separator. When the field separator is detected in the input stream, the field is assumed to be complete. For output mapping, a field separator is written at the end of the field. The data separator is written only when the field and data separators are equal.

ALWAYS indicates that the separator should be written to the output buffer, even if the field is empty.

This parameter is optional. When FSEP is not specified, no field separator is used.

LENGTH

The length of the field to be mapped to the line. This can be a number from 1 to 32767. If the specified length exceeds the actual length of the field in the TOF, the field is padded with blanks to the length specified in this operand unless the subparameter NOPAD is specified. If the specified length is less than the actual length of the field in the TOF, the field is truncated to the length specified in this operand. If LENGTH is not specified, the actual length of the field is taken.

OPTLIST

A list of possible options for the field in the network buffer. The options specified must all be of the same number of characters (that is, the same length), and that length must be the same as the number of hyphens (-), if any, specified in the TAG operand, if a TAG specification is made. The total number of characters specified must not exceed 253 for MVS and 225 for VSE.

If no option list is specified, all characters are accepted for a length defined by the number of hyphens specified in the TAG operand.

SEP

The separator used to show the end of a data area for a DSLLDEV TYPE=NET input stream. It can be a literal character string of 0 to 8 characters enclosed in single quotation marks (') or an even number (2 to 16) of hexadecimal characters enclosed in single quotation marks and preceded by an X. The default is the separator specified in DSLLDEV TYPE=NET unless only a TAG is being defined in which case no SEP is assumed.

ALWAYS indicates that the separator should be written to the output buffer, even if the data area is empty.

EOD indicates that, when reading from the input buffer, the separator is ignored. The data in the input buffer up to the end of the data is written as one data area of the specified TOF field.

TAG

The character code of the field tag. The field tag identifies an input or output field in the network buffer.

The specification can be a literal character or hexadecimal string, followed by hyphens (-) to show option characters, followed by another literal character or hexadecimal string.

Hyphens represent the position and number of variable option characters in the field tag. Valid options may be defined in the OPTLIST operand. If OPTLIST is specified, all valid options have the same length, and the number of hyphens in the TAG definition must be equal to the number of characters in each option.

If the TAG definition contains hyphens, the literal character strings before and after the hyphens may contain from 0 to 8 characters enclosed in single quotation marks ('), or an even number (2 to 16) of hexadecimal characters enclosed in single quotation marks and preceded by an X. Do not mix character and hexadecimal representation within one TAG definition.

If the TAG definition does not contain hyphens, the literal character string may contain from 0 to 225 characters in VSE or from 0 to 253 characters in MVS enclosed in single quotation marks, or an even number (2 to 16) of hexadecimal characters enclosed in single quotation marks and preceded by an X.

For example, TAG=':32'-':' means that the field tag is 5 characters long: the first 3 and the last characters are always ':32' and ':', respectively; the fourth character is the option (for example, A or B). The valid option characters are defined in the OPTLIST operand.

ALWAYS indicates that the tag is to be written to the output buffer, even if the field is empty.

VFIRST

The first field whose name appears in the TOF should be used for output mapping. For input mapping, the parameter is ignored and the current nesting and field group identifier is taken. This parameter can be used when the specific field group index of a field, which should be extracted with a general MCB, is unknown, or when it can differ between message types. The default is NO.

DSLLSUBF: Defining a Subfield

The DSLLSUBF macro defines a subfield. For easy reference, DSLLSUBF macros should immediately follow the DSLLFLD macro to which they belong. A field can have different layouts as defined by the DSLLSUBF macros belonging to it. A subfield cannot be further subdivided into subfields. A total of 4095 fields and subfields can be defined in the FDT.

Name	Operator	Operands

name	**DSLLSUBF**	[**CHECK=**{**ALPHA** }]
		{**NUMERIC** }
		{**ANUM** }
		{**HEX** }
		{*n* }
		{**0** }
		{**YYYYMMDD**}
		{**YYYYDDD** }
		{**YYMMDD** }
		{**YYDDD** }
		{**MMDD** }
		{**DD** }
		{**HHMM** }
		{**HH** }
		{**MM** }
		{**SEPR** }
		{**LABEL** }
		
		[,**DEFAULT=**{*n* }]
		{**0** }
		{**YYYYMMDD**}
		{**YYYYDDD** }
		{**YYMMDD** }
		{**YYDDD** }
		{**MMDD** }
		{**DD** }
		{**HHMM** }
		{**HH** }
		{**MM** }
		
		[,**EDIT=**{**0**|*n*|**AMOUNT**}]
		
		[,**FIELD=***name*]
		
		[,**LENGTH=**([**0**|*n*][,*n*[,{[**F**|**U**|**V**}]]])]
		
		[,**MAND=**{**NO**|**YES**}]
		
		[,**OFFSET=**{**0**|*n*}]
		
		[,**PAD=**{'*c*'|*X'xx'*}]
		
		[,**SEPR=**{**0**|*n*|**STANDARD**|**SYSTEM**}]
		
		[,**STRIP=**{**NO**|**YES**}]

Programming Notes: The numbers specified for CHECK, EDIT, DEFAULT, and SEPR are used for the invocation of the module as specified in the MFS program table DSLMPTT.

name
Uniquely identifies the entry in the FDT and names the subfield.

CHECK

The number of the routine used to check the contents of the subfield. A value from 0 to 32767 can be specified. If 0 is specified, no module is called. The default is 0.

The following mnemonics can be used to identify standard routines:

```
ALPHA     alphabetic character set     (n=901)
NUMERIC   character set 0-9            (n=902)
ANUM      alphanumeric character set   (n=903)
HEX       hexadecimal character set    (n=911)
YYYYMMDD  for dates as year month day  (n=914)
YYYYDDD   for dates as year day        (n=915)
YYMMDD    for dates as year month day  (n=904)
YYDDD     for dates as year day        (n=905)
MMDD      for dates as month day       (n=906)
DD        for dates as day             (n=907)
HHMM      for times as hours minutes   (n=908)
HH        for times as hours           (n=909)
MM        for times as minutes         (n=910)
SEPR      Checking is carried out by
          the separation routine
          specified in the SEPR operand.
LABEL     Assembler type label         (n=912)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided checking routines.

DEFAULT

The number of the routine used to set default values in the subfield. This routine is called during initialization or if an empty field is read. A value from 0 to 32767 can be specified. If 0 is specified, no module is called. The default is 0.

The following mnemonics can be used to identify standard routines:

```
YYYYMMDD  for dates as year month day  (n=914)
YYYYDDD   for dates as year day        (n=915)
YYMMDD    for dates as year month day  (n=904)
YYDDD     for dates as year day        (n=905)
MMDD      for dates as month day       (n=906)
DD        for dates as day             (n=907)
HHMM      for times as hours minutes   (n=908)
HH        for times as hours           (n=909)
MM        for times as minutes         (n=910)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided default routines.

EDIT

The number of the routine used to edit the contents of the subfield before printing, displaying, sending, or receiving. Input to the subfield is also edited. A value from 0 to 32767 can be specified. If 0 is specified, no module is called. The default is 0.

The following mnemonic can be used to identify the standard routine:

```
AMOUNT    for amount editing (inserting decimals) (n=901)
```

Note: The numbers 900 to 32767 are reserved for IBM-provided editing routines.

FIELD
The name of the field containing the subfield being defined. The default is the name specified as the label for the last DSLLFLD macro preceding the DSLLSUBF macro.

LENGTH
The length of the subfield as a minimum length and a maximum length. The values for *n* can be in the range from 0 to 65535. The default is 0. If only one value is specified, it is the maximum length of the subfield; the minimum length is assumed to be 0.

F A fixed length subfield and is the default if no type is specified for the subfield length.

U A subfield of unlimited length; if U is specified, no value specification is allowed for *n*.

V A variable length subfield, and the values specified for *n* must not be equal.

For fixed length subfields, two different values can also be specified; the subfield can then be either of the two lengths. If more than two fixed lengths are possible for a subfield, define the subfield as variable and check it using a checking routine. Specify length values for subfields only if the layout for the field is fixed in all possible cases; otherwise, use a separation routine to subdivide the field into subfields. LENGTH must be specified if MERVA ESA separation routines are used.

The maximum length of the subfield plus the offset value must not exceed the maximum length of the field to which the subfield belongs. However, if no length or a maximum length of 0 was specified for the field to which the subfield belongs, no check is carried out.

Note: These values are only used for the validation of the field data. When writing data to a TOF field, they are ignored.

MAND
Whether the occurrence of the subfield is mandatory. The default is NO. This information can be used by a separation routine.

OFFSET
The number of characters between the beginning of the field and the beginning of the subfield. A value from 0 to 65535 can be specified; the default is 0, that is, the subfield is the first subfield in the field. The offset value must not exceed the maximum length of the field to which the subfield belongs. However, if no length or a maximum length of 0 was specified for the field to which the subfield belongs, no check is carried out.

PAD
A padding character, either a character enclosed in single quotation marks (for example, '0') or two hexadecimal characters enclosed in single quotation marks and preceded by an X (for example, X'0F'). When the subfield is read from the TOF, it is padded with the specified character up to the minimum length of the subfield. If the data length of the subfield exceeds the minimum length specified in the LENGTH parameter, no padding is performed.

SEPR
The number of the routine that separates the field into subfields. The routine is

called when a subfield is referred to. A value from 0 to 32767 can be specified. If 0 (the default) is specified, no module is called. If CHECK=SEPR is specified, this operand is required.

The following mnemonics can be used to identify MERVA ESA provided separation routines:

```
STANDARD  standard separation     (n=901)
SYSTEM    system field separation (n=902)
```

Note: The numbers 900 to 32767 are reserved for separation routines provided by IBM.

STRIP
Whether blanks at the end of a data area should be truncated. This truncation is performed only for fields entered on a screen terminal. When no data is left after truncation, that is, only blanks have been entered, the data area is deleted. The default is NO.

DSLLUEND: Ending a Display Unit

The DSLLUEND macro defines the end of a unit.

Name	Operator	Operands
	`DSLLUEND`	`[COMMENT=Y]`

Programming Notes:

COMMENT
Y indicates that an MNOTE will be issued showing the macro parameters at the time of expansion.

Note: When a label is specified on the DSLLUEND macro, this label refers to the next statement in the MCB following the DSLLUEND statement.

DSLLUNIT: Defining a Display Unit

The DSLLUNIT and DSLLUEND macros enclose a group of fields that are to be displayed as a unit on the screen or printed output. DSLLUNIT macros are used to show field compression. Up to 9999 units can be defined for each MCB. A unit can be nested within another unit. Up to a maximum of 10 unit levels can be nested for screen device descriptions. A unit ends when a DSLLUEND, DSLLDEV, or DSLLGEN macro is defined.

Name	Operator	Operands
	DSLLUNIT	[**,COMMENT=Y**] [**COMPRES=**{**YES**|**NO**}] [**,DACNT=**({**1**|*min*},{**1**|*max*})] [**,REPSEQ=**({**0**|*min*},{**1**|*max*})] [**,SEQTYPE=**{**FIX**|**VAR**}]

Programming Notes:

COMMENT
Y indicates that an MNOTE will be issued showing the macro parameters at the time of expansion.

COMPRES
Whether unit compression is available for message printing and displaying. The default is YES. When YES is specified (or used as the default value), fields within a unit are displayed only when at least one nonempty data area exists within the unit; literals that have no related data are suppressed.

Note: To display or print protected fields from nesting ID 0 (NI=0), COMPRES=NO must be specified. These fields are treated like literals and would be suppressed with COMPRES=YES.

DACNT
The number of data areas that are to be displayed or printed for the fields defined by the DSLLDFLD macros in this unit. Values from 1 to 32767 can be specified for the maximum and minimum number of times the unit can occur. The default is 1. If only one specification is made, it is the maximum.

Starting with the *min* specification, one data area is displayed or printed for each field in the order that the fields are specified within the unit, and this cycle is repeated until the *max* number specified for DACNT is attained. If no specification is made, one data area is displayed. The DACNT operand can only be specified within a DSLLDEV TYPE=SCREEN, DSLLDEV TYPE=HARDCOPY, or DSLLDEV TYPE=SYSP definition. The DACNT, SEQTYPE, and REPSEQ operands are mutually exclusive. If DACNT is specified for the unit, it must not be specified in the DSLLDFLD macros in the unit. For nested units, DACNT must be specified on the lower unit level.

REPSEQ
The fields defined by the DSLLxFLD macros in this unit are a repeatable sequence. A value from 1 to 32767 can be specified for the maximum number of times the unit can occur; a value from 0 to 32767 can be specified for the minimum number of times the unit can occur. The default for *min* is 0 and for *max* is 1. If only one specification is made, it is the maximum. The DACNT, SEQTYPE, and REPSEQ operands are mutually exclusive. For nested units,

REPSEQ must be specified on a higher level than DACNT or SEQTYPE. Only one REPSEQ specification can be made within a group (DSLLGRP). A specific occurrence can be accessed using the RSNUM operand (see the DSLLxFLD and DSLLCOND macros).

SEQTYPE
The sequence in which the fields are stored in the TOF. The SEQTYPE operand is only allowed in a DSLLDEV TYPE=NET definition.

FIX
The sequence in which the fields occur is the same as that defined in the DSLLDEV TYPE=MESSAGE. The default is FIX.

VAR
The sequence in which the fields occur is not necessarily the same as the sequence defined in the DSLLDEV TYPE=MESSAGE, and the fields should be processed in the sequence defined in the DSLLDEV TYPE=NET.

DSLMFS: Defining the Message Format Services

The Message Format Service macro DSLMFS serves to:

- Initialize or terminate the Message Format Service.
- Call DSLMMFS for mapping services or exit programs. The Message Format Service maps data from the TOF to a device buffer, or from a device buffer to the TOF. Check, edit, expand, separate, and default routines may be invoked specifically or as mapping exits.
- Map the MFS parameter list and storage areas.
- Generate common assembler code for MFS programs and user exits.
- Carry out MFS basic functions.

MFS Initialization and Termination

Each program that loads DSLMMFS must initialize the service before issuing other MFS requests. Before ending, the program should terminate the Message Format Service to release acquired storage and modules loaded by MFS.

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=INIT** **,MEDIUM=MFS** **,PS=**{*addr*|(*r*)} **,TOF=**{*addr*|(*r*)} **,TS=**{*addr*|(*r*)} [**,ENVIR=**{*addr*|(*r*)}] [**,MF=(E,**{*addr*|(*r*)})] [**,OPT=**{**CONT**|*addr*|((*r*))}]

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=TERM** [**,MF=(E,**{*addr*|(*r*)})]

Programming Notes: Before using this macro, you must ensure that DSLCOM is addressable and that the field COMMFSA contains the entry address of the program DSLMMFS.

label
A unique statement label according to assembler language conventions.

TYPE
The type of MFS request:

INIT Initializes the MFS environment for the calling program.

TERM Releases acquired storage, loaded exit modules, and MCBs.

MEDIUM
The device environment of the request. MFS specifies that basic MFS functions are requested.

PS
The address of the MFS permanent storage acquired by the calling program. This storage area provides continuity between MFS requests.

The MFS permanent storage size is obtained from DSLPRM in the field NPMFSPS. The area is mapped using the macro DSLMFS TYPE=MAP, MF=PS. The first halfword must contain the storage length.

The INIT request initializes permanent storage with binary zeros unless OPT=CONT is also specified.

TOF
The address of TOF storage acquired by the calling program. The TOF size is obtained from DSLPRM in the field NPTOFSZ.

TS
The address of the MFS temporary storage pool acquired by the calling program. Storage is allocated from this pool during each MFS request. For nested requests the allocated storage is chained together.

The MFS temporary storage size is obtained from DSLPRM in the field NPMFSTS and mapped using the macro DSLMFS TYPE=MAP, MF=TS. The first halfword must contain the storage length.

ENVIR
Four characters that define the caller of MFS. This information is kept in MFS permanent storage throughout the current session. It may be accessed by MFS exits to identify the caller. This identification need not be set at MFS initialization, but once it is set in permanent storage it must not be changed by exits that operate at lower calling levels.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list.

If the MF parameter is omitted, MF=(E,(1)) is assumed. In an internal MFS exit MF=(E,MFSTLIST) is assumed.

OPT
Changes the type of MFS request. The option may be a value, or the label of a 2-byte field that contains the option code, or a general register containing the code in the 2 low-order bytes.

Option codes are defined in the MFS parameter list. You can specify the following value:

CONT
Shows the continuation of a previously initialized MFS session. The MFS permanent storage areas will not be cleared. This option is used to continue pseudo-conversational dialogs.

MFS TOF Initialization

The DSLMFS macro may be used to initialize the TOF storage area according to the message definition in an MCB.

Options are available for clearing permanent fields, setting field default values, and selecting the TOF nesting identifier.

Name	Operator	Operands

[*label*]	**DSLMFS**	**TYPE=INIT**
		,MEDIUM={**MESSAGE**|**MSG**}
		,MSGID={*addr*|(*r*)}

Name	Operator	Operands
		[,**FLD**={*addr*|(*r*)}] [,**MF=(E**,{*addr*|(*r*)})] [,**OPT**={(*opt*,...,*opt*)|*addr*|((*r*))}] [,**PREFIX**={**MFS**|*ccc*}] [,**PS**={*addr*|(*r*)}] [,**TOF**={*addr*|(*r*)}] [,**TS**={*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that DSLCOM is addressable and that the field COMMFSA contains the entry address of the Message Format Service interface DSLMMFS.

label
A unique statement label according to assembler language conventions.

TYPE
The type of MFS request.

INIT initializes the TOF for a message. The message is initialized on TOF nesting identifier 1 unless OPT=NXTID is specified. Existing fields on TOF nesting identifier 0 that are defined with PERM=NO are deleted and fields with PERM=YES are kept, unless OPT=CLRPERM is specified.

MEDIUM
The device environment of the request.

MESSAGE
Mapping is between an MCB message definition and a TOF.

MSG This abbreviation has the same meaning as MESSAGE.

MSGID
An 8-character field containing the message type to be initialized, or a general register containing the address of that field.

The message type must match an entry in the MFS message type table; the MTYPE parameter of the DSLMTT macro contains the same message identification, and the MCB parameter identifies the message control block to be used to initialize the TOF.

For values shorter than 8 characters, the contents of the MSGID field must be left justified and padded with blanks.

FLD
Shows a TOF field in an existing nesting identifier, which contains the pointers to the nesting identifier now to be initialized.

The FLD parameter specifies the label of a MERVA ESA field reference, or a general register containing the address of the field reference. The field reference storage area may be defined using the macro DSLMFS MF=FLDREF and must contain at least a valid TOF field name.

The FLD parameter is necessary when OPT=NXTID is specified. If the parameter is omitted, the last previous field reference address is used.

When nesting identifier 1 is initialized, the FLD parameter is ignored; the field DSLEXIT is always established in field group 1 of nesting identifier 0.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is (E,(1)).

OPT
Change of the basic TOF initialization function.

The option may be a value, or the label of a 2-byte field that contains the option code, or a general register containing the code in the 2 low-order bytes.

If more than one option is needed, a list of values may be specified, or the option codes may be added together.

Option codes are defined in the MFS parameter list. The following values can be used for this MFS request:

CLRPERM
All fields on TOF nesting identifier 0 that are defined with PERM=YES are now to be deleted. This option is used only when nesting identifier 1 is to be initialized.

NXTID
Establish the next nesting identifier in the TOF and initialize it for a message. All fields in the nested identifier are considered optional, even if they are defined with MAND=YES in the MCB, unless OPT=RETRY is also specified.

RETRY
Retains the mandatory status of fields in the nested level when OPT=NXTID is specified.

CONT
Keeps the old TOF and establishes a new nesting identifier in the TOF, if provided for by the message identifier.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Mapping for Queue

The DSLMFS macro may be used to map a message from the TOF to a queue buffer, or from a queue buffer to the TOF.

Before a message is mapped into the TOF, the TOF is initialized again. The TOF initialization function may be changed by the OPT parameter of the DSLMFS macro.

DSLMFS

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=**{**PUT**|**GET**} **,MEDIUM=QUEUE** [**,INBUF=**{*addr*|(*r*)}] [**,MF=(E,**{*addr*|(*r*)})] [**,OPT=**{(*opt*,...,*opt*)|*addr*|((*r*))} [**,OUTBUF=**{*addr*|(*r*)}] [**,PREFIX=**{**MFS**|*ccc*}] [**,PS=**{*addr*|(*r*)}] [**,TOF=**{*addr*|(*r*)}] [**,TS=**{*addr*|(*r*)}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The type of MFS request.

PUT A message contained in the TOF is to be stored in a queue buffer. All QUEUE=YES fields are included with the message.

GET A message contained in the queue buffer is to be stored in the caller's TOF.

MEDIUM
The device environment of the request. QUEUE specifies that mapping is between a queue buffer and a TOF.

INBUF
For TYPE=GET only, specifies the address of the queue buffer containing the message to be transferred to the TOF.

The buffer must be in standard MERVA ESA format; its size can be up to the size specified in the MAXBUF parameter of the DSLPARM macro. This buffer size may exceed 32KB.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is (E,(1)).

OPT
The following values can be used for this MFS request:

CLRPERM
All fields on TOF nesting identifier 0 that are defined with PERM=YES are to be deleted. Fields defined with PERM=NO are always deleted for TYPE=GET.

DYNBUF
For TYPE=PUT only, the request may return a dynamic buffer in the output buffer field of the MFS parameter list. A dynamic buffer is allocated by the service program when the specified buffer is too small to contain the message in queue format. Refer to the *MERVA for ESA Customization Guide* for details.

OUTBUF
For TYPE=PUT only, specifies the address of the queue buffer to receive the message to be transferred from the TOF.

The buffer must be in standard MERVA ESA format; its size can be up to the size specified in the MAXBUF parameter of the DSLPARM macro. This buffer size may exceed 32KB.

If the buffer specified is too small to contain the message in the queue format and OPT=DYNBUF has been specified, the service program will allocate a dynamic buffer. The message is moved into this buffer and the address of the buffer is passed to the caller in the output buffer address field of the MFS parameter list.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Message Mapping for a Logical Data Stream

The DSLMFS macro is used to transform data from the TOF to a logical data stream (LDS), or from a screen via an LDS to a TOF.

The LDS is always associated with a data buffer. For output, the MFS edit buffer is required. For input, the screen I/O buffer is used.

The macro transforms one page of data, depending on the device specification in the TUCB.

This form of the DSLMFS macro is also used to initialize the LDS work areas.

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=**{**INIT**|**PUT**|**GET**} **,MEDIUM=LDS** [**,INBUF=**{*addr*|(*r*)}] [**,MF=(E,**{*addr*|(*r*)})] [**,OUTBUF=**{*addr*|(*r*)}] [**,PREFIX=**{<u>**MFS**</u>|*ccc*}] [**,PS=**{*addr*|(*r*)}] [**,TOF=**{*addr*|(*r*)}] [**,TS=**{*addr*|(*r*)}]

Programming Notes: Before using this macro, the programmer must ensure that DSLCOM is addressable and that the field COMTUCBA contains the address of an initialized TUCB for the target device. A TUCB storage map can be generated by using the macro DSLMFS MF=TUCB.

The LDS work areas must be initialized using TYPE=INIT before issuing any TYPE=PUT or TYPE=GET requests for the LDS medium.

label
An assembler label for the macro.

TYPE
The MFS function to be performed.

INIT Initializes internal work areas of MFS for screen and printer processing.

PUT Creates a logical data stream from the current TOF.

GET Transfers received data to the TOF. TYPE=GET is used only for input received via the macro DSLMFS MEDIUM=SCREEN, TYPE=GET.

MEDIUM=LDS
A logical data stream for screen and printer devices.

INBUF
For TYPE=GET only, specifies the label of the received data buffer, or a general register containing the address of the buffer. This is the same buffer as specified in the INBUF parameter of the macro DSLMFS MEDIUM=SCREEN, TYPE=GET.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

If INBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is (E,(1).

OUTBUF
For TYPE=PUT only, specifies the label of the edit buffer, or a general register containing the address of the buffer.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

If OUTBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Message Mapping for a Physical Data Stream

The DSLMFS macro may be used to transform information from a logical data stream (LDS) to a device-specific physical data stream for presentation on screen and printer devices. The MFS reason code shows if the end of the logical data stream has been reached.

The macro is also used for input, to update the LDS with data received from a screen.

Name	Operator	Operands

[*label*]	**DSLMFS**	**TYPE=**{**PUT**|**GET**}
		,INBUF={*addr*|(*r*)}
		,MEDIUM={**SCREEN** }
		{**HARDCOPY**}
		{**SCSP** }
		{**SYSOUT** }
		{**SYSP** }
		,OUTBUF={*addr*|(*r*)}
		[**,PREFIX=**{**MFS**|*ccc*}]
		[**,PS=**{*addr*|(*r*)}]
		[**,TOF=**{*addr*|(*r*)}]
		[**,TS=**{*addr*|(*r*)}]
		[**,MF=(E,**{*addr*|(*r*)})]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The MFS function to be performed.

PUT The content of the current edit buffer is to be converted to a physical data stream, according to the specification in the LDS.

GET For MEDIUM=SCREEN only, the LDS is to be updated with data received from an IBM 3270 terminal.

INBUF
For TYPE=PUT, specifies the label of the MFS edit buffer, or a general register containing the address of the buffer. The MFS edit buffer is the same buffer used in the OUTBUF parameter of the macro DSLMFS MEDIUM=LDS, TYPE=PUT.

For TYPE=GET, specifies the label of the device I/O buffer, or a general register containing the address of the I/O buffer.

The buffer specified by INBUF must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

MEDIUM
The device environment of the request.

SCREEN	An IBM 3270 display terminal
HARDCOPY	An IBM 3270 hardcopy printer
SCSP	An SCS hardcopy printer
SYSOUT	A line printer device
SYSP	A line printer device

OUTBUF
For TYPE=PUT only, specifies the label of a device I/O buffer, or a general register containing the address of the buffer.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

If OUTBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is (E,(1)).

MFS Command Interface

You can use the DSLMFS to interpret and execute a MERVA ESA screen command.

The MFS command interpreter is needed when processing multiple-page messages for display devices. The return code from this macro shows when the complete message has been processed.

The command input for this macro is a command parsed in a DSLNPAR parameter list. The parameter list must be defined using the macro DSLNPA MF=L. It must be filled using the macro DSLNPA MF=E.

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=COMMAND** [**,INBUF=**{*addr*|(*r*)}] [**,MEDIUM=MFS**] [**,MF=(E,**{*addr*|(*r*)})]

Name	Operator	Operands
		[,**PREFIX=**{<u>**MFS**</u>|*ccc*}] [,**PS=**{*addr*|(*r*)}] [,**TOF=**{*addr*|(*r*)}] [,**TS=**{*addr*|(*r*)}]

Programming Notes: DSLCOM must be addressable and the field COMMFSA must contain the entry address of the MFS service program DSLMMFS. The field COMTUCBA must contain the address of a valid TUCB.

label
An assembler label for the macro.

TYPE
The MFS function to be performed. COMMAND calls the MFS command interpreter.

INBUF
The DSLNPAR parameter list after parsing is complete.

INBUF is the label of the parameter list, or a general register containing the address of the parameter list.

MEDIUM
The device environment of the request. MFS specifies that basic MFS functions are requested.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Mapping for Network Buffers

The DSLMFS macro may be used to map a message from a network buffer to a TOF, or from a TOF to a network buffer.

Before a message is mapped into the TOF, the TOF is initialized again. The TOF initialization function may be changed by the OPT parameter of the DSLMFS macro.

DSLMFS

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=**{**PUT**|**GET**} **,MEDIUM=NET** [**,FORMID=**{'*c*'|*addr*|(*r*)}] [**,INBUF=**{*addr*|(*r*)}] [**,MF=(E,**{*addr*|(*r*)})] [**,MSGID=**{*addr*|(*r*)}] [**,OPT=**{(*opt*,...,*opt*)|*addr*|((*r*))}] [**,OUTBUF=**{*addr*|(*r*)}] [**,PREFIX=**{**MFS**|*ccc*}] [**,PS=**{*addr*|(*r*)}] [**,TOF=**{*addr*|(*r*)}] [**,TS=**{*addr*|(*r*)}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The MFS function to be performed.

PUT Maps data from a TOF to a network buffer.

GET Maps data from a network buffer to a TOF.

MEDIUM
The device environment of the request. NET specifies a network buffer.

FORMID
A 1-character external line format identifier. The parameter may be a character in single quotes, or the label of a 1-byte field that contains the character, or a general register containing the character.

This parameter identifies the network description in the MCB (refer to the ID parameter of the macro DSLLDEV TYPE=NET). If this parameter is not specified or the specified external line format is not found in the MCB, mapping uses the first network format of the MCB as default.

INBUF
For TYPE=GET only, specifies the label of the network buffer, or a general register containing the address of the network buffer.

The buffer contains a message to be mapped into the TOF.

The buffer must be in standard MERVA ESA format; its size can be up to the size specified in the MAXBUF parameter of the DSLPARM macro. This buffer size may exceed 32KB.

If INBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

MSGID

The label of an 8-byte message identification field, or a general register containing the address of that field.

The message identification must match an entry in the MFS message type table; the MTYPE parameter of the DSLMTT macro contains the same value, and the MCB parameter identifies the Message Control Block to be used.

The first character of the MSGID field identifies the network. The remaining characters identify the message type. If the identification is less than 8 characters, the field must be padded with blanks.

For TYPE=PUT, if MSGID is omitted or does not contain a message type, the message type is taken from the TOF.

For TYPE=GET, if MSGID is omitted or does not contain a message type, an MFS user exit is called to analyze the input buffer.

OPT

The option may be a value, or the label of a 2-byte field that contains the option code, or a general register containing the code in the 2 low-order bytes. If more than one option is needed, a list of values may be specified, or the option codes may be added together.

Option codes are defined in the MFS parameter list. You can specify the following values:

CHECK	Calls MFS message checking.
CONT	For TYPE=GET only, bypasses TOF message initialization.
CLRPERM	For TYPE=GET only, all fields in the TOF are deleted before the network buffer is mapped into the TOF.
DYNBUF	For TYPE=PUT only, the request may return a dynamic buffer in the output buffer field of the MFS parameter list. A dynamic buffer is allocated by the service program when the specified buffer is too small to contain the message in queue format. Refer to the *MERVA for ESA Customization Guide* for details.

OUTBUF

For TYPE=PUT only, specifies the label of the network buffer, or a general register containing the address of the network buffer. The buffer will receive the message from the TOF in network format.

The buffer must be in standard MERVA ESA format; its size can be up to the size specified in the MAXBUF parameter of the DSLPARM macro. This buffer size may exceed 32KB.

If the buffer specified is too small to contain the message in the requested format and OPT=DYNBUF has been specified, the service program will allocate a dynamic buffer. The message is moved into this buffer and the address of the buffer is passed to the caller in the output buffer address field of the MFS parameter list.

If OUTBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

PREFIX

The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Mapping for External Line Format

The DSLMFS macro may be used to map a message in the TOF from the external line format to the fully tokenized format, or vice versa. The original format can be completely replaced or both formats can coexist in the TOF.

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=**{**PUT**|**GET**} **,MEDIUM=ELFORM** [**,FORMID=**{'*c*'|*addr*|(*r*)}] [**,MF=(E,**{*addr*|(*r*)})] [**,MSGID=**{*addr*|(*r*)}] [**,OPT=**{(*opt*,...,*opt*)|*addr*|((*r*))}] [**,OUTBUF=**{*addr*|(*r*)}] [**,PREFIX=**{**MFS**|*ccc*}] [**,PS=**{*addr*|(*r*)}] [**,TOF=**{*addr*|(*r*)}] [**,TS=**{*addr*|(*r*)}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The MFS function to be performed.

PUT Maps a message from the internal tokenized format to the external format.

GET Maps a message from the external format to the tokenized format.

MEDIUM
The device environment of the request. ELFORM specifies that the mapping is done for the external format.

FORMID
The format identification specified is used to perform the mapping from or to the tokenized format. The format identification selects a specific network device in the MCB used for the mapping. If FORMID is not specified for the PUT request, the first network device description in the MCB is used. If FORMID is not specified for the GET request, the mapping is performed

according to the contents of the field DSLTYPE. If the field DSLTYPE is empty, the format of the message is determined by the message type determination exit which inspects the external line format string.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

MSGID
The message identification specified is used to perform the mapping from or to the tokenized format. If MSGID is not specified for the PUT request, the message identification of the tokenized format (DSLEXIT field) is used. If MSGID is not specified for the GET request, the mapping is performed according to the contents of the field DSLTYPE. If the field DSLTYPE is empty, the message identification is determined by the message type determination exit which inspects the external line format string.

OPT
The option may be a value, or the label of a 2-byte field that contains the option code, or a general register containing the code in the 2 low-order bytes. If more than one option is needed, a list of values may be specified, or the option codes may be combined.

Option codes are defined in the MFS parameter list. You can specify the following values:

CONT	The original format, either the external or the tokenized format, is not overwritten, but remains in the TOF. Using this option, both message formats can be generated within a message.
DYNBUF	This option is only used when the OUTBUF parameter is specified. The request may return a dynamic buffer in the output buffer field of the MFS parameter list. A dynamic buffer is allocated by the service program when the specified buffer is too small to contain the message in external format. Refer to the *MERVA for ESA Customization Guide* for details.

OUTBUF
The output buffer is optional and specifies the address of a buffer to receive the intermediate results of the mapping operation, the message in external line format.

The buffer must be in standard MERVA ESA format; its size can be up to the size specified in the MAXBUF parameter of the DSLPARM macro. This buffer size may exceed 32KB.

If the buffer specified is too small to contain the message in external line format and OPT=DYNBUF has been specified, the service program will allocate a dynamic buffer. The message is moved into this buffer and the address of the buffer is passed to the caller in the output buffer address field of the MFS parameter list.

If no buffer is specified, an internal buffer is allocated and discarded after the mapping operation is finished.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Mapping for NOPROMPT Buffers

The DSLMFS macro may be used to map a message from a TOF to a NOPROMPT buffer, or from a NOPROMPT buffer to a TOF.

NOPROMPT mapping is like network buffer mapping, except that it is carried out on a line-by-line basis.

The specified network description from the MCB is used to divide the message into lines. The NOPROMPT LINE separator is CrLf (X'0D25').

TYPE=PUT creates a control table entry for each of the NOPROMPT LINEs, and moves the display lines to the output buffer, replacing binary zeros with blanks. MFS message checking is performed.

TYPE=GET uses the control table as input and moves the message into the TOF. Lines are inserted, replaced, or deleted as specified in the control table entries. The TOF is reinitialized before the message is moved.

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=**{**PUT**|**GET**}
		,INBUF={*addr*|(*r*)}
		,MEDIUM=NOPR
		,OUTBUF={*addr*|(*r*)}
		[**,FORMID=**{'*c*'|*addr*|(*r*)}]
		[**,MF=(E,**{*addr*|(*r*)})]
		[**,PREFIX=**{<u>**MFS**</u>|*ccc*}]
		[**,PS=**{*addr*|(*r*)}]
		[**,TOF=**{*addr*|(*r*)}]
		[**,TS=**{*addr*|(*r*)}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The MFS function to be performed.

PUT Maps data from a TOF to a NOPROMPT buffer.

GET Maps data from a NOPROMPT buffer to a TOF.

INBUF
The label of the internal control table, or a general register containing the address of the table.

The table has the standard MERVA ESA buffer format and the table size must be less than 32768 bytes.

MEDIUM
The device environment of the request. NOPR specifies a NOPROMPT buffer.

OUTBUF
The label of the NOPROMPT buffer, or a general register containing the address of the buffer.

For TYPE=PUT, the buffer will receive the message from the TOF in NOPROMPT format.

For TYPE=GET, the buffer contains a message in NOPROMPT format to be transferred to the TOF.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

FORMID
A 1-character external line format identifier. The parameter may be a character in single quotes, or the label of a 1-byte field that contains the character, or a general register containing the character.

FORMID identifies the NOPROMPT network description in the MCB (refer to the ID parameter of the macro DSLLDEV TYPE=NET).

If FORMID is not specified or the specified external line format is not found in the MCB, FORMID='X' is assumed. If network format 'X' is not found in the MCB, the first network format of the MCB is used.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Data Manipulation and Checking Modules

The DSLMFS macro may be used to explicitly call the modules for data manipulation and checking.

DSLMFS

To check or expand a complete message, standard MFS service modules are available. All other functions require a module number to identify the required program.

The module number must be found in the MFS program table DSLMPTT.

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=**{**CHECK** }
		{**DEFAULT**}
		{**EDIT** }
		{**EXPAND** }
		{**SEPR** }
		,MEDIUM={**DATA** }
		{**FIELD** }
		{**MFS** }
		{**MESSAGE**}
		{**MSG** }
		[**,FLD=**{*addr*|(*r*)}]
		[**,FORMID=**{'*c*'|*addr*|(*r*)}]
		[**,INBUF=**{*addr*|(*r*)}]
		[**,MF=(E,**{*addr*|(*r*)})]
		[**,MODNUM=**{*nnnnn*|*addr*|(*r*)}]
		[**,MSGID=**{*addr*|(*r*)}]
		[**,OPT=**{(*opt*,...,*opt*)|*addr*|((*r*))}]
		[**,OUTBUF=**{*addr*|(*r*)}]
		[**,PREFIX=**{**MFS**|*ccc*}]
		[**,PS=**{*addr*|(*r*)}]
		[**,TOF=**{*addr*|(*r*)}]
		[**,TS=**{*addr*|(*r*)}]

Programming Notes: When you call an MFS module explicitly, the MFS parameter list must contain the address of MFS permanent and temporary storage. The MODNUM parameter is required except when MEDIUM=MESSAGE is specified. The need for any other parameters depends on the logic of the individual module to be called.

The address of the parameter list generated by this macro is passed to the called module in general register 7.

Before using this macro, the programmer must ensure that DSLCOM is addressable and that the field COMMFSA contains the entry address of DSLMMFS.

label
A unique statement label according to assembler language conventions.

TYPE
The MFS function to be performed.

CHECK	Carries out message or field checking.
DEFAULT	Sets the default value for a field or data component.
EDIT	Edits or de-edits the data components of a field.
EXPAND	Gets an expanded data equivalent for a field or carries out expansion for all fields in a message.

SEPR	Separates a data component into its subfields.

MEDIUM
The device environment of the request.

DATA	Refers to a data component of a field.
FIELD	Refers to an entire field.
MFS	Shows a general service module.
MESSAGE	For TYPE=CHECK and TYPE=EXPAND only, carry out check or expansion for all eligible fields in the message.
MSG	This abbreviation has the same meaning as MESSAGE.

FLD
For MEDIUM=DATA or MEDIUM=FIELD, specifies the TOF field for which the function is requested. The FLD parameter specifies the label of a MERVA ESA field reference or a general register containing the address of the field reference. The field reference storage area can be defined using the macro DSLMFS MF=FLDREF.

FORMID
A 1-character format identifier. The parameter can be a character in single quotes, or the label of a 1-byte field that contains the character, or a general register containing the character.

INBUF
The label of an input data buffer or a general register containing the address of the buffer.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

If INBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

MODNUM
The identification number of the requested module.

The parameter may be a decimal number from 1 to 5 digits long in the range from 1 to 32767, or the label of a halfword containing the hexadecimal equivalent, or a general register containing the hexadecimal equivalent.

The MODNUM parameter is ignored when MEDIUM=MESSAGE is specified.

MSGID
The label of an 8-byte message identification field, or a general register containing the address of that field. The first character of the MSGID field identifies the network. The remaining characters identify the message type.

OPT
Change of the basic exit module processing. The option may be a value, or the label of a 2-byte field that contains the option code, or a general register containing the code in the 2 low-order bytes.

If more than one option is needed, a list of values may be specified, or the option codes may be added together. Option codes are defined in the MFS parameter list.

You can specify the following values:

CONT	For TYPE=CHECK and MEDIUM=MESSAGE, do not reinitialize the field DSLMSG. DSLMSG is the stack for field-checking errors.
DEEDIT	For TYPE=EDIT only, convert the data from external to internal format.
DELETE	For TYPE=SEPR, delete the subfield part of the data component.
ERRMSG	Store MFS error message in TOF field DSLERR on nesting identifier 0. The called program must provide a return code in general register 15 and a reason code in the caller's MFS parameter list.
NOMSG	For TYPE=CHECK and MEDIUM=MESSAGE, do not add MFS error messages to the TOF field DSLMSG.
READ	For TYPE=DEFAULT and TYPE=SEPR, perform the default setting or separation while reading data from the TOF.
WRITE	For TYPE=DEFAULT and TYPE=SEPR, perform the default setting or separation while writing data to the TOF.

OUTBUF
The label of an output data buffer, or a general register containing the address of the buffer.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

If OUTBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Calling a User Exit

The DSLMFS macro may be used to transfer control to a user-written exit module. The module must be included in the MFS program table DSLMPTT (see macro DSLMPT).

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=USER**

Name	Operator	Operands
		,MODNUM={*nnnnn*|*addr*|(*r*)} [**,FLD=**{*addr*|(*r*)}] [**,FORMID=**{'*c*'|*addr*|(*r*)}] [**,INBUF=**{*addr*|(*r*)}] [**,MEDIUM=MFS**] [**,MF=(E,**{*addr*|(*r*)})] [**,MSGID=**{n*cccccc*|*addr*|(*r*)}] [**,OPT=**{**ERRMSG**|*addr*|((*r*))} [**,OUTBUF=**{*addr*|(*r*)}] [**,PREFIX=**{MFS|*ccc*}] [**,PS=**{*addr*|(*r*)}] [**,TOF=**{*addr*|(*r*)}] [**,TS=**{*addr*|(*r*)}]

Programming Notes: When calling a user exit, the DSLMFS macro must include the MODNUM and MF parameters. The MFS parameter list must contain the address of MFS permanent and temporary storage. The need for any other parameters depends on the logic of the individual module to be called.

DSLCOM must be addressable and the field COMMFSA must contain the entry address of DSLMMFS.

label
A unique statement label according to assembler language conventions.

TYPE
The MFS function to be performed. USER calls a user exit module.

MODNUM
The identification number of the module.

The parameter may be a decimal number from 1 to 5 digits long in the range from 1 to 32767, or the label of a halfword containing the hexadecimal equivalent, or a general register containing the hexadecimal equivalent.

The module number must be found in one of the entries of the MFS program table.

FLD
A MERVA ESA field reference. It can be the label of a field reference area or a general register containing the address of that area.

The storage description of a field reference area can be generated using the macro DSLMFS MF=FLDREF.

FORMID
A 1-character format identification. It can be a character in single quotes, or the label of a 1-byte field that contains the character, or a general register containing the character.

Use the FORMID parameter to select a device description within an MCB.

INBUF
The label of an input data buffer, or a general register containing the address of the buffer.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

If INBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

Note: Data can also be passed to the user exit module in the MFS permanent storage field MFSPUCOM. It is a 4-byte field where the caller can provide input data or the address of a data storage area.

MEDIUM
The device environment of the request. MFS specifies that basic MFS functions are requested.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

MSGID
An 8-character message identification. It may be the label of an 8-byte field containing the message identification, or a general register containing the address of the field.

MSGID is used to find an entry in the MFS message type table.

OPT
Change of the user-exit processing.

The option can be a value, or the label of a 2-byte field that contains the option code, or a general register containing the code.

Option codes are defined in the MFS parameter list. You can specify the following value:

ERRMSG
Stores MFS error message in TOF field DSLERR on nesting identifier 0. The user exit must provide a return code in general register 15 and a reason code in the caller's MFS parameter list.

OUTBUF
The label of an output data buffer, or a general register containing the address of the buffer.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

If OUTBUF=0 is specified, the buffer address in the MFS parameter list is cleared to binary zeros.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

Mapping MFS Storage Areas

The DSLMFS macro may be used to generate storage definitions of the various areas needed to execute MFS service functions and to create MFS programs and user exits.

Name	Operator	Operands
[*label*]	**DSLMFS**	**,MF=**{**L**|**PS**|**TS**|**TUCB**|**FLDREF**} [**,OPT=**{**REASON**|**EXTTS**}] [**,PREFIX=**{*ccc*}] [**,TYPE=**{**MAP**|**DSECT**}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

If the label parameter is omitted and TYPE=DSECT is specified, a label is generated depending on the format of the macro (see MF parameter below). If TYPE=MAP is specified, no default label is generated.

MF
The format of the macro.

L	Maps the MFS parameter list. The default DSECT label is MFSL.
PS	Maps the MFS permanent storage. The default DSECT label is MFSPS.
TS	Maps the MFS temporary storage. The default DSECT label is MFSTS.
TUCB	Maps a MERVA ESA terminal and user control block. The default DSECT label is MFSTUCB.
FLDREF	Maps a MERVA ESA field reference area. The default DSECT label is MFSFLDRE.

OPT
A change to the storage description generated by the macro. You can specify the following values:

REASON
For MF=L only, the parameter list expansion will include equate statements for the MFS reason codes.

EXTTS
For MF=TS only, an extended temporary storage area will be mapped.

The extended storage definition includes additional parameter lists for DSLMMFS and DSLTOFSV, and a work buffer for DSLTOFSV. These storage areas are used by MFS programs and user exits to carry out the program's own MFS and TSV calls.

Note: The temporary storage allocated to an MFS program or user exit may also be increased by resuming the MFSTS DSECT within the program and defining additional storage.

PREFIX
The first 3 characters of the field names to be generated in the MFS storage descriptions. If the specified prefix is less than 3 characters, it is padded with dollar signs ($).

If the value of MF is:

- FLDREF, the default of this parameter is FLD
- TUCB, the default of this parameter is TUC
- Anything else, he default of this parameter is MFS

TYPE
The form of storage map to be generated.

MAP	The storage definition is to begin with a double word alignment DS 0D. This is the default.
DSECT	The storage is to be mapped as a dummy section.

Generate Common Assembler Entry Code for MFS Programs and User Exits

This form of the DSLMFS macro generates the beginning or entry point code for programs that will be controlled by DSLMMFS through the MFS program table DSLMPTT.

Name	Operator	Operands

[*name*]	**DSLMFS**	`[,TYPE={MFS    }]`
		`       {CHECK  }`
		`       {DEFAULT}`
		`       {EDIT   }`
		`       {EXPAND }`
		`       {SEPR   }`
		`       {USER   }`
		`MF={START│ENTRY}`
		`[,AMODE=ccc]`
		`[,COPR={NO│yyyy}]`
		`[,MODNUM=nnn]`
		`[,OPT=(opt,...,opt)]`
		`[,RMODE=ccc]`

Programming Notes: The program must include the label MFSTTSLL which shows the total temporary storage required. At execution time, this length is used to allocate storage from the MFS temporary storage pool.

The DSLMFS MF=START macro generates a DSECT for MFS temporary storage, using the label MFSTS. In the program, the DSECT must be continued and must end with an equate for the total length of the area. It may be coded as follows:

```
MFSTS    DSECT              continue DSECT
         ...                (additional program TS)
MFSTTSLL EQU   *-MFSTS      total TS length
```

name

The external name of the program or program entry point. This name relates to an item in the MFS program table defined by the macro DSLMPT, using the NAME parameter.

If the name is omitted, a label of the form **DSLMannn** is generated, where:

a	The first letter of the TYPE parameter ('X' is used for EXPAND)
nnn	The number specified in MODNUM

TYPE

The type of MFS program to be established.

The first character may be used to formulate the program or entry name as described in the name parameter above. This character is also used for the TYPE parameter of the DSLMPT macro, when adding the program to the MFS program table.

MFS	The program is an MFS general service module.
CHECK	The program carries out message checking, field checking, or both.
DEFAULT	The program sets the field data defaults.
EDIT	The program carries out data editing or de-editing for screens and printers, or for a network buffer.
EXPAND	The program carries out data expansion, for example by using a VSAM or DL/I file.
SEPR	The program carries out field separation into the structures defined in the Field Definition Table.
USER	The program is a user exit.

MF

The format of the macro.

START
: The main entry code for an MFS program or MFS exit is to be generated. It includes DSECTs for all necessary storage and control blocks, unless OPT=NOMAPS is also specified. In addition, the return points to the MFS interface DSLMMFS are always generated.

  There can be only one DSLMFS MF=START in a program and it must precede any DSLMFS MF=ENTRY statements.

ENTRY
: The code for a secondary MFS program entry is to be generated.

  There may be any number of DSLMFS MF=ENTRY statements in a program, but they must be preceded by a DSLMFS MF=START macro.

AMODE

The value of the assembler AMODE instruction created for the control section defined with an MF=START definition. Specify *NONE* to suppress the automatic creation of an AMODE instruction. The default for MERVA ESA programs is AMODE ANY.

COPR

The year of the copyright for an IBM MFS module. The default is NO. For user programs, COPR=NO should be defined.

MODNUM
A number from 1 to 32767 that is used only when the name is omitted: the last 3 digits are taken to formulate the program or entry name as described in the name parameter above. The default is 000.

OPT
Changes the macro expansion for MF=START only. A list of option values may be specified in any order. You can specify the following values:

BASE11	General registers 10 and 11 are to be used as program base registers. If BASE11 is not specified, only general register 10 is used.
EXTTS	The extended version of the MFS temporary storage is to be provided. This generates an MFS parameter list at label MFSTLIST and a TOF supervisor parameter list at label MFSTSVL.
REASON	The equates for the MFS error reason codes are to be generated together with the MFS parameter list DSECT.
EXICAL	The MFS entry or exit code should allow the use of CICS command level instructions.
NOMAPS	The MFS and DSLCOM DSECTs are not generated.
NOPRINT	The MFS and DSLCOM DSECTs are generated but printing is suppressed.

RMODE
The value of the assembler RMODE instruction created for the control section defined with an MF=START definition. Specify *NONE* to suppress the automatic creation of an RMODE instruction. The default for MERVA ESA MFS programs is RMODE ANY.

MFS Error Messages

The DSLMFS macro may be used to set an error message in the message format service permanent storage. Optionally, the error message is also written to a field in the TOF.

The error message is recorded in the trace table if the MERVA ESA processing trace is active (refer to the TRACE parameter of macro DSLPARM).

The error message identification has the format: *ccc3nnn*, where:

ccc	Are the first 3 characters of the internal environment information. These are usually the first 3 characters of the module name.
3	A constant meaning “MFS”.
nnn	A decimal value from the MSGNUM parameter specified in this macro.

The message identification must exist in the message text table supplied by the caller.

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=ERRMSG** **,MEDIUM=MFS** **,MSGNUM=**{*nnn*|*addr*|(*r*)} [**,MF=(E,**{*addr*|(*r*)}**)**]

Name	Operator	Operands
		[,**OPT**={(*opt*,...,*opt*)|*addr*|((*r*))}] [,**PREFIX**={**MFS**|*ccc*}] [,**PS**={*addr*|(*r*)}] [,**TOF**={*addr*|(*r*)}] [,**TS**={*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that DSLCOM is addressable and that the field COMMFSA contains the entry address of DSLMMFS.

The field COMMSGTA must contain the address of the message text table, and COMOMSGA must contain the entry address of the operator message program DSLOMSG.

If COMTUCBA contains the address of a valid TUCB, the TUCB language indicator will be used.

label
An assembler label for the macro.

TYPE
The MFS function to be performed.

ERRMSG creates an error message in MFS permanent storage.

MEDIUM
The device environment of the request. MFS shows a general service module.

MSGNUM
The identification number of the message text.

The parameter may be a decimal number 3 digits long in the range from 000 to 999, or the label of a halfword containing the hexadecimal equivalent, or a general register containing the hexadecimal equivalent.

Descriptive text for MFS errors can be obtained by specifying the MFS reason code in this parameter. User-defined messages must be in the range from 500 to 899.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

OPT
The error message is also written to the indicated TOF field. The option may be a value, or the label of a 2-byte field that contains the option code, or a general register containing the code in the 2 low-order bytes.

If more than one option is needed, a list of values may be specified, or the option codes may be added together.

Option codes are defined in the MFS parameter list. You can specify the following values:

CHECK	Writes the error message in the next available data area of the TOF field DSLMSG on nesting identifier 0.

ERRMSG
Writes the error message for data area 1 of the TOF field DSLERR on nesting identifier 0, unless OPT=CHECK is also specified.

Note: The field DSLERR is deleted from the TOF when it is displayed on a MERVA ESA screen terminal.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TOF
The address of TOF storage acquired by the calling program. TOF need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

MFS Locating Message, Device, and PF-Key Definitions

The DSLMFS macro may be used to:

- Get an entry from the message type table
- Get the address of a device description in an MCB (the MCB module is loaded if it is not yet available)
- Load a PF-key table

Name	Operator	Operands
[*label*]	**DSLMFS**	**TYPE=**{**GETMTT**|**GETDEV**|**GETPFK**} **,MEDIUM=**{**MFS** } {**MESSAGE** } {**MSG** } {**SCREEN** } {**HARDCOPY**} {**SCSP** } {**SYSOUT** } {**SYSP** } {**NET** } {**NOPR** } [**,FORMID=**{'*c*'|*addr*|(*r*)}] [**,MF=(E,**{*addr*|(*r*)})] [**,MODNAME=**{*addr*|(*r*)}] [**,MSGID=**{*addr*|(*r*)}] [**,OPT=FUNC**] [**,OUTBUF=**{*addr*|(*r*)}] [**,PREFIX=**{**MFS**|*ccc*}] [**,PS=**{*addr*|(*r*)}] [**,TS=**{*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that DSLCOM is addressable and that the field COMMFSA contains the entry address of DSLMMFS. For TYPE=GETMTT and TYPE=GETDEV, the field COMMTTA must contain the address of the message type table.

label
A unique statement label according to assembler language conventions.

TYPE
The MFS function to be performed.

GETMTT
Gets an entry from the message type table. The entry is returned in the buffer specified by the parameter OUTBUF. When OPT=FUNC is specified, the descriptive text defined for the message identification is returned in the buffer.

GETDEV
Finds a device description in an MCB. The MCB module is loaded if it is not already available. The device description address is returned in MFS permanent storage field MFSPMCBD.

GETPFK
Loads a PF-key table. The load address is returned in MFS permanent storage field MFSPPFKA.

MEDIUM
The device environment of the request:

MFS	For TYPE=GETMTT and TYPE=GETPFK, a general service module (this is the default)
MESSAGE	The message portion of an MCB
MSG	The message portion of an MCB
SCREEN	The screen portion of an MCB
HARDCOPY	The hard-copy portion of an MCB
SCSP	For TYPE=GETDEV, the hard-copy portion of an MCB
SYSOUT	A line print portion of an MCB
SYSP	A line print portion of an MCB
NET	A network portion of an MCB
NOPR	A network portion of an MCB

FORMID
For TYPE=GETDEV only, a 1-character format identifier. The parameter can be a character in single quotes, or the label of a 1-byte field or general register containing the character.

If this parameter is omitted and a TUCB address is available in the DSLCOM field COMTUCBA, the TUCB specification is used. The network identifier is taken for MEDIUM=NET and MEDIUM=NOPR. For other specifications of MEDIUM, the TUCB language identifier is used. If this parameter is omitted and no TUCB is available, the first occurrence of the specified device is selected in the MCB.

MF
The format of the macro. The first parameter must be E for execute form. The

second parameter specifies the label of the MFS parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

MODNAME
Required for TYPE=GETPFK only, specifies the name of the PF-key table.

The parameter may be the label of an 8-byte field containing the module name, or a general register containing the address of that field.

The module name must exist in the MFS program table DSLMPTT (see macro DSLMPT).

MSGID
The label of an 8-byte message identification field, or a general register containing the address of that field.

The message type must exist in the message type table (see macro DSLMTT).

The MSGID parameter is required for TYPE=GETMTT and TYPE=GETDEV requests.

OPT=FUNC
For TYPE=GETMTT only, specifies that the descriptive text for a message identification is to be returned in the buffer.

OUTBUF
The label of an output data buffer, or a general register containing the address of the buffer.

The OUTBUF parameter is required for TYPE=GETMTT. The requested entry from the message type table will be returned in this buffer.

The buffer must be in standard MERVA ESA format and the buffer size must be less than 32768 bytes.

PREFIX
The first 3 characters of the field names in the MFS parameter list. The default is MFS.

PS
The address of MFS permanent storage acquired by the caller. PS need not be specified if the address has not changed since the previous use of the MFS parameter list.

TS
The address of MFS temporary storage acquired by the caller. TS need not be specified if the address has not changed since the previous use of the MFS parameter list.

DSLMLIT: Defining the Literal Table DSLMLITT

Use the DSLMLIT macro to define the literal table DSLMLITT. DSLMLITT is a kind of resource file and collects many of the literals used in MERVA ESA. You can use the literals defined here when defining a screen or printer field with the DSLLDFLD macro.

The literal table consists of a sequence of DSLMLIT macros. The last statement of the table must be the END statement.

No assembler language statements except SPACE, EJECT, END, and PRINT can be used in the table definition.

If errors are detected in a specific DSLMLIT macro, appropriate MNOTEs are provided.

Generating a Literal Table

Name	Operator	Operands
[*label*]	**DSLMLIT**	**TYPE=**{**INITIAL**|**FINAL**}

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
TYPE=INITIAL must be the first literal table definition. If the first statement does not contain a label, DSLMLITT is used. In the second and following calls, the label is disregarded. TYPE=FINAL must be the last literal table definition statement.

Generating a Literal Table Entry

Name	Operator	Operands
[*label*]	**DSLMLIT**	[**TYPE=ENTRY**] ,*number*,*language*,*literal*[,*attribute1*][,*attribute2*]

Programming Notes:

number
A number between 1 and 999999. The number is used in the LIT parameter of the DSLLDFLD statement.

language
The language code, for example, E for English or C for Chinese.

literal
The literal.

attribute1
An additional attribute specified.

attribute2
An additional attribute specified.

Example of a Literal Table DSLMLITT

A literal table definition may look like this:

```
***********************************************************************
* DSLMLITT ASSEMBLE - MERVA ESA LITERAL DEFINITION
***********************************************************************
         MACRO
&LABEL   L     &NUM,&LAN,&LIT,&ATTR1,&ATTR2,&TYPE=ENTRY
&LABEL   DSLMLIT &NUM,&LAN,&LIT,&ATTR1,&ATTR2,TYPE=&TYPE
         MEND
*
         DSLMLIT TYPE=INITIAL
*----------------------------------------------------------------------
 L 0001,E,'????????'                                          UNKNOWN!
 L 0002,E,'                Day  Address       ISN'           DWSSWS20
 L 0003,E,'                Day  Address       OSN'           DWSSWS20
 L 0004,E,'               D/C'                               DWSSW37H
 L 0005,E,'           Day  Address       ISN'                DWSSWS20
 L 0006,E,'          Neg.Rate'                               DWSSW*
 L 0007,E,'     Day  Address       ISN'                      DWSSWS20
 L 0008,E,'     Day  Address       OSN'                      DWSSWS20
 L 0009,E,'   Currency'                                      DWSSW3*
...
*----------------------------------------------------------------------
         DSLMLIT TYPE=FINAL
         END
```

Figure 1. The DSLMLITT Table

DSLMPFK: Defining the Program Function-Key Table

The DSLMPFK macro defines a Program Function-Key table (PF-Key Table). A PF-key table is used by MERVA ESA during a user session. The table defines which command is to be executed when the user presses a PF key. The table name to be used can be defined in the user profile for each user individually. This PF-key table name can be changed with the PFKEYS screen command. If neither a PF-key table is defined in the user profile nor the PFKEYS command is issued, a default PF-key table name as specified in the function table entry is used. The current PF-key settings can be inspected by using the SHOW DSLHPFK command.

The PF-key definitions are defined in groups that refer to the actual function. The group number is specified in the PFGROUP= parameter of the function table entry. The GROUP parameter allows the specification of different PF-key settings for different user functions in the same PF-key table.

Within each PF-key table up to 255 PF-key information lines can be specified. These PF-key information lines can be displayed on a reserved line on the screen and give a direct information on the current PF-key settings. To display the *n*-th PF-key information line, the field PFKLINE, DANUM=*n* has to be specified in the screen device section of the MCB which is used to define the display format. This MCB is preferably a frame definition MCB, which is specified in the FRAME= parameter of a DSLFNT macro.

The following is a description of the three types of the macro:
- DSLMPFK TYPE=INITIAL to define a PF-key table header
- DSLMPFK TYPE=ENTRY to define a PF-key entry, or a PF-key information line
- DSLMPFK TYPE=FINAL to close the definition of a PF-key table

Generating a Program Function-Key Table Header

Name	Operator	Operands
[*label*]	**DSLMPFK**	**TYPE=INITIAL**

Generating a Program Function-Key Table Entry

Name	Operator	Operands
	DSLMPFK	[**TYPE=ENTRY**] [**,AID=***value*] [**,CMD=***literal*] [**,CS=**{<u>**NO**</u>|**YES**}] [**,GROUP=***nnn*] [**,PFKLINE=***literal*]

Ending the Program Function-Key Table Definition

Name	Operator	Operands
	DSLMPFK	**TYPE=FINAL**

Programming Notes:

label
The label of the DSLMPFK TYPE=INITIAL macro is used to define the name of the PF-key table. This name must be used in the PFKEY= parameter of the DSLFNTT macro instruction, and in the user profile of the MERVA ESA user file records. For TYPE=ENTRY and TYPE=FINAL definitions, the label is ignored. The default is DSLMPF00.

TYPE
The type of table entry.

INITIAL	Defines the PF-key table header. This must be the first DSLMPFK call.
ENTRY	Defines a PF-key definition, or PF-key line. This is the default. If this is specified, either AID= or PFKLINE= must also be specified.
FINAL	Finishes a PF-key definition. This must be the last DSLMPFK call.

AID
The attention ID value for this definition. This value can be either a number from 0 to 255, or a hexadecimal constant from X'00' to X'FF'. The following symbolic names for 3270 attention IDs are generated by the macro; these names can be used in the AID= specification.

```
DEFAULT  X'00'    PF01  X'F1'    PF13  X'C1'
NOAID    X'60'    PF02  X'F2'    PF14  X'C2'
ENTER    X'7D'    PF03  X'F3'    PF15  X'C3'
PA1      X'6C'    PF04  X'F4'    PF16  X'C4'
PA2      X'6E'    PF05  X'F5'    PF17  X'C5'
PA3      X'6B'    PF06  X'F6'    PF18  X'C6'
CLEAR    X'6D'    PF07  X'F7'    PF19  X'C7'
SYSREQ   X'F0'    PF08  X'F8'    PF20  X'C8'
                  PF09  X'F9'    PF21  X'C9'
                  PF10  X'7A'    PF22  X'4A'
                  PF11  X'7B'    PF23  X'4B'
                  PF12  X'7C'    PF24  X'4C'
```

CMD
The command connected to a PF key. You can only specify CMD when AID is specified.

CS
CS=YES specifies that this PF key can be used together with the cursor selection capability. The cursor selection source data and target field are defined in the DSLLDFLD macro of the MCB, using the parameters CSSRC and CSTARG, respectively.

When both CMD and CS=YES are specified for one key, the cursor selection source data is concatenated to the CMD literal and the result is placed in the cursor selection target field.

CS can only be used when AID is also specified.

NO defines that this PF key cannot be used together with the cursor selection capability. CS=NO is the default.

GROUP
A PF-key group number. The corresponding specification in the function table entry determines whether a PF-key definition is effective. A specification of

GROUP=0 indicates that the specification is valid for all PF-key groups. This value remains in effect for all following definitions until another group specification is made.

The following group numbers are used by MERVA ESA and its components, the SWIFT Link and the MERVA Link:

```
  0        Default when not specified in a higher group
  1 to 3   Reserved
  4        Function selection group and panel
  5 to 7   Reserved
  8        Operator command group
  9          Operator command panel
 10 and 11 Reserved
 12        User File maintenance group
 13          User File maintenance record panel (update)
 14          User File maintenance record panel (read only)
 15          User File maintenance list panel
 16        Message processing group and selection panel
 17          Message processing PROMPT panel
 18          Message processing NOPROMPT panel
 19        Message selection list panel
 20        General file maintenance group and selection panel
 21          General file maintenance record panel (update)
 22          General file maintenance record panel (read only)
 23          General file maintenance list panel
 24        Authenticator-Key File file maintenance group
           (SWIFT Link only)
 25          Authenticator-Key File file maintenance record panel
             (update and authorization)
 26          Reserved
 27          Authenticator-Key File file maintenance list panel
             (update and authorization)
 28          Reserved
 29          Authenticator-Key File file maintenance record panel
             (display only)
 30          Reserved
 31          Authenticator-Key File file maintenance list panel
             (display only)
 32          Reserved
 33          Authenticator-Key File file maintenance record panel
             (update)
 34          Reserved
 35          Authenticator-Key File file maintenance list panel
             (update)
 36          Reserved
 37          Authenticator-Key File file maintenance record panel
             (authorization)
 38          Reserved
 39          Authenticator-Key File file maintenance list panel
             (authorization)
 40        Reserved
 41        MERVA System Control Facility MERVA Link panels
 42        MERVA System Control Facility MERVA ESA operator command panel
 43 to 100 Reserved
101 to 247 Free for user
248        For display with the SHOW command
252        For display with the HELP command
```

Note: The groups defined for MERVA ESA and its components, SWIFT Link and MERVA Link, must not be changed.

PFKLINE
A PF-key information line that can be displayed on screen. Up to 255 PF-key

information lines can be specified in a PF-key table. When AID is specified, PFKLINE may not be specified. When PFKLINE is specified, neither CMD nor CS may be specified.

DSLMPFK Macro Programming Notes

The MFS print and edit service programs use the PF-key table to generate the command connected to a received attention ID. The command is written into the TOF as field DSLCMDL.

The system field separation routine extracts the PF-key information lines from the current PF-key table. Up to 255 PF-key information lines can be specified in a table. The data area index is used to select a specific information line. The DSLMMFS interface provides an access to the current PF-key table. The type code for this request is TYPE=GETPFK. This interface is functionally equivalent to the TYPE=GETDEV load interface. The address of the requested PF-key table is communicated in a field of the MFS permanent storage. PF-key tables must be specified in the MFS program table. (See Generating the MFS Program, Exit Definition, and MCB Link Table.)

DSLMPT: Defining the MFS Program Table Entry

Generating the MFS Program, Exit Definition, and MCB Link Table

The macro can be used to generate the MERVA ESA MFS program, exit definition, and MCB link table (DSLMPTT).

This table is exclusively used by the module DSLMMFS to set up the linkage to MFS programs, terminal exit routines, user exits, field checking exits, default setting exits, editing exits, field component separation exits, and field expansion exits. Additionally, table entries for prelinkage of frequently used MCBs can be defined.

All programs are identified by *module numbers* from 1 to 32767 for each class of modules and exits. These numbers are used to build internal names and, as reference indication in the MERVA ESA MCB, definition statements DSLLFLD, DSLLSUBF, and DSLLMFLD. External program names and entry names in more complex modules can be used and should be specified in the appropriate entries. For further description on how to define MFS programs and exits, refer to “DSLMFS: Defining the Message Format Services” on page 106.

Name	Operator	Operands

	DSLMPT	**TYPE=**{**INITIAL** }
		{**MFS** }
		{**C**|**CHECK** }
		{**D**|**DEFAULT**}
		{**E**|**EDIT** }
		{**M**|**MCB** }
		{**P**|**PFKSET** }
		{**S**|**SEPR** }
		{**U**|**USER** }
		{**X**|**EXPAND** }
		{**FINAL** }
		
		,NAME={(*modname*[,*entry*])}
		{*mcbname* }
		{*pfkeyset* }
		
		,NUMBER={*nnnnn*|**ALL**}
		
		[**,LANG=**{**NO** }]
		{**ASSEMBLER**|**ASM**}
		{**COBOL** }
		{**PL/I**|**PLI** }
		{**C** }
		
		[**,LINK=**{**YES**|**NO**}]

Programming Notes:

TYPE
The type of the MERVA ESA MFS program table entry to be generated:

INITIAL
The MFS program table header is to be generated.

MFS
An entry for an MFS formatting service module is to be generated. This is

the default. The module numbers are predefined by the TYPE and MEDIUM codes of the MFS program codes. For MFS internal functions and presentation modules:

```
MODNUM = TYPE/4               (for Medium  =  MFS )
MODNUM = TYPE/4 + 32*MEDIUM    (for all other media)
```

Note: For TYPE and MEDIUM codes refer to the MFS parameter list definitions.

C | CHECK
An entry for the MFS field checking program is to be generated. Refer to the parameter CHECK of the DSLLFLD statement.

D | DEFAULT
An entry for the MFS default setting program is to be generated. Refer to the parameter DEFAULT of the DSLLFLD statement.

E | EDIT
An entry for the MFS data editing program is to be generated. Refer to the parameter EDIT of the DSLLFLD and DSLDFLD statements.

M | MCB
An entry for an MCB is to be generated. Frequently used MCBs can thus be link-edited to the MFS program table and need not be loaded dynamically.

P | PFKSET
An entry for a PF-key table is to be generated. Refer to the DSLMPFK macro description.

S | SEPR
An entry for an MFS field separation program is to be generated. Refer to the parameter SEPR of the DSLLFLD statement.

U | USER
An entry for an MFS user exit program is to be generated. The program numbers 1 to 999 are reserved for use by MERVA ESA.

X | EXPAND
An entry for an MFS field expansion program is to be generated. Refer to the parameter EXPAND of the DSLLFLD statement.

FINAL
Shows the end of the DSLMPTT definition.

NAME
Modname specifies the external name of the program by which it can be retrieved from the module library for link-editing or loading

- For TYPE=MCB, *mcbname* specifies the name of the MCB to be link-edited to the MFS program table.
- For TYPE=PFKSET, *pfkeyset* specifies the name of the PF-key table to be link-edited to the MFS program table.
- For program names, which are entries in major programs, *entry* must be specified as a reference. This is the entry name of the program as specified in the external module. If it is not specified, the entry name is determined as described below.

If the NAME= parameter is not specified, the internal program name DSLMannn as described above is used for link reference. When building the

8-byte name, leading zeros are provided. Only the last 3 digits are used to build the name. For numbers from 1 to 32767, a label of the form **DSLMannn** is generated, where:

a	The first letter of the TYPE parameter (X is used for EXPAND)
nnn	The number specified in MODNUM

For high-level language user exits, do not specify an entry name.

NUMBER

One of the following:

- A program number from 1 to 32767 for the specific program
- Exit class to be introduced into the MERVA ESA MFS
- ALL to define a default entry for all remaining numbers, which are not explicitly defined in the MFS program table

If the number of a module name is not defined in the MPT, but an entry exists with NUMBER=ALL, the module name is determined as follows: The first 5 characters of the NAME parameter and the last digits of the program number are concatenated. Leading zeros are provided, when building the 8-byte name.

Note: The NUMBER parameter should not be specified for MCBs and PF-key tables.

LANG

The programming language of the user exit and the calling method used to call the user exit:

- When LANG=NO is specified, the user exit must be written in Assembler language and is called directly by DSLMMFS. This is the default.
- In all other cases the exit is called in the initialized environment of the specified language. For CICS the exit must be coded as a CICS program and is called with EXEC CICS LINK. The name of the module is specified with the NAME= parameter; an additional entry name is not allowed.

LINK

Whether the program is to be link-edited to DSLMMFS (LINK=YES) or loaded dynamically before execution (LINK=NO). The default is NO. Dynamically loaded programs must be defined in a CICS program definition. A high-level language user exit must not be link-edited to DSLMMFS.

Note: Care should be taken when defining external program or program entry names, to avoid conflicting references during link-editing of the module DSLMMFS. Especially the use of identical names for MCBs and programs must be avoided.

DSLMSG: Defining the Diagnostic and Error Message Table

The DSLMSG macro is used to define error messages, diagnostic messages, and command responses. All MERVA ESA messages, issued during execution time, are combined in one table. The name of the table can be defined in the MERVA ESA customizing parameter module DSLPRM (see “DSLPARM: Generating the DSLPRM Module” on page 174).

Note: MNOTEs issued by MERVA ESA macros are not contained in the MERVA ESA operator message table.

This table is loaded by all MERVA ESA application programs.

Messages can be defined in different languages according to the LAN parameter. The application program selects the language according to the definition in a function table entry (see macro DSLFNT), or takes the default 'E' if no language is defined. For user sessions the language can be defined in a user profile or set by the FORM (LANGUAGE) command.

For unsolicited command responses and error messages only one language per MERVA ESA installation is supported.

Name	Operator	Operands
[*label*]	**DSLMSG**	[**TYPE=**{**INITIAL**|<u>**ENTRY**</u>|**FINAL**}] **,***'message text'* [**,LAN=**{<u>**E**</u>|*lan*}] [**,MARKER=**[c]] [**,USE=**{<u>**ALL**</u>|**VSE**|**MVS**}]

Programming Notes:

label
For TYPE=INITIAL, the name of the message table. For TYPE=ENTRY the label specifies the identification of the message. The message identification has the structure *aaannni* or *aaannnn*.

aaa
The component identification which is:

DSL	For the Base Functions
DWS	For the SWIFT Link
EKA	For the MERVA Link and for the FMT/ESA with MERVA Link
ENL	For the Telex Link

nnn **or** *nnnn*
A unique 3- or 4-digit number identifying the message.

i An action indicator that is used only with 3-digit number and can be one of the following:

I	For information messages
A	For messages requiring an operator action
W	For warning messages

E	For error messages

The message identification is used for the retrieval of the messages from the message table. The macro DSLOMS is used to retrieve a message.

TYPE
The type of the message table definition. TYPE=INITIAL must be the first definition for an operator message table. TYPE=FINAL must be the last definition for an operator message table. The default is TYPE=ENTRY.

'message text'
The text of the message. It can contain literals and variables. Variables are defined using the '@' sign and a 1-digit numeric identification, for example, @0, @1, @2. The number must be specified according to the variable definitions in the DSLOMS parameter list used for retrieval of the message (see "DSLOMS: Formatting Display Messages" on page 171).

Note: For VSE: The positional parameter 'message text' must precede any value parameters.

LAN
The 1-byte language identification of the message. This identification must be specified in the DSLOMSG parameter list to retrieve the correct message. The default value is E.

MARKER
A character that is to be inserted between the message identification and the message text. The marker can be used as an action indicator. The default is a blank.

USE
For which environment the message is to be generated:

ALL	The message is used in all MERVA ESA environments. This is the default.
VSE	The message is to be used only for MERVA ESA running in VSE.
MVS	The message is to be used for MERVA ESA running in MVS.
CICS, IMS	These values are no longer supported as there is only one MERVA ESA product for MVS.

This parameter need not be coded if MERVA ESA is run only in one environment.

DSLMTT: Defining the Message Type Table

The macro can be used in two ways, to:

- Map the MERVA ESA message type table header, entry section, and index layout
- Generate the MERVA ESA message type table (DSLMTTT)

Mapping the Message Type Table Sections

Name	Operator	Operands
[*label*]	**DSLMTT**	**TYPE=**{**MAP**|**INITIAL**|**FINAL**} [**,NETSPEC=**(*specification list*)]

Programming Notes:

label
The name for the entry DSECT or the name of the entry mapping (INITIAL) to be generated.

When the header is mapped with TYPE=MAP, the name is built from the first 3 characters of this parameter concatenated with the literal 'HEAD'. For the header and entry section, MTTHEAD and MTT are used by default, respectively. If NETSPEC is specified, the specification list is mapped also for the entry section.

TYPE
How the data areas of the message type table are to be mapped:

MAP	Generates DSECTs of the header and entry sections.
INITIAL	Generates a CSECT, the message type table header section and starts the definition of a message type table. This must be the first macro call for a message type table definition.
FINAL	Generates the last table entry and initializes the table header values and the sorted table index. This must be the last macro call for a message type table definition.

NETSPEC
Defines the network characteristics of a message identification. See description of NETSPEC in the following.

Generating the Message Type Table

In this table, all message types to be used in a MERVA ESA installation for mapping by the MERVA ESA MFS components are defined.

All references to specific network identifications, allowed nesting of message types, and used MCBs are set up in the tables.

Name	Operator	Operands
[*label*]	**DSLMTT**	[**TYPE=ENTRY**]
		,MTYPE=*mtype*
		[**,APINFO=**([*network*][,*category*][,*message type*])]
		[**,CHECK=***nnn*]
		[**,DESCR=***quoted literal*]
		[**,LENGTH=***nnnnn*]
		[**,MCB=***mcbname*]
		[**,MTGEN=**{**YES**|**NO**}]
		[**,NESTING=**{**NO**|**YES**}]
		[**,NETSPEC=**(*specification list*)]
		[**,NLIND=***cccccccc*]
		[**,NL0=**{**NO**|**YES**}]
		[**,PANEL=**{**NO**|**YES**}]
		[**,SYNONYM=**(*list of synonyms*)]

Programming Notes:

label
An optional assembler label for the entry.

TYPE
The type of MERVA ESA message type table entry to be generated. TYPE=ENTRY indicates that a table entry is to be generated. This parameter is optional.

MTYPE
An 8-byte alphanumeric message identification. It is used to uniquely distinguish the message formats during MERVA ESA message generation and mapping (see also the description of the parameter MSGID of the macro DSLMFS on page 108). A message identification consists of a 1-byte network identifier followed by a 1- to 7-byte message type. The following 1-byte network identifiers are used for message type determination and generation:

0	MERVA ESA internal messages
S	SWIFT Link
T	Telex Link
E	Financial EDIFACT messages

APINFO
Application-specific information for the message type. This information is required only by the MERVA Message Processing Client, it is not used by MERVA ESA. The values for SWIFT, telex, and EDIFACT messages are derived by default from the MTYPE parameter.

CHECK
The exit number of a message-checking program, which is executed for

message-checking operations. This exit may carry out additional checking on a given message. When this parameter is not specified, no additional message-specific checking exit is called during message checking.

When PANEL=YES is specified, this parameter is ignored. Panels cannot be checked by a message-checking routine.

Note: The MFS user exit 9 (DSLMU009) is called at the end of all message-checking operations. If both exits are specified, such as CHECK=*nnn* in the MTT entry and user exit 9 in the MFS program table, the message-specific exit is executed first, then the general exit program, user exit 9.

DESCR
A description for the message identification. The description is displayed at the top of each message identification if the field DSLMIDN is defined in the screen section of the top frame MCB DSL0TOP. The field DSLMIDN is filled by the separation routine DSLMS904 with the description. The literal can be up to 50 characters long. If this length is exceeded, the literal is truncated from the right. If the text is shorter than 50 characters, the separation routine inserts blanks so that the text appears as a centered header line.

LENGTH
Defines a maximum length for the message in its external line format. If the length of the message after formatting is greater than the value specified, a warning reason code is issued by the line formatter. The SWIFT Link programs observe this warning and will not send the message to SWIFT, but will route the message with an error indication. If this parameter is not specified, no length checking is performed.

MCB
Defines the 8-byte alphanumeric name of the Message Control Block (MCB), in which the corresponding message formats are defined. If this parameter is not specified, the MCB name is built by concatenating 'DSL', network identification, and then the last 4 characters of the message type.

MTGEN
Whether a message with the specified message identification may be generated by the message-processing command MT. If MTGEN=NO is specified, a message with the specified message identification can only be generated by an application program within MERVA ESA, but not by a user. Default is MTGEN=YES.

When PANEL=YES is specified, this parameter is ignored and MTGEN=NO is assumed.

NESTING
Whether message types can be nested in this message type. When nesting is allowed for a message identification and the MCB which describes the message identification contains the appropriate DSLLEXIT statement, then nesting is possible. Default is NESTING=NO.

When PANEL=YES is specified, this parameter is ignored and NESTING=NO is assumed.

NETSPEC
Defines a list of up to 9 valid assembler data-constant definition items which are generated as assembler constants into the message type table entry. The format of these subparameters has to be as specified for the operands of an assembler DC (Define Constant) instruction.

Each operand consists of 4 subfields:

```
[ Duplication Factor ] { Type } [ Length Modifier ] { Nominal Value }
```

Besides these forms the following is supported:

'*charstring*'	Generated as C'charstring'
term	Generated as AL1(term)

This information can be used by network-specific routines.

Example:

```
NETSPEC=(AL2(3),2XL1'15',C'MTTE',4,'TEXT')
```

The DSLMTT macro generates into the field MTTSPEC0 the length of the network-specific part.

The layout of the network-specific part can be mapped with the TYPE=DSECT macro expansion. The specification list items are mapped with labels MTTSPEC1 to MTTSPEC9.

The MERVA ESA Message Format Service provides an access to a message type table entry via DSLMFS service call:

```
DSLMFS TYPE=GETMTT,MSGID=addr,OUTBUF=bufaddr
```

The message type entry is copied into the specified output buffer; if the request fails, an appropriate return and reason code is given.

Application Example

A message identification is specified with:

```
AUTHENT  EQU     X'80'
         DSLMTT  ... ,NETSPEC=(AUTHENT)
```

The application program can use the following:

```
AUTHENT  EQU    X'80'
         DSLMTT TYPE=MAP,NETSPEC=(AUTHENT)
MTTACT   EQU    NETSPEC1
         .
         .                        get table entry via DSLMFS
         .
         TM     MTTACT,AUTHENT    inspect setting of authent
                                  for a table entry
```

See also the *MERVA for ESA Customization Guide*.

NLIND
The nesting level indicator to contain the message identifier in the TOF. The default is DSLEXIT.

NL0
Whether the message fields are to reside on nesting level 0. The default is NO; that is, a nesting level is added to the appropriate nesting level indicator to contain the message fields.

PANEL
Whether the message type is used as an internal panel identifier only (PANEL=YES), or whether the message identification can be initialized in the TOF and processed as a message (PANEL=NO). The default is PANEL=NO.

Examples: MTYPE=0SON or MTYPE=0SOF which are used as panel identifications only.

SYNONYM

An 8-byte alphanumeric synonym, or a list of synonyms enclosed in parentheses.

These synonyms can be used as parameters for the HELP, SHOW, or the MT command in place of the original message identification. This allows for more intelligible commands, for example, 'SHOW ERRORS' instead of 'SHOW 0ERR'.

DSLNCM: Defining a Command Table

The DSLNCM macro defines MERVA ESA commands.

The commands generated by DSLNCM are contained in command table modules that are used by the MERVA ESA command parsing module for formal command analysis. For details on customizing MERVA ESA commands refer to the *MERVA for ESA Customization Guide*.

The following command tables are available:

- DSLNCMT for the MERVA ESA operator commands (user-defined operator commands must also be included in DSLNCMT)
- DSLMCMDT for the MERVA ESA Message Format Service commands used by users working at screen terminals (screen commands)
- DSLECMDT for the MERVA ESA User commands (session commands)
- DSLECCMT for the MERVA ESA queue utility commands (see the *MERVA for ESA Customization Guide*)
- DSLEFCMT for the MERVA ESA general file maintenance
- DSLEMCMT for the MERVA ESA message-processing functions (this table is generated with the DSLGEN macro)
- DSLEUCMT for maintenance of the MERVA ESA user file
- DWSECMDT for maintenance of the SWIFT Link authenticator-key file
- EKAMSCMT for the MERVA ESA System Control Facility

Name	Operator	Operands
[*label*]	**DSLNCM**	*code* [,*parm1*,*parm2*,....,*parmn*] [,**TYPE**={**INITIAL**|**FINAL**}] [,**DESC**=*cccccccc*] [,**LANG=HLL**] [,**NAME**={<u>**0**</u>|*pgmname*}] [,**PERR**={<u>**NO**</u>|**YES**}] [,**SYN**=*cccccccc*]

Programming Notes:

label

The command word that an operator or user should enter to run this command. The label must be unique in all command tables used in a MERVA ESA command parsing call. The same command words can be used in the user function program command tables, DSLECCMT, DSLEFCMT, DSLEMCMT, DSLEUCMT, DWSECMDT, and EKAMSCMT.

Note: As the MERVA ESA command parsing allows the use of abbreviations of command words up to at least 4 characters, care should be taken that the abbreviations are unique.

label can have 1 to 8 characters. Labels of 1, 2, or 3 characters cannot be abbreviated when the command is entered, but the same command code (see below) can be defined several times thus specifying abbreviations explicitly in the command table.

code

The command code associated with the command. It is the first positional parameter. The command code is a 2-digit numeric value that is only used by the command execution routine. If used, the code must be unique for the same command with different labels or explicit abbreviations of the label.

Note: Command execution routines should use the command code rather than the command label when checking which command was entered.

If a command execution routine processes only one command, the command code can have any value.

parm1,parm2,....,parmn

Specify the command parameters. A maximum of 16 parameters can be specified. The presence of a parameter specification shows to the MERVA ESA command parsing program that the parameter must be formally checked. Each parameter specification must be coded as follows:

```
(MAN,NUM,nn)
```

MAN This parameter must be entered together with the command. The absence of MAN indicates that this parameter needs to be entered only for a particular use of the command.

NUM This parameter is numeric, that is, can consist of the digits 0 to 9 only, and the parameter is presented to the command execution routine right-justified with leading zeros. The absence of NUM indicates that the command can contain any character, and the parameter is presented to the command execution routine left-justified.

nn Shows both the maximum length of the parameter and the length of the token reserved for this parameter by the MERVA ESA command parsing. You can specify values from 1 to 63 for *nn*. With these tokens, the command execution routine finds the parameters always at the same place, no matter how long the parameters entered are.

Note: The parameter specifications MAN, NUM, *nn* can be specified in any order, and no leading commas are necessary if a parameter is omitted. The following are all valid:

```
(MAN,NUM,8)
(NUM,8)
(MAN,8)
(MAN,8,NUM)
(8,NUM,MAN)
(8)
```

TYPE

The beginning and the end of a command table. If TYPE is used, no command table entry is generated. That is, for a command definition the type parameter must not be specified.

INITIAL

The beginning of the command table. The label field is used as the CSECT name (command table name).

FINAL
The end of the command table. It sets the indicator for the last command table entry.

Note: TYPE=INITIAL and TYPE=FINAL can be specified only once each in a command table.

DESC
The descriptive command name that is used in command responses and for checks of the End-User driver if a user is allowed to use a command (this applies only to user commands and not to operator commands). This parameter can be up to 8 characters long. It is recommended, when an abbreviation or a synonym is defined for a command. The default is the command word.

LANG
LANG=HLL specifies that the command execution routine is written in a high-level language and that the parameter passing method is by address list rather than by registers. This parameter is only valid for entries of the DSLNCMT. If this parameter is omitted, the parameter passing method is as in MERVA/ESA V3.1. For more information, refer to the chapter on how to create MERVA ESA operator commands in the *MERVA for ESA Customization Guide*.

NAME
The name of the command execution routine. This routine is called after successful MERVA ESA command parsing for execution of the command. It can be a separate module or can be part of an other module, yet it must be a valid external reference name for the linkage editor. By default NAME=0 is assumed, which allows specific processing of the program calling MERVA ESA command parsing. In DSLNCMT, NAME must always be specified.

PERR
Whether the command execution routine is to be called even when MERVA ESA command parsing is not successful. This parameter can be used in the DSLNCMT command table only. The default is NO.

SYN
A command word different from the label, if the command word contains characters not allowed by the assembler in the label field, for example, special signs such as >. The label field can contain any name, if SYN is used.

DSLNIC: Defining the Intertask Communication

The DSLNIC macro serves the following purposes:

- Request a central service
- Allocate or free an interregion communication block (ICB)
- Map the parameter list for DSLNICT

The MERVA ESA intertask communication interface DSLNICT performs intertask requests. In the following description, the word *region* also means partition, if applicable.

The type of the intertask communication chosen depends on the specification of the *ITC* parameter in DSLPRM.

In CICS systems, if the DSLCOM field COMEIB contains an address, intraregion communication is established. If the intraregion communication does not find a MERVA ESA running in the same region, and CINTER=YES is specified in DSLPRM, an interregion communication is tried to a MERVA ESA running in another CICS region.

If the DSLCOM field COMEIB is empty, interregion communication is used. IMS systems always use interregion communication.

Requesting a Central Service and Checking MERVA ESA Status

Name	Operator	Operands

[*label*]	**DSLNIC**	**TYPE=**{**REQ**|**REQDYN**}
		,BUF={*addr*|(*r*)}
		,MF=(E,{*addr*|(*r*)})
		,NAME={*cccccccc*|(*r*)}
		[**,PL=**{*addr*|(*r*)}]

Programming Notes: For the address parameters in this macro, the following notations are possible:

addr A symbolic label.

(r) A general register containing the address.

label
A unique statement label according to assembler language conventions.

TYPE
A request for services.

REQ Requests a MERVA ESA central service.

REQDYN
The request may return a dynamic buffer in the DSLNIC parameter list in case the provided parameter list buffer or data buffer is too small to contain the data returned by the central service.

Refer to *MERVA for ESA Customization Guide* for details about handling of dynamic buffers. It is the responsibility of the caller to release the storage when the buffer is no longer needed.

BUF
The location of a data buffer for a central service request. It must be a 4-byte field or general register containing the address of the buffer. The maximum length of this buffer may not be larger than specified in DSLPRM by the MAXBUF parameter of the DSLPARM macro. The size of this buffer can exceed 32KB.

The use of BUF depends on the service that is requested:

- If NAME=DSLNMOP, BUF specifies the address of the MERVA ESA operator message.
- If NAME=DSLQMGT, BUF specifies the address of the message to be retrieved from or to be stored in the queue data set.
- If NAME=DSLNCS, BUF specifies the address of the MERVA ESA command and response buffer obtained with a DSLNMO TYPE=BUF macro.
- If NAME=DSLNUSR, BUF specifies the address of the buffer necessary for data exchange with DSLNUSR.
- If NAME=DSLJRNP, BUF specifies the address of the buffer necessary for data exchange with DSLJRNP.
- If NAME=DWSAUTP, BUF specifies the address of the buffer necessary for data exchange with DWSAUTP in DWSAUT TYPE=UPDATE and DWSAUT TYPE=CONT calls. For DWSAUT TYPE=AUT, this parameter specifies the address of the buffer that contains the SWIFT message to be authenticated.

MF
The format of the macro.

The first parameter must be E for execute form. The second parameter specifies the label of the DSLNICT parameter list or a general register containing the parameter list address. It must be the same as the address specified in the MF operand when the ICB was allocated by the macro DSLNIC TYPE=ALLOC.

NAME
The name of the servicing program or a general register containing the address of an 8-byte field containing the program name. If a literal is used, it may be up to 8 characters long. The name specified must be contained in the MERVA ESA task server request table DSLNTRT.

PL
The location of the parameter list required by the servicing program specified in the NAME operand. It must be a 4-byte field or general register containing the address of the parameter list. The maximum length of this parameter list is specified in DSLPRM by the NICPL parameter of the DSLPARM macro.

Allocating and Freeing an ICB

Name	Operator	Operands
[*label*]	**DSLNIC**	**TYPE=**{**ALLOC**|**FREE**} **,MF=(E,**{*addr*|(*r*)}) [**,COM=**{*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that DSLCOM is addressable.

label
A unique statement label according to assembler language conventions.

TYPE
The following functions of this macro:

ALLOC	Allocates an ICB. ALLOC can also be used to check, if MERVA ESA is still running and if an ICB was allocated earlier.
FREE	Frees an ICB.

MF
The format of the macro.

The first parameter must be E for execute form. The second parameter specifies the label of the DSLNICT parameter list or a general register containing the address of the parameter list.

COM
For TYPE=ALLOC, COM specifies the label of the MERVA ESA communication area DSLCOM or a general register containing the DSLCOM address. The default is DSLCOM.

Note: After an ALLOC request, the address of the DSLNICT parameter list must not be changed.

Mapping the Parameter List for DSLNICT

The DSLNIC macro may be used to generate a storage description of the DSLNICT parameter list.

Name	Operator	Operands
[*label*]	**DSLNIC**	**MF=L**

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

DSLNMO: Defining the Operator Interface

The DSLNMO macro serves the following purposes:

- Prepare the parameter list for the MERVA ESA operator interface DSLNMOP (the EP operand determines if the request will be processed as a direct or central service)
- Map the MERVA ESA operator interface parameter list
- Map the MERVA ESA command and response buffer for the nucleus command server DSLNCS

Calling the Operator Interface

DSLNMOP adds messages to the display message (DM) table. It is also used to issue unsolicited operator messages at the console and to add the messages to the MERVA ESA journal.

Name	Operator	Operands

[*label*]	**DSLNMO**	**TYPE=**{**PUT**|**PUTC**|**PUTJ**|**PUTJC**}
		,FROM={*addr*|(*r*)}
		[**,EP=DSLNMOP**]
		[**,MF=(E,**{*addr*|(*r*)}**)**]

Programming Notes: For the address parameters in this macro, the following notations are possible:

addr A symbolic label.

(r) A general register containing the address.

label
A unique statement label according to assembler language conventions.

TYPE
The MERVA ESA operator interface function to be performed:

PUT Adds a message to the display message table, but does not send the message to the system console nor add it to the MERVA ESA journal.

PUTC Adds a message to the display message table and sends it to the system console, but does not write it to the MERVA ESA journal data set.

PUTJ Adds a message to the display message table and writes it to the MERVA ESA journal data set, but does not send it to the system console.

PUTJC Adds a message to the display message table, writes it to the MERVA ESA journal data set, and sends it to the system console.

FROM
The address of a buffer that contains the message to be displayed. The buffer must have the standard MERVA ESA format which is described in *MERVA for ESA Customization Guide*, in the chapter dealing with the buffer standard of MERVA ESA.

EP
The only possible value for the EP parameter is 'DSLNMOP'. It specifies that the request will be executed as a direct service.

The EP parameter must be specified by all programs that are linked to DSLNUC. Before issuing the DSLNMO macro with EP=DSLNMOP, the program must ensure that general register 12 contains the address of the MERVA ESA communication area DSLCOM and that general register 13 points to a usable save area.

The parameter EP=DSLNMOP must not be used by programs that are not linked to DSLNUC. After using the DSLNMO macro to prepare the parameter list, a DSLNIC macro with TYPE=REQ and NAME=DSLNMOP is required to request execution of a central service.

If the EP parameter is omitted or invalid, a central service request is assumed.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the DSLNMOP parameter list or a general register containing the address of the parameter list. The default is MF=(E,(1)).

Mapping the MERVA ESA Operator Interface Parameter List

The *list* form of the DSLNMO macro is used to generate a storage description of the DSLNMOP parameter list.

Name	Operator	Operands
[*label*]	**DSLNMO**	**MF=L**

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

Mapping the MERVA ESA Command and Response Buffer

The MERVA ESA command and response buffer is used by programs that prepare and submit command input for the nucleus command server DSLNCS.

Name	Operator	Operands
[*label*]	**DSLNMO**	**MF=BUF** [**,LINES=**{**<u>10</u>**|*nn*}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro.

It must be BUF for buffer form. BUF maps the MERVA ESA command and response buffer that contains all fields necessary for the execution of a MERVA ESA command. This buffer is required if MERVA ESA operator command execution is requested.

LINES
The maximum number of display lines in the command response. The maximum value for this parameter is 20. The default is 10.

Note: The buffer is used by the command server to store continuation information for multiple response displays. The calling program should never change the continuation data.

DSLNPA: Defining Command Parsing

The DSLNPA macro is used for the following purposes:

- Call the MERVA ESA command parsing for formal analysis of MERVA ESA commands
- Map the command parsing parameter list

Calling DSLNPAR

The DSLNPA macro is used to find the command in a command input buffer and to separate its parameters into the DSLNPAR parameter list.

The caller supplies a list of command table addresses. These command tables have been generated with the DSLNCM macro. The command being parsed is located in the tables and checked for the presence of mandatory and optional parameters, the data type of the parameters (numeric or alphanumeric), and the length of the parameters.

The result is returned in the DSLNPAR parameter list. The parsing provides the full command name and code, the command execution module name and address, and a list of parameters for the command.

For every parameter in the list there is a 1-byte field containing the actual data length followed by a field containing the data from the input buffer. The order of the parameters and their field lengths are defined in the related command table entry. Each parameter data field has the length defined in the command table entry and is padded with blanks (X'40') when the actual data length is less than the maximum. Numeric parameters are right-justified with leading zeros. If a parameter is omitted, the data length field contains zero.

If a validation error occurs, a return code and an index to the first erroneous parameter are returned in the DSLNPAR parameter list.

Parsing returns the address of the current command table entry, and an index to the caller's list of command table addresses.

Name	Operator	Operands

[*label*]	**DSLNPA**	**,FROM=**{*addr*|(*r*)}
		,LIST={*addr*|(*r*)}
		[**MF=(E,**{*addr*|(*r*)}**)**]

Programming Notes: For the address parameters in this macro, the following notations are possible:

addr	A symbolic label.
(r)	A general register containing the address.

label
: A unique statement label according to assembler language conventions.

FROM
: The label of the command input buffer, or a general register containing the address of the command input buffer.

The first 4 bytes of the input buffer must be a binary halfword containing the length of the input data, followed by a reserved halfword and followed by the command input.

LIST

A field containing the address of a list of consecutive fullwords, each containing the address of a MERVA ESA command table, or a general register containing the address of that list.

The last address in the list must have the high order bit (X'80') on to show the end of the list.

After parsing, the field NPACTIND contains an index to the address of the command table where the command was found.

MF

The format of the macro. The first parameter must be E for execute form. The second parameter specifies the address of the parsing parameter list. The default is MF=(E,(1)).

Mapping the Parameter List for DSLNPAR

The list form of the DSLNPA macro is used to generate a storage description of the DSLNPAR parameter list.

Name	Operator	Operands
[*label*]	**DSLNPA**	**MF=L** [**,PREFIX=**{<u>**NPA**</u>|*ccc*}]

Programming Notes:

label

A unique statement label according to assembler language conventions.

MF

The format of the macro. It must be L for list form.

PREFIX

The first 3 characters of the field names in the DSLNPAR parameter list. The default is NPA.

DSLNPT: Defining the Nucleus Program Table

The DSLNPT macro is used to generate the MERVA ESA nucleus program table (DSLNPTT). There is only one DSLNPTT in a MERVA ESA system, and it is used by the MERVA ESA nucleus only. The DSLNPTT enables you to add programs to DSLNUC, such as external network interfaces without changing code in the existing MERVA ESA programs. DSLNPTT is generated with the DSLGEN macro.

Programs named in the DSLNPTT can use the services of MERVA ESA directly via the call interface. Programs not linked to DSLNUC such as other MERVA ESA applications requesting central MERVA ESA services must use the MERVA ESA intertask communication. Yet, all programs named in the DSLNPTT use the resources of DSLNUC concurrently, especially the CPU time. Care must be taken that a program of the DSLNPTT does not keep control too long in order not to disturb the processing of the other programs.

It is possible to run a program defined in the DSLNPTT as a separate task. This can be done by defining the program in the nucleus server table DSLNSVT as SERVER=TASK. In this way it is possible to minimize the effect on the processing of the other programs.

When running MERVA ESA in a multisystem environment, each MERVA ESA instance must have link-edited the same nucleus program table.

Name	Operator	Operands
[*label*]	**DSLNPT**	**TYPE=**{**FINAL**|**INITIAL**|**INTER**|**INTRA**|**PGM**}
		[**,AUTO=**{**NO** }
		{**YES** }]
		{**CICS** }
		{**NOTCICS** }
		{**IMS** }
		{**NOTIMS** }
		{**BATCH** }
		{**NOTBATCH**}
		[**,DESC=***name*]
		[**,ECB=**{**0**|*number*}]
		[**,ECBREQ=**{**0**|*number*}]
		[**,LANG=HLL**]
		[**,MF=D**]
		[**,NAME=***pgmname*]
		[**,PARM=***cccccccc*]
		[**,PRTY=**{**5**|*priority*}]
		[**,STOP=**{**YES**|**NO**}]
		[**,STOPREQ=**{**0**|*number*}]
		[**,STRT=**{**YES**|**NO**}]
		[**,STRTREQ=**{**0**|*number*}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The type of the nucleus program table entry to be generated:

FINAL	Shows the end of the DSLNPTT and causes additional control blocks to be generated.
INITIAL	Shows the beginning of the DSLNPTT and causes the DSLNPTT header to be generated.
INTER	Generates the entry for the DSLNTS for MERVA ESA interregion communication. If TYPE=INTER is not contained in the DSLNPTT, MERVA ESA interregion communication cannot be used, and the DSLNICP program need not be installed.
INTRA	Generates the entry for the DSLNTS for MERVA ESA intraregion communication. TYPE=INTRA is only valid for MERVA ESA running in CICS. If TYPE=INTRA is not contained in the DSLNPTT, MERVA ESA intraregion communication cannot be used.
PGM	Generates the entry for the program with the name indicated with the NAME parameter.

AUTO
Defines the start of a program.

NO	This program is not to be started during MERVA ESA initialization. Instead, the program must be started using the MERVA ESA operator command **start**. This is the default.
YES	This program is to be started during MERVA ESA initialization.
CICS	This program is to be started during MERVA ESA initialization if the MERVA ESA nucleus is running as a CICS task.
NOTCICS	This program is to be started during MERVA ESA initialization if the MERVA ESA nucleus is not running as a CICS task.
IMS	This program is to be started during MERVA ESA initialization if the MERVA ESA nucleus is running as an IMS BMP.
NOTIMS	This program is to be started during MERVA ESA initialization if the MERVA ESA nucleus is not running as an IMS BMP.
BATCH	This program is to be started during MERVA ESA initialization if the MERVA ESA nucleus is running as a native MVS batch program, that is, neither as a CICS task, nor as an IMS BMP.
NOTBATCH	This program is to be started during MERVA ESA initialization if the MERVA ESA nucleus is running either as a CICS task or as an IMS BMP.

DESC
A descriptive name of the program for the responses of the **dp**, **start**, and **stop** commands. For example, SWIFTII is the descriptive name for the program DWSDGPA. DESC cannot be used for TYPE=INITIAL or TYPE=FINAL.

You can use this parameter to create multiple copies of a program specified in the NAME parameter. Servers will be known by the value you assign to the

DESC parameter. If the reason you do this is to increase throughput, you should also define a server entry in the nucleus server table for each name.

ECB

The number of event control blocks that the program needs. The default is 0.

- For TYPE=PGM, these are the ECBs that the program needs for the external network line, or timer, or input/output operations, and so on. The DSLNPT macro sets the correct ECB value for the programs known to MERVA ESA.
  - For NAME=DSLNTSA, the number of ECBs specifies the number of parallel APPC/MVS connections supported by the MERVA ESA intertask communication server for APPC/MVS. Multiple servers of this type can be specified; each intertask communication server needs a different *DESC* parameter specification.
  - For NAME=DSLNTSQ, the number of ECBs specifies the number of parallel sessions supported by the MERVA ESA intertask communication server for CICS temporary storage queues. Each ECB represents one of the CICS tasks requesting MERVA ESA central services via this communication method at the same time.
  - For NAME=DSLNTSM, the number of ECBs specifies the number of parallel sessions supported by the MERVA ESA intertask communication server using MQSeries.
- For TYPE=INTER, these are the ECBs that DSLNTS needs for the MERVA ESA interregion communication. Each ECB represents one of the regions requesting MERVA ESA central services at the same time. For each region, buffer space is required in the interregion communication area DSLICA. The initial size is calculated as the sum of the values specified for NICPL and NICBUF in the DSLPARM macro. The size can increase up to the sum of the values specified for NICPL and MAXBUF in the DSLPARM macro.
- For TYPE=INTRA, these are the ECBs that DSLNTS needs for the MERVA ESA intraregion communication. Each ECB represents one of the CICS tasks requesting MERVA ESA central services at the same time.

Note: Under CICS, the MXT parameter of DFHSIT must be increased by the number of all ECBs defined in DSLNPTT. This number is indicated by MNOTEs when assembling DSLNPTT.

ECBREQ

The content of general register 1 if the program is called when one of its ECBs is posted. If this number is found in register 1, the program should carry out the appropriate processing and clear this ECB. The default is 0. The DSLNPT macro sets the correct ECBREQ value for the programs known to MERVA ESA.

LANG

LANG=HLL specifies that the nucleus program is written in a high-level language and that the parameter passing method is by address list rather than by registers. If this parameter is omitted, the parameter passing method is as in MERVA/ESA V3.1. For more information, refer to the chapter on how to write MERVA ESA application programs link-edited to DSLNUC in the *MERVA for ESA Customization Guide*.

MF

Generates the DSECTs of the DSLNPTT header and the DSLNPTT entry.

NAME

Defines, for TYPE=PGM only, the name of the program. This name must be a valid external reference for the linkage editor. MERVA ESA provides the following programs:

DSLCNTP	Message counter program
DSLISYNP	Synchronization point program
DSLNMOP	Operator interface program for the system console
DSLNMQS	Interservice communication program
DSLNRTCP	Remote task communication program
DSLNSFPP	Timer-controlled start function program
DSLNTS	Intertask communication via memory copy (SVC, XPCC)
DSLNTSA	Intertask communication via APPC/MVS
DSLNTSM	Intertask communication via MQSeries
DSLNTSQ	Intertask communication via CICS temporary storage queues
DWSAUTIN	Initializing the SWIFT authentication
DWSDGPA	Program for connecting to the SWIFT network
DWSDLSK	Loading pregenerated session keys for the SWIFT Secure Login/Select (SLS) into MERVA ESA queues
ENLSTP	Station program of the Telex Link via a fault-tolerant system

PARM

Defines, for TYPE=PGM only, a parameter for the program. The MERVA ESA programs DSLNRTCP and DSLISYNP use this parameter. The MERVA ESA program DSLNTSA uses this parameter to overwrite the TP name specified with the ITCASRV parameter in the DSLPRM module. User-written programs may use the parm field for customization purposes.

If you have specified multiple nucleus task servers for MQSeries (DSLNTSM) to force parallel processing, you must distinguish them by specifying a different DESC parameter. You must also specify a different value in the PARM parameter. Program DSLNTSM uses this parameter to append it to the local MQI receive queue defined in ITCMRCV.

PRTY

The priority of the program table entry. This parameter is valid for all types except TYPE=INITIAL. The priority concept applies to programs running under direct control of DSLNUC. For programs running as separate tasks, the priority has no meaning. Whether a program runs under direct control of DSLNUC or as a separate task is determined by the corresponding specification in the nucleus server table DSLNSVT.

PRTY must have a value from A (lowest priority) to Z or from 0 to 9 (highest priority). The default is 5.

The priority defined in the DSLNPTT can be changed during the execution of MERVA ESA using the operator command **priority**.

The priority in the DSLNPTT defines for DSLNUC the scanning order of the DSLNPTT entries after completion of the multiple wait. A high priority means that this entry is scanned before entries with lower priority are scanned, and if an ECB of a program with high priority is posted, this program gets control.

The ECBs of programs with lower priority are only scanned and these programs given control if none of the programs with higher priority has a posted ECB.

For programs having the same priority, MERVA ESA provides an equal service for all programs by giving control a second time to the same program only if none of the other programs in the group has an ECB posted or has been given control for a posted ECB.

This concept was supported in previous MERVA ESA releases by a specific entry in the DSLNPTT, the dynamic dispatching (DYN) group. As the dynamic dispatching is now applied to all programs in the DSLNPTT, the DYN group is no longer needed.

STOP
Shows whether the MERVA ESA operator command **stop** is allowed for this program.

YES The MERVA ESA operator is allowed to stop this program. This is the default.

NO The MERVA ESA operator is not allowed to stop this program.

STOPREQ
The content of general register 1 if the program is called during execution of the MERVA ESA operator command **stop** or during MERVA ESA termination. If this number is found in register 1, the program should carry out a termination. STOPREQ=0 is the default definition. The DSLNPT macro sets the correct STOPREQ value for the programs known to MERVA ESA.

STRT
The MERVA ESA operator command **start** is allowed for this program.

YES The MERVA ESA operator is allowed to start this program. This is the default.

NO The MERVA ESA operator is not allowed to start this program. Instead, the program must be started during MERVA ESA initialization using the AUTO=YES parameter.

STRTREQ
The content of general register 1 if the program is called during execution of the MERVA ESA operator command **start** or during MERVA ESA initialization if AUTO=YES is specified. If this number is found in register 1, the program should carry out an initialization. The default is 0. The DSLNPT macro sets the correct STRTREQ value for the programs known to MERVA ESA.

DSLNSV: Defining the Nucleus Server Table

The DSLNSV macro is used to generate the MERVA ESA nucleus server table (DSLNSVT). There is only one DSLNSVT in a MERVA ESA system, and it is used by DSLNUC only. The DSLNSVT entries define how nucleus server programs run under control of DSLNUC. It is possible to run nucleus servers (nucleus programs, central services, and command execution routines) as separate tasks. Exploiting the parallel processing of MERVA ESA can increase the throughput of a MERVA ESA system depending on the workload profile. If no nucleus server table entry is defined as SERVER=TASK, the MERVA ESA nucleus runs without using parallel processing, that is, in the same way as in earlier MERVA ESA releases.

With MERVA ESA V4, you can distribute multiple MERVA ESA instances among several systems of a multisystem environment with shared resources. In this case the nucleus server table must be identical on each MERVA ESA instance.

The DSLNSVT has no external references and is loaded dynamically by DSLNUC.

The programs named in the DSLNSVT must also be specified in one of the other MERVA ESA nucleus tables: DSLNPTT, DSLNTRT, or DSLNCMT.

Name	Operator	Operands
[*label*]	**DSLNSV**	**TYPE=**{**DSECT**|**ENTRY**|**FINAL**|**INITIAL**}
		[,**MODNAME=***modulename*]
		[,**NAME=***servername*]
		[,**PARM=***cccccccc*]
		[,**PRIO=**{**0**|*priority*}]
		[,**QNAME=**{*queuename*}]
		[,**SERVER=**{**SUBROUT**|**MAIN**|**TASK**|**BATCHTASK**}]
		[,**SHELL=**{*shellname*}]
		[,**STCNAME=**{*startedtaskname*}]
		[,**SYSNAME=**{*systemname*}]
		[,**TSSIZE=**{**4096**|*nnnn*}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The type of the nucleus program table entry to be generated:

DSECT	Generates the DSECT of the DSLNSVT entry.
ENTRY	Generates the entry for the nucleus server with the name specified with the *NAME* parameter. This is the default.
FINAL	Ends the definition of the DSLNSVT.
INITIAL	Starts the definition of the nucleus server table (DSLNSVT).

MODNAME
A module name for the nucleus server program. The nucleus server program itself is link-edited to DSLNUC, therefore the name must be a valid external reference for the linkage editor. The default is the value specified for the NAME parameter. This parameter is valid for TYPE=ENTRY only.

NAME
Defines the name of the nucleus server. This name must be the same name defined in the corresponding MERVA ESA table DSLNPTT, DSLNTRT, or DSLNCMT. For DSLNPTT, it is the specification of the *DESC* parameter. For DSLNTRT and DSLNCMT, it is the specification of the *NAME* parameter. The *NAME* parameter is valid and mandatory for TYPE=ENTRY only.

PARM
Defines, for TYPE=ENTRY only, a parameter for the nucleus server. The parameter value is filled into an 8-byte field in the server table entry. User-written programs may use it for processing purposes.

PRIO
Defines the priority of the nucleus server program relative to the nucleus main task. The *PRIO* parameter is valid for TYPE=ENTRY only. *PRIO* is a signed integer between -255 and 255 (lowest to highest priority). The default value is 0, which causes the priority to be determined by the system according to the default dispatching priority.

QNAME
An alphanumeric string of up to 16 characters that follows the MQSeries object name rules and that contains the part of the MQI send queue name to which a service request is to be sent. Depending on the third value specified in the ISCMQID parameter of the DSLPARM macro, this name part is either used as a prefix or as a suffix to the base name specified in the ISCMSND parameter of the DSLPARM macro. Both parts are combined to form a unique MQI send queue name.

The part specified in this operand identifies the MERVA ESA instance that should process the service request. However, if QNAME matches the local MERVA ID specified in the second ISCMQID parameter, the service request is processed by the local MERVA ESA instance.

SERVER
Shows the type of the nucleus server defined by this entry.

SUBROUT	This nucleus server is a subroutine of another nucleus server specified earlier in the table. This specification ensures that the subroutine and the nucleus server run in a synchronized way and can share their resources. SUBROUT is the default if the SERVER parameter is not specified.
MAIN	A nucleus server running under direct control of DSLNUC and within the same task as DSLNUC. The interface from DSLNUC to this server is via a direct call interface. All calls are processed synchronously. This specification has to be made when the program should run without using parallel processing under DSLNUC.
TASK	The nucleus server runs as a separate task. This specification has to be made when the nucleus server should exploit the MERVA ESA parallel processing. The processing is done

asynchronously to the other activities executed by DSLNUC or other nucleus servers. In CICS, the nucleus server runs as a CICS task.

BATCHTASK The nucleus server should run as an MVS task, even when executing under CICS. Note that the nucleus server program cannot use any CICS services and the MERVA ESA service program DSLSRVP cannot be used to load other modules or tables.

SHELL
The name of the nucleus server shell program.

For the nucleus running as a CICS task, it is the transaction code followed by an optional terminal code. The transaction code must be defined as a CICS resource. The default value is DSLL. The terminal code specification allows to run a nucleus server on a CICS terminal for debugging purposes.

For the nucleus running as an IMS BMP or native batch program under MVS, it is the name of the nucleus server shell load module. The default is DSLNSHEL.

The *SHELL* parameter is valid for TYPE=ENTRY only.

STCNAME
An alphanumeric string of up to 8 characters containing the started task name of the MERVA ESA instance to be automatically started via the internal START command. The name must reflect an existing cataloged procedure containing the job control to activate a MERVA ESA instance on the system.

This parameter is used by the primary MERVA ESA instance and is only valid if the ISCSTART parameter of the DSLPARM macro is set to *AUTO*.

SYSNAME
An alphanumeric string of up to 8 characters containing the name of the system to which the internal START command is to be routed. The name must reflect an existing system name. If not specified, the command is not routed to another system; in other words, the default is the local system.

This parameter is used by the primary MERVA ESA instance and is only valid if the ISCSTART parameter of the DSLPARM macro is set to *AUTO*.

TSSIZE
The size in bytes of the dynamic working storage for the nucleus server program. The default value is 4096.

DSLNTR: Defining the Task Server Request Table

The DSLNTR macro generates the MERVA ESA task server request table for the MERVA ESA central services (DSLNTRT). There is only one DSLNTRT in a MERVA ESA system, and it is used by DSLNUC only. The DSLNTRT lets you add programs executing central services without changing code in the existing MERVA ESA programs. DSLNTRT is generated by the DSLGEN macro.

A central service can run as a separate task if the appropriate entry is made in the nucleus server table DSLNSVT. When this is the case, the calls to this central service are made in an asynchronous way, and the processing of the other programs of the MERVA ESA DSLNUC is not impeded. When a program runs under direct control of DSLNUC and not as a separate task, all programs running under direct control of DSLNUC use its resources concurrently, especially the CPU time. Care must be taken that such a program does not keep control too long in order not to disturb the processing of the other programs. If such a program has to wait for an event, for example, the completion of an I/O operation, you should use the wait facilities of CICS so as not to give control to a region other than CICS, as this will decrease the performance of the other CICS transactions.

Name	Operator	Operands
[*label*]	**DSLNTR**	[**TYPE=**{**FINAL**|**INITIAL**|**PGM**}] [**,MF=D**] [**,NAME=***pgmname*]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The type of DSLNTRT table entry to be generated:

FINAL
Shows the end of the DSLNTRT and sets the pointer to the last entry of DSLNTS.

INITIAL
Shows the beginning of the DSLNTRT and generates the DSLNTRT header and the entries for the programs DSLJRNP (MERVA ESA journal service), DSLNCS (MERVA ESA operator command service), DSLNMOP (MERVA ESA operator message service), DSLNUSR (MERVA ESA user file services including signon and signoff), and DSLQMGT (MERVA ESA queue management service).

PGM Generates the entry for the program with the name indicated with the NAME parameter. Use this type if you want to add a central service to MERVA ESA.

MF
Generates the DSECTs of the DSLNTRT header and the DSLNTRT entry.

NAME
For TYPE=PGM only, the name of the program. This name must be a valid external reference for the linkage editor.

DSLOMS: Formatting Display Messages

The DSLOMS macro is used to do the following:

- Call the operator message program DSLOMSG
- Map the parameter list of DSLOMSG and generate a parameter or substitution list for DSLOMSG

Calling the Operator Message Program DSLOMSG

Use the DSLOMS macro to prepare a message display.

The caller provides a table of message texts, a message identifier, and an output buffer. You can specify current values for variables in the text in the DSLOMSG parameter list.

DSLOMSG selects the identified message from the table and replaces any text variables with values in the DSLOMSG parameter list. The resulting message is placed in the output buffer.

DSLOMSG is entered directly from the calling program.

Name	Operator	Operands
[*label*]	**DSLOMS**	**,MF=(E,**{addr|(*r*)})
		MSGID={'*aaannna*'|*addr*|(*r*)}
		[**,BUF=**{**(R0)**|*addr*|(*r*)}]
		[**,LAN=**{'*c*'|*addr*|(*r*)}]
		[**,PREFIX=**{**OMSP**|*cccc*}]
		[**,TABLE=**{**COMMSGTA**|*addr*|(*r*)}]

Programming Notes: Before using this macro, the caller should establish the current values of any text variables by placing the data in the fields generated by the positional parameters of the MF=L form of this macro.

For programs not linked to DSLNUC, you must ensure that DSLCOM is addressable and that the field COMOMSGA contains the entry address of the operator message program DSLOMSG.

If the TABLE parameter is not used, the field COMMSGTA must contain the address of the operator message table.

label
: A unique statement label according to assembler language conventions.

MF
: The format of the macro. The first parameter must be **E** for execute form. The second parameter specifies the label of the DSLOMSG parameter list, or a general register containing its address.

MSGID
: The message identification as a literal in single quotes, or the label of a field containing the message identification, or a general register containing the address of that field. The message identification field must be 7 characters long.

BUF
The label of the buffer or a general register containing its address. The default is general register 0. The buffer must have an 8-byte prefix where the first halfword contains the total length of the buffer.

LAN
A 1-character language identifier as a literal in single quotes, the label of a field containing the language identifier, or a general register containing the address of that field. If this parameter is omitted, the value from the previous call with the same parameter list is taken.

PREFIX
The prefix for the labels of the fields of the parameter list. The prefix must be 1 to 4 characters long and contain only characters that result in valid assembler language symbols. The default is OMSP.

TABLE
A field or general register containing the address of the operator message table. DSLOMSG searches this table to find the required message text. Symbolic names may be imbedded in the text, which are replaced at execution time by the positional parameters given in the DSLOMSG parameter list. (To define the table and message text, refer to "DSLMSG: Defining the Diagnostic and Error Message Table" on page 144.)

If this parameter is omitted, the table address is taken from the field COMMSGTA in the DSLCOM.

Mapping the Parameter and Substitution List

Name	Operator	Operands
[*label*]	**DSLOMS**	**MF=L** [**,DSECT=**{<u>**NO**</u>|**YES**}] [**,PREFIX=**{<u>**OMSP**</u>|*cccc*}] [,*list of positional parameters*]

Programming Notes: When coding the DSLOMS macro in a VSE environment, the positional parameters must precede the value parameters.

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

DSECT
How the parameter list is to be mapped:

NO Does not generate a DSECT statement. This is the default.

YES Generates a DSECT statement.

PREFIX
The prefix for the labels of the fields of the parameter list. The prefix must be 1 to 4 characters long and must contain only characters that result in valid assembler language symbols. The default value is OMSP.

list of positional parameters
A list of substitution items. Each positional parameter results in one

substitution item. The substitution items are positional, corresponding to the symbolic names (@0 to @23) used in the message text. The symbolic name represents an index to the list of substitution items.

Each occurrence of an @n variable in a message text is substituted at execution time by the related item (positional parameter) in the parameter list. @0 is the first positional parameter. The maximum number of positional parameters is 24.

You can code substitution characters in either of the following ways:

- As a literal value enclosed in single quotes
- (label, code, length)

label

The name of a field to be generated in the list of substitution items. It must be a valid assembler language symbol.

code

The length and characteristic of the field to be generated in the list of substitution items. A define storage (DS) instruction is generated as follows:

```
nC  character string of length n   (label   DS  CLn' ')
nX  hex string of length n         (label   DS  XLn'00')
B   1-byte binary number           (label   DS  X'0')
H   2-byte binary number           (label   DS  H'0')
F   4-byte binary number           (label   DS  F'0')
```

Binary numbers are converted to printable decimal numbers before they are put into the message buffer.

length

The length of the substitution text in the message buffer:

nn	Length of substitution text in the message buffer. For B, H, F, and X it must be a value from 1 to 16.
S	Actual length of the substitution item is used. For character strings trailing blanks are removed. For numeric values leading zeros are suppressed.
SP	For numeric fields only, if the actual value is zero, text is suppressed from the preceding blank until the end of the item.
SC	For numeric fields only, if the actual value is zero, text is suppressed from the preceding comma until the end of the item.

If the message text does not fit into the message buffer during execution time, it is truncated and a reason code is given in the parameter list.

DSLPARM: Generating the DSLPRM Module

The DSLPARM macro has two functions:

- Generate the customizing parameter table
- Map the customizing parameters

Generating the Customizing Parameter Table

The DSLPARM macro generates a CSECT named DSLPRM containing the MERVA ESA customizing parameters.

The MERVA ESA tables specified in this macro are individual modules, not linked to DSLPRM.

Name	Operator	Operands
	DSLPARM	[,APISMSG={4096|*nnnn*}]
		[,APIUID={MASAPI|*cccccccc*}]
		[,CID={DSL|*ccc*}]
		[,CINTER={NO|YES}]
		[,CLIENTS={0|*nnnn*}]
		[,CURCODE={TABLE|FILE }] {(TABLE,FILE)} {(FILE,TABLE)}
		[,CVTEXTO={0|*nnn*}]
		[,CWAOFF={20|*nnnn*}]
		[,DBCS={NO|YES}]
		[,DB2PLB={DSLNUC|*cccccccc*}]
		[,DB2SS={DB2|*cccc*}]
		[,DM={90|*nnn*}]
		[,DSLID={DSL|*cccc*}]
		[,EUDTRAN={DSLE|*cccccccc*}]
		[,EXAFO={NO|YES}]
		[,EXDSP={YES|NO}]
		[,EXJRN={NO|YES}]
		[,EXLOAD={CONT|STOP}]
		[,EXQUE={NO|YES|MASTER}]
		[,EXSEC={NO|YES|(YES,NOCHECK)}]
		[,EXUID={NO|YES}]
		[,EXUMASK={NO|YES}]
		[,EXUSR={(YES,ORIGINID)}] {(YES,GROUPID) } {NO }
		[,FDT={DSLFDTT|*cccccccc*}]
		[,FLT={DSLFLTT|*cccccccc*}]
		[,FLVSTOR={4096|*nnnnn*}]
		[,FNT={DSLFNTT|*cccccccc*}]
		[,IRQNO={20|*nnn*}]
		[,ISCMQID={(*prim*,*local*[,*fix*])}]
		[,ISCMRCV={*rcvq*}]

Name	Operator	Operands
		[,ISCMRTQ={*rtq*}]
		[,ISCMSND={*sndq*}]
		[,ISCNUC={PRIMARY|SECONDARY}]
		[,ISCSTART={MANUAL|AUTO}]
		[,ISCXCF={(*xcfgrp*[,*xcfmem*])}]
		[,ISCXJWT={30000|*nnnnnn*}]
		[,ITC=({INTRA|*ccccc*},{INTER|*ccccc*})]
		[,ITCAREQ={(*c1*,*c2*,*c3*,*c4*)}]
		[,ITCASRV={(*c1*,*c2*,*c3*,*c4*,*c5*,*c6*)}]
		[,ITCMRCV={*rcvq*}]
		[,ITCMRTQ={(*rtq*[,*rtqdyn*])}]
		[,ITCMSND={*sndq*}]
		[,ITCMWTT={30000|*nnnnnn*}]
		[,ITCQSRV={*ccccc*}]

		[,JRNBUF={(*nnnnn* [,YYYY])}]
		{(*nnnnn* [,SEG]) }
		{ 16000 }

		[,JSWITCH={ONCE|MANUAL|CYCLE}]
		[,LAN={(*l*[,c1 [,c2]])}] {(E[,','[,'.']])}
		[,LRGMSG={NO|(YES,*nnnnn*)}]
		[,MAXBUF={*nnnnn*}]
		[,MCBNUM={10|*nnn*}]
		[,MERVUSR={DSLUSER|*cccccccc*}]
		[,MFSSTOR={({6144|*nnnnn*},{32000|*nnnnn*},{1536|*nnnnn*})}]
		[,MQMNAME={*mqmname*}]
		[,MSG={DSLMSGT|*cccccccc*}]
		[,MSGLIM={65535|*nnnnn*}]
		[,MTT={DSLMTTT|*cccccccc*}]
		[,MVSSS={NO|YES}]
		[,NAME={MERVAESA|*cccccccc*}]
		[,NICBUF={16384|*nnnnn*}]
		[,NICPL={1024|*nnnn*}]
		[,NQE={2000|*nnnnnn*}]
		[,OPID={MAS|*ccc*}]
		[,OPINT={YES|NO}]
		[,PGBUF={1024|*nnnn*}]
		[,PGCALL={NO|YES}]
		[,PRTNAME={*cccccccc*}]
		[,QDS={1|(2,STOP)|(2,CONT)}]
		[,QIO={DSLQMGIO|(DSLQMCNV,*cccc*)|DSLQMDIO}]
		[,QTRACE={OFF|SMALL|LARGE}]
		[,RACFSVC={*nnn*}]
		[,RECON={NO|(YES,STOP)|(YES,CONT)}]

Name	Operator	Operands
		[,RTLOAD={CONT|STOP}]
		[,RTRACE={OFF|ALL|WARNING|SEVERE}]
		[,SDDB2={NO|YES}]
		[,SDRC={NO|YES}]
		[,SONNUM={0|*nn*}]
		[,SVC={*nnn*}]
		[,TFD={DSLTFDT|*cccccccc*}]
		[,TIER={0|*nnnnnn*}]
		[,TOFSIZE={({18432|*nnnnn*},{0|*nnnnn*})}]
		[,TRACE={INT|EXT|OFF}]
		[,TSAUX={YES|NO}]
		[,UCTRAN={NO|YES}]
		[,UMR={NO|YES|(YES,IMM)}]
		[,USER={20|*nnnnn*}]
		[,USERSTO={({0|*nnnn*},{208|*nnnnn*})}]
		[,USFPW={YES|NO}]

		[,USGRP={NO }]
		 {(YES,OPT) }
		 {(YES,OPTIONAL)}
		 {(YES,REQ) }
		 {(YES,REQUIRED)}

		[,WSASRV={(c1,c2,c3,c4)}]
		[,WSSEC={AUTHENTICATION|ENCRYPTION}]
		[,WSTSRV={7117|*nnnnn*}]

Programming Notes: There is no default for the SVC parameter. Under MVS if the SVC parameter is omitted, interregion communication is not available. Under VSE the SVC parameter is ignored.

The tables specified in this macro are individual load modules that are loaded by the MERVA ESA programs, and their addresses are stored in DSLCOM. The TFD table is an exception. Programs that require the terminal feature definitions table must load the table module and move the required values to the TUCB.

APISMSG

The size of the message buffer (MSGSWIFT buffer) used by the MERVA ESA application programming interface DSLAPI. The default is 4096. This value is used for compatibility with application programs created for MERVA/ESA V3.1 or earlier.

For MERVA ESA V3.2 and later releases the copy books for the MSGSWIFT buffer used by MERVA ESA application programs already define a buffer size of 12288. The size has been increased to allow for large SWIFT messages of up to 10000 bytes. When all your applications have been recompiled using the new copy books, you should increase the APISMSG parameter to 12288.

It is recommended for all new application programs to use the API calls MSGG and MSGP to format messages. For these API calls, the APISMSG parameter is not used. Refer to the section about customizing DSLAPI in the *MERVA for ESA Application Programming Interface Guide*.

APIUID
The user identification used by the DSLAPI command and user-file functions, CMD, USRG, and USRN. This identification may be used for an authorization check in the command execution exit DSLNCU01, and is located in the MERVA ESA journal record for the executed command. The default is MASAPI.

Note: If you want this identification to get the authorization for the restricted operator commands, it must start with the same three characters as specified in parameter OPID. This also applies when OPID=/// is specified.

CID
For CICS only, a 3-character operator identifier used for CICS signon. This user is allowed to start MERVA ESA and to enter MERVA ESA commands at the VSE console. The operator identifier is defined in the CICS signon table with the OPIDENT parameter of the macro DFHSNT. The default is DSL.

If CID=*** is specified, the operator identifier is not checked against the CICS signon table and MERVA ESA is started.

CINTER
For CICS only, specifies that interregion communication is tried (YES) if intraregion communication to MERVA ESA in the local CICS region was not possible. NO is used by default.

CLIENTS
The number of MERVA Message Processing Client licenses you have acquired for this installation.

The default value is 0, that means no clients are allowed to sign on to MERVA ESA.

CURCODE
Whether the currency code checking routine should use the internal currency code table (TABLE) or use the currency code file (FILE). TABLE is used by default. If (FILE,TABLE) or (TABLE,FILE) is specified, the currency code file (FILE) is used first and then the internal currency code table (TABLE).

If FILE is specified, the currency code file must be installed and defined in the MERVA ESA file table DSLFLTT.

For batch programs, an appropriate DD statement must be added. For IMS transaction programs, the PSBGEN and ACBGEN must be executed to allow access to the database.

CVTEXTO
For MVS only, a decimal offset into the CVTUSER extension table. The offset must be a number from 0 to 156, and a multiple of 4. The default is 0. At the specified offset MERVA ESA establishes a pointer to its interregion communication area (DSLICA), used by the program DSLNICP.

When more than one MERVA ESA exists in the environment, each one must have a unique offset in the CVTUSER extension. For more information about using the CVTUSER field refer to MVS requirements in the *MERVA for ESA Installation Guide*.

CWAOFF
For CICS only, a decimal offset to the MERVA ESA part of the Common System Work Area (DFHCWA).

MERVA ESA uses 112 bytes of CWA storage. The CWAOFF parameter can be used to coordinate CWA usage with other CICS transactions. The default is 20. In IMS systems, the parameter is ignored.

The specified offset must be compatible with the CICS parameter WRKAREA of the macro DFHSIT. CWAOFF cannot be greater than WRKAREA minus 112. The maximum value for WRKAREA is 3584, so the maximum value for CWAOFF is 3472.

DBCS
Whether double-byte character set (DBCS) is supported. The default is NO.

DB2PLB
The DB2 plan name for MERVA ESA applications running in a batch region when queue management using DB2 has been specified and direct DB2 (parameter SDDB2) is in use. The default is DSLNUC.

DB2SS
The DB2 subsystem name for MERVA ESA applications running in a batch region when queue management using DB2 has been specified and direct DB2 (parameter SDDB2) is in use. The default is DB2.

DM
The number of unsolicited MERVA ESA operator messages that can be stored at one time. These messages can be displayed with the **dm** operator command. The value can be from 90 to 380. The default is 90.

DSLID
A MERVA ESA identifier of up to 4 characters. It is displayed in all MERVA ESA system console messages. The default is DSL.

Under VSE, this identifier is used to coordinate interregion communication. The first 3 characters of DSLID identify the MERVA ESA with which an interregion communication is to be established. When more than one MERVA ESA exists in the environment, each one must have a unique DSLID identifier. Programs that communicate with a MERVA ESA in another partition must load the same DSLPRM module as the target MERVA ESA to use the same MERVA ESA identifier.

For MVS only, if the parameter MVSSS is specified with the value YES, the 4-character identifier is used to identify the SSCT to be used by DSLNICP (SVC).

The MERVA ESA identifier can also be displayed on a user screen or printer output by specifying *DSLID* as a TOF field name within the screen or printer section of an MCB (see macro DSLLDFLD).

EUDTRAN
A transaction code of up to 8 characters to be assigned to the MERVA ESA end user driver program (DSLEUD). The code must also be defined in a CICS or IMS TRANSACTION definition. The default value is DSLE.

EXAFO
Whether the signon request of an end user is accepted by MERVA ESA when the end user is already signed on. The default is NO. If a user treis to sign on when there is already an active session for that user and:

- EXAFO=YES, the previous session is terminated and a new session is started. This is equivalent to an automatic force of a session, but without operator intervention. For example, when an end user looses his session because his terminal has been disconnected, this parameter provides an easy way of reconnecting to MERVA ESA.

If the already active session is on a different terminal, this session is only terminated if function selection is entered or a user file request is done. In this case the following message is issued:

```
DSL1014 You have been signed off
```

- If EXAFO=NO, the following error message is issued:

  ```
  DSL1025 User identification is already signed on
  ```

EXDSP

Whether MERVA ESA user file records can be accessed directly by application programs. The default is YES. Only the display and the list functions are allowed; adding, changing, or deleting user file records from an application program is not allowed. With the exception of the password fields, which are always empty, the entire user file record is passed to the application program. EXDSP=YES must be specified when the DSLAPI functions USRG or USRN are to be used. Using these functions, an application program can inspect the MERVA ESA user file records and produce a report containing all defined users and their authorizations. The security aspects of this should be carefully considered.

EXJRN

Whether the MERVA ESA journal display command is allowed in the CMD function. The default is NO. This command supports the installation test (refer to the *MERVA for ESA Installation Guide* for more information). If EXJRN=YES is specified, the journal display command **jrn** is allowed in your installation.

Note: For security reasons, YES should be specified for testing purposes only, because an operator can use this command to display the contents of messages transferred directly through a network link.

Specify EXJRN=NO to remove the **jrn** command from the system.

EXLOAD

Whether MERVA ESA continues processing when the loading of a user exit fails:

CONT	MERVA ESA continues processing. MERVA ESA issues the message DSL048I for each module that cannot be loaded. This is the default.
STOP	MERVA ESA stops processing.

EXQUE

Whether the MERVA ESA queue test commands **copy**, **delete**, **delx**, **free**, and **move** are allowed in the CMD and MSC function. The queue test commands should be used for testing purposes only, because with these commands the normal message flow which is defined by the function table and the routing tables, can be bypassed. The influence on security considerations has to be evaluated carefully. The purpose of this component is to support an installation during testing (refer to the *MERVA for ESA Installation Guide* for more information).

NO	The queue test commands are not allowed. This is the default.
YES	The queue test commands are allowed for all users.
MASTER	The queue test commands are allowed for master users only.

EXSEC
Whether MERVA ESA checks the password:

NO MERVA ESA checks the password. This is the default.

YES MERVA ESA does not check the password. The password must be checked using any external security feature.

When the password is checked externally, the association of the password to the appropriate MERVA ESA user identifier is lost. As a result, **any** user identifier defined in MERVA ESA could be used to sign on to MERVA ESA from a CICS or IMS terminal. Therefore, MERVA ESA allows the signon from a terminal only if two other parameters are specified as follows:

- EXUID=YES
- PGCALL=YES

In other words: MERVA ESA checks the signon from a terminal. It accepts the signon only when an external security manager like RACF is used (refer to the *MERVA for ESA Customization Guide* for more information).

With PGCALL=YES user-written programs can call the MERVA ESA End User Driver program (DSLEUD). Under IMS, MERVA ESA checks whether the SPA contains one character at least that is not equal to X'00'. Therefore, user-written programs must provide a SPA that does not entirely consist of X'00' (refer to the *MERVA for ESA Customization Guide* where the rules for the program-to-MERVA switch are explained).

The subparameter NOCHECK specifies that MERVA ESA does not check the signon from a terminal nor the SPA under IMS. Thus the following can occur:

- Any user identifier defined in the MERVA ESA user file is accepted for the signon from a terminal.
- The transaction code associated to the DSLEUD can be entered together with the user identifier, optionally followed by the name of a MERVA ESA function. For the entered user identifier, MERVA ESA then displays the Function Selection panel or the first panel of the entered function.

EXUID
How to sign on to MERVA ESA:

NO The MERVA ESA signon panel is displayed to enter the required user identification, provided that parameter EXSEC=YES is not specified. This is the default.

If the MERVA ESA user file is not empty, a user record must exist for the entered user identification.

YES The MERVA ESA signon is bypassed for an external user identification defined to the ESM. The external user identification must have signed on to CICS or IMS. The parameters EXSEC=YES and PGCALL=YES must also be specified.

If the MERVA ESA user file is not empty, a user record must exist for the external user identification.

EXUMASK

Whether additional data masking of the user file records is performed:

NO Additional data masking of the user file records is not performed. Application programs performing a direct access to the user file via VSAM and using their own procedure for unscrambling user file records will not work unless EXUMASK=NO is specified. In this case the user file records are scrambled using the same algorithm as used for MERVA/ESA V3.1 and earlier. This is the default.

YES Additional data masking of the user file records is performed each time an existing record is changed or a new record is added. The enhanced scrambling algorithm offers more protection against unauthorized access to the user file. Application programs using the MERVA ESA API functions for accessing user file records continue to work unchanged.

EXUSR

Which user records a user can maintain during online user file processing:

YES,ORIGINID

During online user file processing:

- A general user can only maintain user records of general users with the same first 8 characters in the Origin ID field as that user's identification
- An authorized user that is not a master user can maintain user records of:
  - General users that have the same first 8 characters in the Origin ID field as that user's identification
  - Authorized users that have the same user type and the same first 8 characters in the Origin ID field as that user's identification
- A master user can maintain the user records of all users

This is the default. The second subparameter can be specified as ORIGIN or ORIGINID.

YES,GROUPID

During online user file processing:

- A general user can only maintain user records of general users that have the same group ID as that user's identification
- An authorized user that is not a master user can maintain user records of:
  - General users that have the same group ID as that user's identification
  - Authorized users that have the same user type and the same group ID as that user's identification
- A master user can maintain the user records of all users

With EXUSR=(YES,GROUPID) also USGRP=(YES,··) must be specified.

The second subparameter can be specified as GROUP or GROUPID.

NO No restriction applies.

For an explanation of how to specify a user as general user, authorized user, or master user, see the *MERVA for ESA User's Guide*.

FDT
The name of the MERVA ESA Field Definition Table to be used. The default is DSLFDTT.

FLT
The name of the MERVA ESA file table to be used. The default is DSLFLTT.

FLVSTOR
The size of temporary storage for the MERVA ESA file service program DSLFLVP. You can specify a value from 0 to 32760. The default is 4096.

FNT
The name of the MERVA ESA function table to be used. The default is DSLFNTT.

IRQNO
The number of request queue elements allocated for the asynchronous nucleus server processing. IRQNO must be large enough to accomodate all asynchronous server requests at any point in time. You can specify a value from 2 to 100000. The default is 20.

ISCMQID
This parameter is used for interservice communication via MQSeries.

If specified, the first (*prim*) and the second (*local*) operand must be specified. Otherwise the interservice communication will not be established. These two operands specify the MERVA IDs used as queue name parts and how they are to be used when MQI send queue names are constructed.

prim
The MERVA ID of the primary MERVA ESA instance. It is used by all MERVA ESA instances during initialization. The MERVA ID must match the value assigned to the QNAME parameter of the nucleus server table entries for the primary MERVA ESA instance.

local
The MERVA ID of the local MERVA ESA instance. If the local MERVA ESA instance is primary, *prim* and *local* must be equal. This parameter value is used by the local MERVA ESA instance to decide whether a service is to be routed to another MERVA ESA instance.

If the local MERVA ID matches the QNAME value of a certain service in the nucleus server table, a local MERVA ESA instance assumes the service in its own environment. Otherwise, the requested service is routed to the MERVA ESA instance identified by the constructed MQI send queue name.

fix Whether the value specified in the QNAME parameter of the nucleus server table should precede (PREFIX) or follow (SUFFIX) the base name specified in the ISCMSND parameter. The prefix or suffix is separated from the base name by a period. The default is SUFFIX.

For interservice communication, the related parameters ISCMRCV, ISCMRTQ, and ISCMSND must also be specified.

ISCMRCV
The name of the local MQI receive queue for interservice communication via MQSeries. It is used by the MERVA ESA MQSeries nucleus server DSLNMQS.

rcvq
The name of the predefined static local MQI receive or send/receive queue. It must be an alphanumeric string of up to 48 characters according to the MQSeries object naming conventions.

ISCMRTQ

The name of the local MQI reply-to queue for interservice communication via MQSeries. It is used by the MERVA ESA MQSeries nucleus server DSLNMQS.

rtq

The name of the predefined static local MQI reply-to queue. It must be an alphanumeric string of up to 48 characters according to the MQSeries object naming conventions.

ISCMSND

The names of the locally defined remote MQI send queue or the name of a local MQI send/receive queue for interservice communication via MQSeries. It is used by the MERVA ESA MQSeries nucleus server DSLNMQS.

sndq

The base name for the local MQI send or send/receive queue. The final message queue name is constructed by combining it with the QNAME value of the selected nucleus server table entry. Depending on whether PREFIX or SUFFIX is specified in the third ISCMQID parameter value, QNAME precedes or follows the specified base name, separated by a period. Thus, the base name must not exceed 46 characters. The assembled queue name must match the name assigned during message queue generation and must follow the MQSeries object naming conventions.

ISCNUC

You can specify the following two values:

PRIMARY	For the interservice communication via MQSeries, the local MERVA ESA instance acts as the primary instance in a multi-system environment of distributed multiple MERVA ESA instances. The primary MERVA ESA instance serves also as a focal point for all MERVA ESA requesters and forwards requests to the secondary MERVA ESA instance defined to service it. This is the default.
SECONDARY	The local MERVA ESA instance acts as a secondary in a multi-system environment of multiple MERVA ESA instances. The secondary MERVA ESA instance services requests forwarded by the primary MERVA ESA instance.

ISCSTART

For the interservice communication via MQSeries, the method by which secondary MERVA ESA instances are started:

MANUAL	All secondary MERVA ESA instances are started manually. This is the default.
AUTO	Specify this operand only if you have ensured that all secondary MERVA ESA instances are defined as started tasks on the system on which they are to be started. The primary MERVA ESA instance starts all secondary MERVA ESA instances defined in the nucleus server table via the internal address space start facility. This value is only valid if the **ISCNUC** operand is *PRIMARY*.

ISCXCF

If the systems MERVA ESA runs on are connected by XCF services, you can specify that MERVA ESA should use the provided capabilities. By specifying this parameter you request remote failure notification to the primary MERVA ESA instance.

You can specify the following two subparameters:

xcfgrp
The XCF group name for MERVA ESA. This value is required.

xcfmem
The XCF group member name of this MERVA ESA instance. The name must be unique for each MERVA ESA instance within the XCF group for MERVA ESA. If not specified, the local MQM name, as specified in the MQMNAME parameter, is substituted.

ISCXJWT
A numeric value from 100 to 999999, specifying the interval in hundredths of a second that the primary MERVA ESA instance waits for secondary MERVA ESA instances to join the MERVA ESA XCF group. Secondary MERVA ESA instances which do not join within this time will not participate in XCF signalling. The default is 300 hundredths of a second.

ITC
The type of MERVA ESA intertask communication to be used.

For CICS only, the first subparameter specifies the method of intertask communication for transactions running in a CICS region:

INTRA	The method of moving storage areas from the requester task to the MERVA ESA nucleus server. This is the default for requesters running in a CICS region. The CICS storage protection facilities cannot be used in this case. An improved intraregion intertask communication method is used. This uses only one single ECB for any number of intertask communication sessions.
INTRAOLD	The method of moving storage areas from the requester task to the MERVA ESA nucleus server used in MERVA ESA V3.3. If there are user-written application programs in assembler language which directly refer to the internal MERVA ESA control blocks (ICBs) and do not use the standard MERVA ESA interface via the DSLNIC Macro or MERVA ESA API interface, the specification INTRAOLD should be made. In this case the ECB within each ICB is individually posted.
TSQ	Uses CICS temporary storage queues to move the storage areas from the requester to the MERVA ESA nucleus server. The related parameter ITCQSRV must also be specified.

The second subparameter specifies the method of intertask communication for programs and transactions running in a different region from the MERVA ESA nucleus:

INTER	The standard method using the MERVA ESA SVC under MVS or XPCC under VSE. This is the default for requesters running in a batch region.
APPC	For MVS only, APPC/MVS services are used to move the buffer areas from the requester to the MERVA ESA nucleus server. The related parameters ITCASRV and ITCAREQ must be specified in this case. This requires the intertask communication server for APPC communication being defined in DSLNPT.

MQI	For MVS only, MQI services are used to move the buffer areas from the requester to the MERVA ESA nucleus server. The related parameters ITCMRCV, ITCMRTQ, and ITCMSND must be specified in this case. It requires also the intertask communication server for MQI communication being defined in DSLNPT.
NONE	Interregion communication is disabled; the MERVA ESA SVC need not to be installed.

ITCAREQ

The parameters for the requestor of a MERVA ESA intertask communication using APPC. The following subparameters can be specified:

- A symbolic destination name representing the partner LU, the partner TP name, and the mode name for the session on which the conversation is to be carried. The symbolic destination name must match that of an entry in the side information data set. The appropriate entry in the side information is retrieved and used to initialize the characteristics for the conversation. This subparameter is optional when the second and third subparameters are specified.
- The partner LU name specifies the name of the LU at which the partner program, the MERVA ESA intertask communication server for APPC is located. MERVA ESA must have registered as a server with APPC/MVS for this LU. The partner LU can be an LU name only (1- to 8-byte character string), or combined network_ID and network LU name (two 1- to 8-byte character strings, concatenated by a period).

  When the LU name is specified, the first subparameter (symbolic destination name) is ignored. This subparameter must be identical to the second subparameter of ITCASRV specified in the customization module DSLPRM on the MERVA ESA server side.
- The MERVA ESA transaction program name used by the requestor. This name must match the value used by the server side of the intertask communication. The server side uses the third subparameter of the ITCASRV parameter in the DSLPRM definition to register with APPC/MVS as a server. The name can be up to 64 characters. If the transaction program name is omitted the third subparameter of ITCASRV is used.
- The mode name designating the network properties for the logical connection to be allocated for the conversation. This value overwrites the current values aquired from the side information using the symbolic destination name. If needed, specify a valid mode name for an APPC connection, for example APPCLU62.

ITCASRV

The parameters for the server side of the MERVA ESA intertask communication using APPC; the server program DSLNTSA is link-edited to DSLNUC. Six subparameters can be specified:

1. A symbolic destination name that represents the transaction program and local LU for which MERVA ESA will register. This parameter, if used, must contain a value that matches a symbolic destination name defined in the active side information data set. APPC/MVS obtains the TP name and local LU name from the side information data set. This subparameter is optional when the following subparameters are specified.
2. The local LU name where the MERVA ESA nucleus server resides; this must be a NOSCHED LU. A NOSCHED LU is an LU that is not associated with a transaction scheduler. Like other LUs, NOSCHED LUs are defined through the LUADD statement in the APPCPMxx member of

SYS1.PARMLIB. For more information about defining NOSCHED LUs, see *MVS/ESA SP V5 Initialization and Tuning Reference*. The length of the local LU name is up to 8 bytes. When the LU name is specified, the first subparameter (symbolic destination name) is ignored.

3. The MERVA ESA transaction program name. The name can be up to 64 characters. This name should be unique for a given MERVA ESA installation using the same local LU name. The same transaction program name is used for the server and for the requestor.
4. The partner LU name. The partner LU can be an LU name only (1- to 8-byte character string), or combined network_ID and network LU name (two 1- to 8-byte character strings, concatenated by a period). This subparameter is optional and can be used to restrict the access to MERVA ESA. If the partner LU name is specified, only requestors from this specific LU are accepted.
5. The user ID of the partner. This subparameter is optional and can be used to restrict the access to MERVA ESA. If the user ID of the partner is specified, only requestors associated with this specific user ID are accepted.
6. The security profile of the partner. This subparameter is optional and can be used to restrict the access to MERVA ESA. If the security profile of the partner is specified, only requestors with this specific security profile are accepted. APPC/MVS treats the profile name as a RACF group name.

ITCMRCV

The name of the local MQI receive queue for intertask communication via MQSeries at the responding side, the MERVA ESA nucleus. It is used by the MERVA ESA MQSeries nucleus server DSLNTSM.

rcvq

The name of a predefined static local MQI receive or send/receive queue. It must be an alphanumeric string of up to 48 characters according to the MQSeries object naming conventions. However, if the PARM operand of the DSLNPT definition for the MERVA ESA nucleus task server for MQSeries has a value, then this value separated by a period is appended to the specified name when the final queue name is constructed. In this case the maximum queue name specification is limited to 46 characters.

ITCMRTQ

The names of the local MQI reply-to queue for intertask communication via MQSeries. These names are used at the requester side by end-user transactions, batch jobs, and application programs.

rtq

The name of the predefined static local MQI reply-to queue.

This can be the predefined static local MQI reply-to queue name or a predefined MQI model queue defined for receiving. It must match the name assigned during the message queue generation process.

If the name specifies the name of an MQI model queue, the MQSeries queue manager (MQM) creates a dynamic MQI queue with the attributes of the model queue and with the name as specified in *rtqdyn*. The model queue should have the default definition type DEFTYPE (TEMPDYN).

rtqdyn

The name you want to give to the dynamic reply-to queue as created by the local MQM, if *rtq* specifies a model queue.

If the last character is an asterisk (*), MQM replaces the asterisk with a string of characters that guarantees that the generated queue name is unique at the local MQM. To allow a sufficient number of characters for this, the asterisk is valid only in positions 2 through 33.

ITCMSND

The name of a locally defined remote MQI send queue or the name of a local MQI send/receive queue for intertask communication via MQSeries. The name is used at the requester side by end-user transactions, batch jobs, and application programs.

sndq

The name of a predefined static local MQI send or send/receive queue. It must be an alphanumeric string of up to 48 characters according to the MQSeries object naming conventions.

ITCMWTT

A numeric value from 100 to 999999, specifying the interval in milliseconds that DSLNICT waits for a response message on its MQI reply-to queue. If no message arrives within this time, the call completes with a message showing that there was no message that matched the selection criteria on the queue. The default is 30000 milliseconds.

ITCQSRV

The parameter for the server side of the MERVA ESA intertask communication using CICS temporary storage queues; the server program DSLNTSQ is link-edited to DSLNUC. The parameter is a five-character queue name prefix used for the CICS temporary storage queues. This prefix should not be used for any other CICS temporary storage queues.

JRNBUF

The size of a MERVA ESA journal record. You can specify a value from 1024 to 2097152 (2MB). The default is 16000. The actual size used is the logical record length of the records defined in the cluster allocation job. When the MERVA ESA size is larger than the logical record length of the records in the cluster a warning message is issued. The actual size can be larger than 32KB when spanned VSAM records are used. In this case the size limit is determined by VSAM. The subparameter SEG specifies that the records may be segmented. In this case MERVA ESA generates unique keys for each segment by adding a segment number to the record key. The subparameter YYYY specifies that a 4-digit year is used in the journal key. A setting of YYYY also includes a setting of SEG.

You are strongly recommended to specify the subparameter YYYY. Otherwise, MERVA ESA is not year 2000 enabled.

Note: This buffer size also determines the length of the storage used for the MERVA ESA trace facilities. The size used for trace buffers is limited to 32760 bytes. See QTRACE (page 193), RTRACE (page 194), and TRACE (page 196) parameters.

JSWITCH

The initial state of the journal data set switching mode. When MERVA ESA is running, the initial switching mode can be modified by a master operator using the JSET command.

ONCE	The switch to data set B is done automatically, any further switches have to be done manually. This is the method of MERVA ESA V3 and should be used if compatibility is needed. This is the default.
MANUAL	For any switch an operator command is required. Thus the

switching is under full operator control. This method is useful when backup procedures are running and an automatic switch to a data set would interfere with the backup procedure.

CYCLE A switch is done automatically from data set A to B and later on back to A and so forth. This is the method for continuous (7x24) operation. Records cannot be deleted or overwritten automatically when this method is used. The cleanup of the journal data set has to be done using a separate batch job or by manual processing. The customer's backup procedures must ensure that the journal data set is emptied after the backup operation.

When a switch is performed and the alternate journal is not available (or full), the journalling stops and MERVA ESA usually terminates.

LAN

The language for the users signon panel and country specific characters for command display output:

- The first subparameter is a character identifying the language for the signon panel; this character must be specified in the signon panel DSL0SON in a DSLLDEV TYPE=SCREEN macro in the ID parameter. The default is E.
- The second subparameter defines the character used as decimal separator for some command responses, for example DIVA and DLMC. The character must be enclosed in quotes. The default is a comma (',').
- The third subparameter defines the character used as thousand separator for some command responses, for example DIVA and DLMC. The character must be enclosed in single quotes. The default is a period ('.').

LRGMSG

The handling of messages depending on their size when they are stored in a queue.

NO

Indicates to DSLQMGT that large messages are not supported in this MERVA ESA installation, that is, the maximum length of a message that can be stored in the MERVA ESA queue data set (QDS) is 31900 characters. This is the default.

YES

Indicates to DSLQMGT that large messages are supported in this MERVA ESA installation, that is, the maximum length of a message that can be stored is 2097152 characters (2MB, see MAXBUF parameter).

The length value *nnnnn* specifies, in the range from 300 to 31900, the limit on which DSLQMGT decides to store the message in the queue data set (QDS) or in the large message cluster (LMC). The default is 31900.

Note: When QIO=DSLQMDIO (queue management using DB2) is specified, this parameter is meaningless. With DB2 large messages are always supported.

MAXBUF

The maximum size of a MERVA ESA buffer. You can specify a value from 1024 to 2097152 (2MB). The size specified for MAXBUF is the upper limit for dynamic buffers and dynamic TOFs used in MERVA ESA. The default is the value specified for NICBUF.

MCBNUM

The number of MCBs to be loaded concurrently. A value from 1 to 255 can be specified. This value is used by MFS in every MERVA ESA application to allocate space for the MCB and exit program load table. MCBNUM+2 modules are held concurrently in storage.

MCBNUM should be specified in the range from 6 to 18, depending on the available storage. The default is 10.

MERVUSR

The identifier of a pseudo user to be used when the MERVA ESA user file is empty. The default is DSLUSER.

The pseudo user identifier must be defined to the ESM if three other parameters of DSLPARM are specified as follows:

- EXSEC=YES
- EXUID=YES
- PGCALL=YES

In this case, the MERVA ESA signon is bypassed and the pseudo user can sign on to MERVA ESA provided that it has already signed on to CICS or IMS.

If NO was specified for any of the three parameters above, the pseudo user identifier has to be entered at the MERVA ESA signon panel. EXUID=NO or PGCALL=NO also require that EXSEC=YES is not specified. The entered pseudo user identifier need not be defined to the ESM and also need not have signed on to CICS or IMS. For EXSEC=NO, the entered password must be equal to the entered pseudo user identifier.

MFSSTOR

The sizes of MFS storage areas. The first subparameter specifies the size of the MFS permanent storage. You must specify a number from 4096 to 32760. The default is (6144,32000,1536).

The second subparameter specifies the size of the MFS temporary storage. During online processing, if user exits require more working storage than the basic MFS temporary storage, this parameter should be increased. Otherwise, when the temporary storage pool is exhausted, MFS will issue individual GETMAINs for each additional storage area. This may degrade the system performance. You must specify a number from 2048 to 32760.

The third subparameter specifies the size of the retype buffer provided for the retype verification of a message. For each field to be retyped, 19 bytes plus the maximum length of the data area are used. You must specify a number from 8 to 32760.

MQMNAME

An alphanumeric string of up to 48 characters containing the name of the local MQSeries queue manager. This name must match the name assigned by the system administrator.

- For a CICS application, this operand is ignored because it will automatically be connected to the queue manager to which CICS is connected.
- For an IMS application, only names listed in the subsystem definition table (CSQQDEFV) and listed in the SSM table in IMS are connectable.
- For MVS batch and TSO, only queue managers that reside on the same system as the application which invokes intertask communication or the MERVA ESA nucleus which invokes interservice communication are connectable.

This operand is used if the second value of the **ITC** operand is *MQI* or if the **ISCMQID** parameter is specified. If it is not specified, MQSeries uses the default queue manager name defined for the local system.

Note: It is up to the installation if MQSeries provides a default queue manager name.

MSG
The name of the MERVA ESA operator message table. The default is DSLMSGT.

MSGLIM
For IMS only, specifies the number of messages a MERVA ESA transaction program can process in a single scheduling. This value is used by the MERVA ESA checking and expansion program DSLCXT, the MERVA ESA hardcopy printing program DSLHCP, and the MERVA-MQI Attachment programs DSLKQR and DSLKQS. After the transaction program has processed the specified number of messages, it inserts its TUCB into the IMS message queue, unless the MERVA ESA queue is empty. Then the program terminates and is available for rescheduling by IMS. This feature makes it possible for IMS to allow scheduling of higher priority transactions that may have entered the system while the MERVA ESA transaction programs were in process.

A number between 1 and 65535 can be specified. Code the MSGLIM value at 65535 if no limit is to be placed upon the number of messages processed at a single scheduling. The default is 65535.

Note: The MSGLIM parameter description is obsolete for those functions which have their own MSGLIM parameter specified in the function table.

MTT
The name of the MERVA ESA message type table. The default is DSLMTTT.

MVSSS
For MVS only, whether MERVA ESA has been defined as an MVS subsystem. Although MERVA ESA will not be run as an MVS subsystem, it will use the SSCT (subsystem control table) defined by MVS.

- If MVSSS=NO, DSLNICP (SVC) uses the CVT extension with an offset as specified in the CVTEXTO parameter, as a pointer to the DSLICA. This is the default.
- If MVSSS=YES, DSLNICP (SVC) uses the SSCT with the 4 characters specified in the DSLID parameter to identify the DSLICA.

NAME
The name used to identify the MERVA ESA installation. When two MERVA ESA systems are communicating, this name must differ from that used on the other system. This name, of between 1 and 8 characters, is used to compose the unique message references of the UMR option. The default is MERVAESA.

NICBUF
The length of the data area used as a service data buffer in DSLNIC TYPE=REQ parameter BUF. A value from 4096 to 32640 can be specified. It must be a multiple of 128. If it is not, it is rounded up to the next multiple of 128. It is not recommended to use a value smaller than 8064. The default is 16384.

Note: The value specified in this parameter is used by many MERVA ESA programs to determine their buffer storage requirements.

NICPL
The length of the data area used as a service parameter list in DSLNIC TYPE=REQ parameter PL. A value from 1024 to 32640 can be specified. It must be a multiple of 128. If it is not, it is rounded up to the next multiple of 128. The default is 1024.

NQE
The number of queue elements to be stored in the queue data set. A value from 1 to 99999999 can be specified. The default is 2000.

This value is used to determine the size of the queue-key table used by DSLQMGT. The minimum size of one queue-key table entry is 13 bytes; the maximum size is 13 plus the maximum of the key lengths specified in the function table. Each queue element has one queue-key table entry.

Note: When QIO=DSLQMDIO (queue management using DB2) is specified, this parameter is meaningless. With DB2 the theoretical number of queue elements that can be stored is 9 999 999 999 999.

OPID
The first 3 characters of the user identification for those MERVA ESA operators who are authorized to use the restricted operator commands. The default is MAS. For more information about using restricted operator commands, refer to the *MERVA for ESA Operations Guide*.

If you specify

- OPID=///, MERVA ESA does not check the first 3 characters of the user identification. Instead, MERVA ESA checks whether the MERVA ESA operators contain the authorization for the restricted operator commands in their user file records.
- OPID=***, MERVA ESA does not check the authorization of the operator. Instead, the user exit DSLNCU01 must check the authorization of the operator.

If you want the user identification specified in parameter APIUID to get the authorization for the restricted operator commands, this user identification must start with the same 3 characters as specified in OPID. That is, the APIUID user identification must start with /// when OPID=/// is specified.

Except for OPID=***, an operator is **always** authorized if the user record of this operator indicates it.

OPINT
For MVS only, whether operator intervention is required when the DSLICA address specified by the CVTEXTO parameter is to be reused:

YES	The message DSL090A is issued at the operating system console, and the operator must answer YES or NO to reuse the DSLICA address. This is the default.
NO	The DSLICA address is reused without operator intervention.

PGBUF
The size of the signon data buffer used when PGCALL=YES is specified. The MERVA ESA End User Driver program (DSLEUD) allocates the specified storage area to store the signon data. This signon data is transferred from the user-written program that calls DSLEUD. The default is 1024.

PGCALL
Whether user-written programs can call the MERVA ESA end user driver program (DSLEUD). When YES is specified, the parameter EXSEC=YES must also be specified. The default is NO.

PRTNAME
An institution name of up to 60 characters. It is printed in the printout of most REXX batch utilities. If the name to be printed contains blanks, enclose it in quotes, for example, PRTNAME='SAMPLE Bank Boeblingen'. If this parameter is omitted, no institution name is printed.

QDS
The number of queue data sets and the processing disposition after an I/O error.

The first parameter shows the number of data sets.

1 A single queue data set is available. The DDNAME/DLBL name is DSLQDS. This is the default.

2 Duplicate queue data sets are available. The primary data set is used for all queue management requests. The secondary data set is used only for requests that change the content of the queue data set. The DDNAME/DLBL name of the secondary QDS is DSLQDS2.

The second subparameter applies only when QDS=2 is specified. It shows if processing will continue with only one data set after an I/O error has caused the other data set to become inoperable.

STOP	Closes the remaining data set and stops MERVA ESA processing. This is the default.
CONT	Continues processing with the remaining data set. The queue data set log record is updated. After normal end of processing, MERVA ESA cannot be restarted until the data sets are made equal. CONT implies the risk that the remaining queue data set becomes inoperable also and no intact queue data set remains for backup.

Note: When QIO=DSLQMDIO (queue management using DB2) is specified, this parameter is meaningless.

QIO
The queue management I/O module to be used in the installation.

DSLQMGIO
The module DSLQMGIO designates the standard I/O module for queue data sets using VSAM for access at record level. This is the default.

DSLQMCNV
Alternatively, the I/O module DSLQMCNV can be specified. This parameter is only valid for MERVA ESA running under MVS. This module achieves performance improvements by using:

- Access on control interval level
- Read buffering in a separate data space
- Write buffering for selected processing modes; write buffering is done only when the integrity of the QDS is guaranteed

The second subparameter consists of up to 4 characters which select the features used by the I/O module. If DSLQMCNV is specified, the value VXBD is recommended as the second subparameter:

- The first character specifies the access method VSAM (V).
- The second character specifies whether the VSAM requests are performed asynchronously (A), synchronously (S), or dependent on the environment (X). When X is specified and MERVA ESA is running under CICS the access is performed asynchronously; this allows other tasks to perform some work during the I/O wait interval. When X is specified and MERVA ESA is not running under CICS the access is performed synchronously.
- The third character specifies whether some eligible write requests can be buffered (B), or all write requests must be executed directly (D). The integrity of the QDS is guaranteed, even when B is specified.
- The fourth character specifies whether a data space should be used as cache (D) or not (blank). The data space cache is used for read requests only.

DSLQMDIO
The module DSLQMDIO specifies that DB2 is used for the queue management.

QTRACE
The status of the queue management trace. When the trace is on, all queue management requests are recorded in the MERVA ESA journal.

OFF The queue trace is off. This is the default.

SMALL The queue trace is on. The queue management parameter list and the queue element prefix will be journalled. The message data is not included in the journal record.

With queue management using DB2 only the queue parameter list will be journalled.

LARGE The queue trace is on. The queue management parameter list and the complete queue element will be journaled. The data of large messages is not journalled.

With queue management using DB2 the queue parameter list, DB2 return information, queue descriptors, and the message are journalled.

Note: If QTRACE=LARGE is specified, and the record size in the journal data set is too small to contain the complete message, the message is truncated in the queue trace.

RACFSVC
For MVS only, a type 3 SVC number for the program DSLEUSVC. This program checks the password entered for the MERVA ESA user file maintenance function against the password recorded by the ESM for the signed-on MERVA ESA user.

The password check is only performed if three other parameters of DSLPARM are specified as follows:

- EXSEC=YES
- EXUID=YES
- PGCALL=YES

A decimal value from 200 to 255 can be specified. The value must be different to the value specified in parameter SVC.

The RACFSVC parameter is optional if MERVA ESA is running under CICS/ESA Version 4.1 or later. Beginning with that version CICS/ESA provides a service equivalent to the function of program DSLEUSVC. If, however, the CICS service is not to be used, the parameter must be specified. In this case, do the following:

- Install the module DSLEUSVC as a type 3 SVC.
- Modify the DSLEUD user exit DSLEU004 to enforce the usage of the SVC, assemble DSLEU004, and link-edit it to the module DSLEUD.

For previous versions of CICS and for the versions of IMS supported by MERVA ESA, the parameter must be specified. If the parameter is omitted in these cases, the entered password is not checked against the ESM and the password is accepted (provided that user exit DSLEU004 was not modified).

Under VSE, the RACFSVC parameter is ignored.

RECON
Whether the MERVA ESA Traffic Reconciliation feature is used. This feature is distributed separately from MERVA ESA and only for MVS. The default is NO.

The second subparameter applies only when RECON=YES is specified. It determines whether processing continues if the Traffic Reconciliation feature cannot be initialized or becomes inoperable:

CONT The MERVA ESA initialization or processing continues.

Note: If you specify CONT and MERVA Traffic Reconciliation terminates, you will lose events that MERVA Traffic Reconciliation might otherwise have captured.

STOP MERVA ESA terminates. This is the default.

RTLOAD
What MERVA ESA is to do when the loading of a routing table fails:

CONT
Continue processing. MERVA ESA issues the message DSL036I for each routing table that cannot be loaded. This is the default.

STOP Stop processing.

RTRACE
The initial status of the MERVA ESA routing trace facility. This facility traces the evaluation of routing tables during message routing operations.

OFF	The routing trace is off. This is the default.
ALL	The routing trace is on, all routing operations that use routing tables are traced.
WARNING	Warning and severe errors are traced.
SEVERE	Severe errors are traced.

The routing trace status can be controlled by the MERVA ESA operator with the command **rswitch** (refer to the *MERVA for ESA Operations Guide*).

SDDB2
With MERVA ESA queue management using DB2 (QIO=DSLQMDIO) under

MVS, whether the MERVA ESA batch programs DSLSDI, DSLSDO, DSLSDY, and DSLSDxR are to access the queue data set on DB2 directly (SDDB2=YES), or use intertask communication to access the central queue management service of the MERVA ESA nucleus (SDDB2=NO). The default is NO.

Important: SDDB2=NO is required when MERVA ESA uses the standard VSAM queue data set.

SDRC

Whether the MERVA ESA batch programs DSLSDI, DSLSDO, and DSLSDY are to indicate their completion state in a return code and issue an additional diagnostic message with a reason code. This is described in detail in *MERVA for ESA Messages and Codes*. The default is NO.

In previous versions of MERVA ESA, the programs returned the reason code. This was changed to simplify the return codes. If you are using the reason codes to control the following job steps and don't want to change an existing JCL, specify SDRC=YES to continue to get the reason code.

SONNUM

The number of successive signon trials with an incorrect password until a user is revoked.

You can specify a value from 0 to 63. If you specify SONNUM=0, the users will never be revoked. The default is 0.

SVC

For MVS only, a type-3 SVC number for the installation of the interregion communication program DSLNICP. A decimal value from 200 to 255 can be specified.

If this parameter is omitted, any request for MERVA ESA interregion communication will result in a return code indicating that MERVA ESA is not ready. Under VSE, the SVC parameter is ignored.

TFD

The name of the MERVA ESA terminal feature definition table. The default is DSLTFDT.

In IMS systems, this table is required for description of all MERVA ESA terminals. In CICS systems, this table may be used to describe the output device of the sequential data set print program DSLSDY and the terminal printers for the hardcopy print program DSLHCP.

TIER

The licensed usage for this MERVA ESA installation. Starting with MERVA ESA V4.1, this number represents a specific MERVA ESA usage, which is calculated as the average monthly number of messages sent via the SWIFT Link, via the Telex Link, and via the FMT. The number of messages licensed for this MERVA ESA installation must be specified. When the monthly average usage exceeds the licensed level, the customer is required to contact the IBM Marketing representative to order the additional message licenses.

The default is 0. A setting of 0 means that the system is not licensed to send any messages via the SWIFT Link, FMT, or Telex Link.

TOFSIZE

The size of the TOF and its increase value for dynamic TOF:

- The first subparameter is the size of the single TOF. It is recommended that you use a value greater than 16383 for the first subparameter.

- The second subparameter is the increase value for dynamic TOF. Setting the second subparameter to 0 disables the dynamic TOF.

A value from 4096 to 2097152 (2MB) can be specified for both subparameters. The default is (18432,0).

Note: The value specified in this parameter is used by many MERVA ESA programs to determine their TOF storage requirements.

TRACE
The MERVA ESA processing trace status.

INT The processing trace table is in main storage only. This is the default.

EXT The processing trace table is in main storage and is written to the MERVA ESA journal.

OFF The processing trace table is not used. For the MERVA ESA nucleus and its associated servers, the internal trace is always used, even when OFF is specified.

Note: The size of the processing trace table is determined by the size of the MERVA ESA journal buffer (see JRNBUF parameter on page 187).

TSAUX
In CICS, how the transaction storage of an end user is saved between transaction steps:

YES Saved on auxiliary temporary storage (disk). This is the default.

NO Saved in main temporary storage.

UCTRAN
Whether uppercase translation is to be done on screen or printer devices:

NO Data is displayed in uppercase and lowercase. This is the default.

YES Lowercase data is translated into uppercase so the display shows data in uppercase only.

UMR
Whether the unique message reference (UMR) option is active:

NO The UMR option is off. No new reference numbers are assigned. This is the default.

YES The UMR option is on. Each message without a UMR is assigned a new UMR at the time the message is put into the queue data set.

YES,IMM
The UMR option is on. A new UMR is immediately assigned whenever a MERVA ESA user creates a new message on an End-User Driver message processing screen. The UMR number is displayed on the screen during message data entry.

USER
The maximum number of active users. A value from 1 to 32000 can be specified. The default is 20.

USERSTO
The storage sizes for the end-user driver DSLEUD. The first subparameter is the SPA size for user exits. The second subparameter is the temporary storage for DSLEUD exits. DSLEUD requires at least 200 bytes for the exit save area. The default is (0,208).

USFPW
Whether the password fields of the user file maintenance record panel are to be displayed:

YES Display the password fields. This is the default.

NO Do not display the password fields. In this case, EXSEC=YES must also be specified.

USGRP
Whether the option to categorize users into groups is active (if so, the values entered in the user file for user ID, origin ID, and user functions are checked against the values of the group):

NO The USGRP option is off. The GROUP field in the user file is not used (and not displayed). This is the default.

YES,OPTIONAL
The USGRP option is on. The 'Group Id' field in the user file can be entered. The second subparameter may be specified as OPT or OPTIONAL.

YES,REQUIRED
The USGRP option is on. The 'Group Id' field in the user file must be entered. The second subparameter may be specified as REQ or REQUIRED.

The group table DSLGRPT is defined with the DSLGRP macro.

WSASRV
If you are using the APPC/MVS version of the MERVA Message Processing Client Server, you must specify this parameter. There are four subparameters:

1. A symbolic destination name that represents the transaction program and local LU for which MERVA Message Processing Client Server will register. If used, this subparameter must contain a value that matches a symbolic destination name defined in the active side information data set. APPC/MVS obtains the TP name and local LU name from this side information. This subparameter is ignored if the following subparameters are specified.
2. The local LU name where MERVA Message Processing Client Server resides; this must be a NOSCHED LU. A NOSCHED LU is an LU that is not associated with a transaction scheduler. Like other LUs, NOSCHED LUs are defined through the LUADD statement in the APPCPMxx member of SYS1.PARMLIB. For more information about defining NOSCHED LUs, see *MVS/ESA SP V5 Initialization and Tuning Reference*. The length of the local LU name is up to 8 bytes.
3. The MERVA Message Processing Client Server transaction program name. The name can be up to 64 characters. This name should be unique for a given MERVA ESA installation using the same local LU name. The same transaction program name is used for the server and for the requestor.
4. The security profile for MERVA Message Processing Clients. This subparameter is optional and can be used to restrict access to the MERVA Message Processing Client Server. If specified, only clients with this specific security profile are accepted. APPC/MVS treats the profile name as a RACF group name.

WSSEC
The security level for application data transferred between MERVA ESA and

MERVA Message Processing Client workstations. Client workstations must specify the same level. The value can be either of:

AUTHENTICATION
Application data is transferred as clear text together with a signature which is used to verify unimpaired data transfer. A proprietary algorithm is used to generate the signature. (Passwords are always encrypted.) This is the default.

ENCRYPTION
Application data is encrypted using a proprietary algorithm.

The specification can be abbreviated by omitting trailing characters, for example WSSEC=ENCR or WSSEC=E.

WSTSRV
The TCP port number on which the TCP/IP version of the MERVA Message Processing Client workstation server listens for client connections. If the parameter is omitted, or if 0 is specified, the port number defaults to 7117.

Note: The sum of the following parameters must not exceed 32760 bytes:

- The size of the End-User Driver permanent storage (3200 bytes)
- The size of the SPA for user exits (USERSTO parameter of the DSLPARM macro)
- The size of MFS permanent storage and the retype buffer (MFSSTOR parameter of the DSLPARM macro)
- The size of the load table for the Message Control Blocks (the value of the MCBNUM parameter of the DSLPARM macro multiplied by 16)
- The size of the permanent storage for each function program (limited to 4096 bytes per function program)

Mapping the Customizing Parameters

Name	Operator	Operands
[*label*]	**DSLPARM**	**TYPE=MAP**

Programming Notes:

label
A unique statement label according to assembler language conventions. The default value is DSLPRM.

TYPE
TYPE=MAP must be specified for this form of the macro. If the TYPE parameter is specified, MAP is assumed and all other parameters are ignored.

DSLQMG: Defining Queue Management Services

The DSLQMG macro serves the following purposes:

- Prepare the queue management parameter list for a MERVA ESA service request. The EP parameter controls whether the request will be executed as a direct or central service.
- Map the queue management parameter list and the LIST response buffer.

Calling DSLQMGT

Use the DSLQMG macro to add, retrieve, replace, or delete queue elements in the queue data set (QDS), and to get status information about queues and the QDS.

Name	Operator	Operands
[*label*]	**DSLQMG**	**TYPE=**{**DELETE** }
		{**FREE** }
		{**GET** }
		{**GETLAST**}
		{**GETNEXT**}
		{**HOLD** }
		{**LIST** }
		{**MPUT** }
		{**PUT** }
		{**REPLACE**}
		{**RESET** }
		{**ROUTE** }
		{**SET** }
		{**SQLSTAT**}
		{**START** }
		{**STATUS** }
		{**TEST** }
		{**UMR** }
		[**,DATA=**{*addr*|(*r*)}]
		[**,ECB=**{*addr*|(*r*)}]
		[**,EP=DSLQMGT**]
		[**,KEY=**({*addr1*|(*r1*)},{*addr2*|(*r2*)})]
		[**,LMOD=**{(*lmod*,...,*lmod*)|(*r*)}]
		[**,LNQE=**{**10**|*addr*|(*r*)|*nnnnn*}]
		[**,MF=(E,**{*addr*|(*r*)})]
		[**,MODIF=**(*mod*,...,*mod*)]
		[**,QSN=**{*addr*|(*r*)}]
		[**,QUEUE=**({*addr1*|(*r1*)},{*addr2*|(*r2*)},{*addr3*|(*r3*)})]
		[**,RES=**{*addr*|(*r*)}]
		[**,RT=**{*addr*|(*r*)}]

Programming Notes: For the address parameters in this macro, the following notations are possible:

addr A symbolic label.

(r) A general register containing the address.

The description of each parameter includes details about how the symbolic label or general register is used.

label
A unique statement label according to assembler language conventions.

TYPE
The MERVA ESA queue management function to be performed. Some functions can be modified by the MODIF parameter.

DELETE
Deletes a queue element from a queue.

FREE
Turns off the *in service* indicator in the queue element identified by key or QSN. If modifier ALL is specified, all queue elements of a queue are freed.

GET
Retrieves a queue element from a queue. If one of the keys or a queue sequence number (QSN) is specified (see KEY parameter on page 202 and QSN parameter on page 206), the queue element with this key or this QSN is returned to the calling program, if this queue element exists in the queue and that it is not flagged *in service*.

If a key or a QSN is not specified, the first queue element in the indicated queue that is not flagged *in service* is returned to the calling program.

GETLAST
Retrieves the last queue element of that queue.

GETNEXT
Retrieves the queue element that is not flagged *in service* with the next higher QSN than specified in the call.

HOLD
Sets a function into HOLD state. This request type is only valid when MERVA ESA is customized for queue management using DB2.

LIST
Prepares a list of queue key table entries for the queue specified by the parameter QUEUE. Using the KEY, QSN and the LMOD parameters, the list can:

- Start at the begin of the queue key table
- Start with a key or QSN
- Start at the end of the queue key table
- Continue going forward or backward
- Select queue key table entries with key values and the *in service* status on

If no additional parameter is specified, the LIST response starts at the beginning of the queue. The LIST response can be mapped using the macro DSLQMG MF=LIST. When the end of the queue is reached, the reason code field of the queue parameter list contains the reason LISTEND.

MPUT
Adds a new queue element at the end of each queue specified in the QUEUE operand of DSLQMG (described on page 206). If the MPUT was originally requested by the calling program, from 1 to 3 queues may be

affected by one request. If the MPUT is the consequence of a ROUTE request, from 1 to 12 queues may be affected by one request.

PUT
Adds a new queue element at the end of the specified queue.

Note: Successful MPUT or PUT or ROUTE requests may also start a transaction or post an ECB (see QUEUE parameter on page 206).

REPLACE
Writes an updated queue element to the QDS and deletes the corresponding old queue element. The *in service* indicator remains unchanged.

RESET
Removes the ECB address from a function table entry. The RESET request must be a direct service call (see EP parameter on page 202).

ROUTE
Calls the MERVA ESA routing facility (DSLRTNSC) using the routing table given with the RT parameter or using the queue name of the function containing the routing table address. After successful routing, the same functions are carried out as for TYPE=MPUT, except that routing is possible to up to 12 queues. ROUTE also returns up to 3 queue names in the fields QPLNAM1, QPLNAM2 and QPLNAM3. All 12 queue names, the QSNs and the keys are returned in the queue parameter list extension obtained with the macro DSLQMG MF=L,EXT=YES.

SET
Adds an ECB address to a function table entry. This ECB will be posted when a PUT or MPUT or ROUTE or FREE request is successfully executed for the related queue. The SET request must be a direct service call (see EP parameter on page 202).

SQLSTAT
For queue management using DB2 (DSLPARM parameter QIO=DSLQMDIO), returns the last SQL error information in the form of message DSL147I. The SQLSTAT request must be a direct service call (see EP parameter on page 202). The data is returned in the data buffer provided by the caller.

START
Starts a transaction without a message having been written to a queue. No error is indicated if a START request is used for a queue that has no associated transaction (no transaction name in function table entry).

START ignores the ACTIVATED status of the function.

STATUS
Prepares the message DSL146I and returns it to the caller with status information about the queue key tables, the queue data set and the unique message reference (UMR) if it is used in this MERVA ESA. The STATUS request must be a direct service call (see EP parameter on page 202). The data is returned in the data buffer provided by the caller or, if not provided, the address of the information message is returned in general register 1.

For queue management using DB2 the status information contains the number of messages, the last used message table number and the last UMR.

TEST
Determines the actual number of elements and the highest QSN in the specified queue (see QUEUE parameter on page 206) and returns the numbers to the calling program in the fields QPLNQE and QPLQSN respectively.

UMR
Assigns the next available unique message reference (UMR) number and returns it to the calling program in the queue parameter list field QPLUMR.

Note: All queue management requests that handle queue elements return the QSN and the keys (if available) in the parameter list fields QPLQSN, QPLKEY1, and QPLKEY2. For MPUT and ROUTE, QSN of the second and third target queues are returned in fields QPLQSN2 and QPLQSN3 respectively, when applicable.

DATA
A 4-byte field containing the data address or a general register containing the data address:

- For GET or GETNEXT or GETLAST, this address shows the area where the retrieved queue element is to be stored for use by the calling program.
- For PUT or MPUT or REPLACE or ROUTE, this address shows where DSLQMGT finds the data to be stored as a queue element in a queue.
- For LIST, this address shows the LIST response buffer.

For SQLSTAT and STATUS, this address shows the response buffer.

Note: The data area must begin with 8 bytes reserved for length information according to MERVA ESA standards (see the *MERVA for ESA Customization Guide* for details).

The rest of the buffer must be cleared to X'00'.

ECB
A field or general register containing the address of an ECB for TYPE=SET.

EP
The only possible value for the EP parameter is DSLQMGT. It specifies that the request will be executed as a direct service.

The EP parameter must be specified by all programs that are link-edited to DSLNUC, that is, which are contained in the MERVA ESA nucleus program table DSLNPTT or which are called by programs in the table. Before issuing a direct service request, the program must ensure that general register 12 points to DSLCOM and that general register 13 points to a usable save area.

The parameter EP=DSLQMGT must not be used by programs that are not link-edited to DSLNUC. These programs must, after using the DSLQMG macro to prepare the parameter list, use a DSLNIC macro with TYPE=REQ and NAME=DSLQMGT to request execution of a MERVA ESA central service.

If the EP parameter is omitted or does not specify DSLQMGT, a central service request is assumed unless it is mandatory for the specified TYPE parameter.

KEY
The fields that contain the first and the second key of a queue element:

- For TYPE=MPUT, PUT, REPLACE, and ROUTE, the KEY parameter must be specified only if DSLQMGT is not to retrieve the keys from the queue element according to the specifications of the queue in the MERVA ESA function table (refer to the description of the macro DSLFNT on page 30). If, for an MPUT, PUT, REPLACE or ROUTE request, more keys are specified than defined in the function table entry for the queue, the keys are ignored without error notification.

- For TYPE=GET, either the first or second key may be specified. If both keys are specified, the first is used and the second is ignored. If, for a GET request, key 1 or key 2 is specified for a queue for which this key is not defined, the request is not executed and an error is returned.
- For TYPE=LIST with LMOD=ONLY, both keys are considered, if specified.
- For TYPE=FREE, DELETE, and REPLACE, use the QSN instead of a key, because a key is not necessarily unique in a queue.

Of the maximum key length of 24 characters possible in the queue parameter list, DSLQMGT uses only the length defined in the function table entry of this queue. Keys that are shorter than the function table specification must be padded with binary zeros or blanks. DSLQMGT replaces the blanks by binary zeros both when storing and retrieving a message.

If KEY=(0,0) is specified, the key fields in the queue parameter list are cleared to binary zeros. If the KEY parameter is not specified, the DSLQMG macro does not change the content of the key fields in the queue parameter list.

LMOD

Used only with TYPE=LIST, this parameter specifies a list of request modifier values or a general register containing the request modifier value:

BACK Select queue key table entries from the specified QSN going backward. If BACK is not specified, entries from the specified QSN going forward are selected.

BUSY Select only queue key table entries that have the *in service* indicator on.

FIRST Start the LIST response at the first appropriate queue key table entry in the queue and go forward.

LAST Start the LIST response at the last appropriate queue key table entry in the queue and go backward.

ONLY Select only queue key table entries that match the specified key parameter for full or generic key 1 and key 2.

More than one value for this parameter can be specified, as long as the values are not conflicting. If this parameter is omitted, all queue key table entries from the specified QSN or KEY going forward are selected. If these parameters are also omitted, LMOD=FIRST is assumed.

LNQE

Used only with TYPE=LIST, this parameter specifies the number of list items to be returned in the LIST response buffer. Its value is one of the following:

- A decimal number
- The address of a halfword field or general register containing a binary number

The default is 10 (decimal).

MF
The format of the macro. The first subparameter must be **E** (for *execute form*). The second parameter, **addr** or **(r)**, specifies the address of the queue parameter list. The default is (E,(1)).

MODIF
A list of up to 4 request modifier values. This parameter can be used with one of the following:

- TYPE=FREE
- TYPE=GET
- TYPE=GETLAST
- TYPE=GETNEXT
- TYPE=INIT
- TYPE=PUT
- TYPE=MPUT
- TYPE=ROUTE
- TYPE=START

For all other specifications of the TYPE parameter, MODIF is ignored. The request modifiers and their meanings are:

ALL
Used with the FREE request, ALL indicates to DSLQMGT to turn off the *in service* indicator in all queue elements of the specified queue.

DIRECT
For queue management using DB2 (DSLPARM parameter QIO=DSLQMDIO) and used with the INIT request, DIRECT indicates that queue management using DB2 as a direct service is required.

DYNBUF
Used with GET, GETNEXT or GETLAST requests, DYNBUF indicates to DSLQMGT that, if the buffer provided by the calling program for the retrieval of the message is too small that DSLQMGT allocates a larger buffer for the message. For direct service calls, the address of this buffer is returned in the field QPLAD of the queue parameter list. For central service calls, the address of the buffer is returned in the field NICBUF of the DSLNIC parameter list.

Refer to the chapter on retrieving messages in the *MERVA for ESA Customization Guide* for details.

FREE
Used with GET, GETNEXT or GETLAST requests, the FREE modifier causes a queue element to be retrieved without setting the element *in service*. If the queue element is already *in service*, its status is not changed by the FREE modifier.

Used with REPLACE requests, the *in service* status of the queue element is reset.

GENERIC
Used only with TYPE=GET, this modifier is used to retrieve a message whose key contains the substitution characters '*' (asterisks) or '%' (percent). The key is used as a generic key:

- The substitute character * represents any number of characters. As many asterisks as required can appear anywhere in the key.

- The substitute character % is a place-holding character, representing any single character. As many percent symbols as necessary may appear anywhere in the key.

If modifier GENERIC is not used, the key is exactly taken as it is.

IGNHOLD
Ignore HOLD status causes the requested queue element to be retrieved although the queue is in HOLD status.

IGNINS
Ignore *in service* status causes the requested queue element to be retrieved although the element is flagged *in service*.

LAZY
Used with PUT requests this operand initiates deferred queue management write requests to the queue data set (for performance reasons). This operand must be used with care. It should only be used when your application provides restart logic. Every subsequent queue management request without the LAZY operand resets the lazy write state.

NEWUMR
Used only with MPUT, PUT or ROUTE requests, this modifier requests a new unique message reference number (UMR), even if the message has already been assigned a UMR.

This modifier cannot be specified together with the NOUMR modifier.

NOCOMMIT
For queue management using DB2 (DSLPARM parameter QIO=DSLQMDIO) and used with the INIT request in concurrence with the DIRECT operand, NOCOMMIT indicates to queue management not to perform any COMMITs to the QDS on DB2. The user is solely responsible for the commit points.

NOUMR
Used only with MPUT, PUT or ROUTE requests, this modifier suppresses the assignment of a unique message reference number, even if the message does not yet have one.

This modifier cannot be specified together with the NEWUMR modifier.

RESTORE
Used only with PUT requests, this modifier indicates to queue management that the queue element identified by its QSN is to be restored to its former queue.

ROUTE
Used only with GET, GETLAST and GETNEXT requests, this modifier gets the message from the input queue and causes an immediate routing of the message with automatic deletion in the input queue (see the RES parameter). After completion of the request, the queue sequence number (QSN) of the message in the input queue is returned in the field QPLQSN, and the up to three first queues resulting from the routing are returned in the fields QPLNAM1, QPLNAM2 and QPLNAM3.

This modifier improves the performance when routing messages from an intermediate queue to final queues, for example, in the program DSLSDI.

RTNONLY
Used only with ROUTE requests, this modifier causes an immediate return after the target queue names have been determined. The MPUT functions are not performed.

SETINS
Used only with PUT requests, this modifier sets the *in service* status when storing a queue element in a queue. This is useful when transaction programs use an intermediate queue and want to prevent other programs from accessing the message before processing is complete.

SFCOMP
Used only with START requests, this modifier indicates to queue management that a delayed start request has been completed (internal use only).

WRTBACK
Used with GET, GETNEXT or GETLAST requests, this modifier will cause to a second retrieval request to the same queue element as the present one an information code to be issued together with the retrieved queue element, indicating that there was a previous retrieval of this element.

This writeback indicator is maintained over a MERVA ESA termination and restart.

QSN
A 4-byte field that contains the queue sequence number (QSN) or a general register that contains the QSN to be used by DSLQMGT.

For GET, the QSN identifies the queue element to be retrieved from the queue specified by the QUEUE parameter.

For GETNEXT, the QSN identifies the position in the queue specified by the QUEUE parameter from which the search should begin. The element identified by QSN need not really exist.

For REPLACE, FREE and DELETE, the QSN identifies the queue element to be effected in the queue specified by the QUEUE parameter.

For MPUT, PUT and ROUTE, the QSN identifies the queue element to be deleted from the queue specified by the RES parameter.

For PUT with modifier RESTORE the QSN identifies the queue element to be restored to the queue specified by the QUEUE parameter.

If QSN=0 is specified, the QSN field in the queue parameter list is cleared to binary zeros.

If the QSN parameter is not specified, the DSLQMG macro does not change the content of the QSN field in the queue parameter list.

QUEUE
A list of up to 3 fields that contain the names of the queues to be affected, or a list of up to 3 general registers that contain the addresses of the fields that contain the queue names.

A queue name field is 8 characters long. Binary zeros or blanks show an empty field. Shorter queue names must be padded with blanks.

For all relevant queue management requests, the first field of the queue name list must contain a valid queue name. For TYPE=MPUT, the second and third fields must contain either valid queue names or binary zeros. If the second or third queue name field is empty (containing binary zeros instead of a queue name), no error is indicated and MPUT is executed for the queue(s) really specified. For TYPE=PUT, the second and third queue name fields are ignored.

For TYPE=ROUTE, QUEUE specifies the name of the function table entry containing the routing information, for example, the name of the queue from which the message to be routed was retrieved. After successful completion of the ROUTE request, the first 3 target queue names are contained in the queue parameter list fields QPLNAM1 to QPLNAM3, and, if the queue parameter list was defined with the EXT=YES parameter, up to 12 queue names are contained in the queue parameter list extension.

If the QUEUE parameter is not specified, the DSLQMG macro does not change the content of the queue name fields in the queue parameter list.

Note: If a message is written to a queue by MPUT, PUT or ROUTE and a transaction name is specified in the function table entry applying to this queue, and if this function table entry is not in a HOLD status, the associated transaction is started, for MERVA ESA running in CICS, by an EXEC CICS START command or, for MERVA ESA running in IMS, by inserting a message into the IMS message queue.

If the function table entry contains the address of an ECB in the field FNTECBA, this ECB is posted. The ECB address can be provided by programs link-edited to DSLNUC by means of a DSLQMG TYPE=SET macro to get notice when a message is written to this queue.

RES

Used together with the QSN parameter, RES indicates that the current message should be deleted from its previous queue (automatic delete) when processing a PUT, MPUT, or ROUTE request. RES specifies an 8-byte field that contains the name of the queue from which the current message is to be deleted, or a general register containing the address of the field that contains the queue name.

QSN must also be specified (see QSN parameter on page 206). If either RES or QSN is empty (that is, contains binary zeros) no automatic delete is performed.

If both RES and QSN are specified, they are verified for being valid before the put is tried. If the verification fails because this queue element is not found, the put request is rejected with the return code for key not found.

The RES queue element will be deleted only after successful completion of the put but before DSLQMGT returns to the calling program. When DSLQMGT is interrupted between the put operation and the delete, the RES queue name and the QSN become the restart information used by DSLQMGT to delete the obsolete queue element during the restart which will be carried out at the next DSLQMGT initialization (see *MERVA for ESA Concepts and Components*).

If RES=0 is specified, the restart queue name field in the queue parameter list is cleared to binary zeros. If the RES parameter is not specified, the DSLQMG macro does not change the content of the restart queue name field.

Note: Successful completion to the MPUT, PUT or ROUTE request changes the contents of the field QPLQSN to show the queue sequence number of the first queue resulting from the put.

RT

A field or general register containing the address of a routing table for

TYPE=ROUTE. This parameter can be used only by programs which are controlled by DSLNUC, that is, which are contained in the MERVA ESA nucleus program table DSLNPTT.

Mapping the Parameter List for DSLQMGT

The DSLQMG macro is used to generate a storage definition of the DSLQMGT parameter list and the LIST response buffer.

Name	Operator	Operands

[*label*]	**DSLQMG**	**MF=**{**L**[**,EXT=**{<u>**NO**</u>|**YES**}]}
		{**LIST** }

Programming Notes: When executing the DSLQMG macro, a field in the parameter list of DSLQMGT is not changed unless the appropriate parameter is specified. However, the unique message reference field QPLUMR is cleared by the queue management program whenever a queue management call is performed.

DSLQMGT gives a return code always in the QPLRETCD field of the queue parameter list, and in general register 15 only when the EP parameter has been used. The return codes are explained in *MERVA for ESA Messages and Codes*.

label
A unique label according to assembler language conventions.

MF
The macro format:

L Maps the DSLQMGT parameter list.

The EXT parameter specifies whether the queue parameter list extension is to be mapped. After a DSLQMG TYPE=ROUTE, this extension can return the names of up to 12 queues, the QSNs, and the keys to the calling program. If EXT=NO is specified, only the names and QSNs of up to 3 queues are returned, even if routing resulted in more than 3 queues.

LIST Maps the LIST response buffer.

DSLROUTE: Defining a Routing Table

Use the DSLROUTE macro to define MERVA ESA routing tables.

A routing table consists of a sequence of DSLROUTE macros which allow for inspecting message contents to produce a list of up to 12 routing target functions which are to be used by DSLQMGT as the target queues of an MPUT request.

Note: Routing can be invoked only by a DSLQMGT ROUTE request.

The routing table evaluation may be traced in the MERVA ESA journal, depending on the RTRACE parameter of the DSLPARM macro or on the operator command **rswitch** (see the *MERVA for ESA Operations Guide*).

The DSLROUTE macro has five functions, to:

- Define a variable routing field
- Test the content of a variable routing field for conditional branching
- Set a target routing function
- Drop a variable routing field
- Provide a default target routing function for error handling

Defining a Variable Routing Field

The DSLROUTE macro can define a variable routing field for use in later DSLROUTE macros for testing its contents or for setting a target routing function. Up to 20 fields can be defined concurrently.

Name	Operator	Operands
[*label*]	**DSLROUTE**	**TYPE=DEFINE** **,FIELD=**(*name*,{'*literal*'|*toffield*}) [**,DISP=***nnn*] [**,EMPTY=***label*] [**,FOUND=***label*] [**,GOTO=***label*] [**,LENGTH=***nnn*] [**,NOTFND=***label*] [**,UCTRAN=**{**NO**|**YES**}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE=DEFINE
Defines a variable routing field for use in the routing table.

FIELD
The name and content of a variable routing field. The maximum length of a variable routing field is 32 characters.

name The first subparameter specifies the name with which the field can be referenced in later DSLROUTE macros. Any name can be chosen which is up to 8 characters long, following assembler language conventions.

The same name can be reused in a later DSLROUTE TYPE=DEFINE macro when necessary. No DSLROUTE TYPE=DROP macro instruction is required then.

The second subparameter specifies the data item:

literal An alphanumeric literal. It may be up to 32 characters long, enclosed in single quotes.

toffield A MERVA ESA TOF field with a list of positional subparameters: field name, nesting identifier, field group index, repeatable sequence index, data area index, and one or two DSLTSV modifiers (see "Examples" on page 211 for an example). The field name is always required. The indexes and modifiers are optional, but either an index or a modifier should be specified to find the field in the TOF. If an index is omitted, the default values described for the appropriate DSLTSV macro parameters are used.

The DSLTSV modifiers are (see also "DSLTSV: Defining the TOF Supervisor Interface" on page 233, parameter MODIF):

OPTION	Gets the option of a TOF field
VFIRST	Gets the very first appearance of the TOF field
LASTDA	Gets the last data area of the TOF field
NEXTDA	Gets the next data area of the TOF field
PREVDA	Gets the previous data area of the TOF field
LASTNI	Gets the specified TOF field from the last nesting identifier
NEXTNI	Gets the specified TOF field from the next nesting identifier

NEXTDA, PREVDA, and NEXTNI must not be used in the first DSLROUTE field definition of a routing table because they require a previous definition which cannot exist at that time. NEXTDA is useful to scan data areas forward after having used VFIRST. PREVDA is useful to scan data areas backward after having used LASTDA.

The DSLROUTE macro checks for valid combinations of the modifiers, such as LASTNI and NEXTNI, are mutually exclusive.

TOF field reference parameters must be separated by commas.

The data used from the TOF field may be changed by the LENGTH, DISP, or UCTRAN parameters.

DISP

For TOF field references only, a displacement into the TOF field data where the variable field data begins. The default is 0.

EMPTY

The label of a DSLROUTE macro where processing is to continue if the TOF field is found, but is empty.

The EMPTY label is also used if the displacement specified by the DISP parameter is greater than the actual TOF data length.

The parameter EMPTY is used only for definitions that specify a TOF field.

If the EMPTY parameter is not specified and there was no data found for the variable field in the TOF (also considering the DISP parameter), processing is continued with the next statement in this routing table.

FOUND
The label of a DSLROUTE macro where processing is to continue if the TOF field is found and contains enough data when considering the DISP parameter to define at least one character for the variable routing field.

The parameter FOUND is used only for definitions that specify a TOF field.

If the FOUND parameter is not specified and there was data found for the variable field in the TOF (also considering the DISP parameter), processing is continued with the next statement in this routing table.

GOTO
The label of a DSLROUTE macro where processing is to branch unconditionally after the DEFINE has been performed. The parameter GOTO is used only for definitions that specify a literal.

LENGTH
The number of characters that are to be taken for the variable routing field from the TOF field. The maximum length is 32 characters. The default is 32.

If the TOF field contains more than 32 characters after any specified displacement, only the specified or default number of characters are defined for the variable routing field.

If the TOF field contains less characters than the specified or default length after any specified displacement, only the remaining characters are defined for the variable routing field.

NOTFND
The label of a DSLROUTE macro where processing is to continue if the field is not found in the TOF.

The NOTFND label is also used if an attempt is made to define the 21st variable routing field.

If the NOTFND parameter is not specified but the EMPTY parameter is specified, and the field is not found in the TOF, processing of the routing table is continued at the statement with the label specified in the EMPTY parameter.

If the NOTFND parameter and the EMPTY parameter are both not specified, and the field is not found in the TOF, processing is continued with the next statement in this routing table.

UCTRAN
For TOF field references only, specifies whether the TOF field data is to be translated to uppercase. The default is NO. For DBCS fields UCTRAN is not allowed.

Examples: DSLROUTE **FIELD** example:

All parameters defined in the **toffield** are **positional** subparameters.

If you do not want to code the **VFIRST** modifier the first time you define a field name, you must explicitly qualify the **toffield** with all positional parameters. Look at the following DSLROUTE statement:

```
DWSL1AI0 DSLROUTE TYPE=DEFINE,FIELD=(SIGN,MSGOK,0,1,1,1),EMPTY=AI0,NOTFND=AI0
```

```
where:
```

```
FIELD=(SIGN,MSGOK,0,1,1,1),
       |    |     | | | |
       |    |     | | | |
       |    |     | | | +-------> data area index
       |    |     | | +--------> repeatable sequence index
       |    |     | +----------> field group index
       |    |     +------------> nesting identifier
       |    +------------------> field name
       +-----------------------> routing variable name
```

Alternatively, you can omit the positional parameters when you specify the **VFIRST** modifier the first time you define a field name. This is shown in the following DSLROUTE statement:

```
DWSL1AI0 DSLROUTE TYPE=DEFINE,FIELD=(SIGN,MSGOK,,,,,VFIRST),EMPTY=AI0,NOTFND=AI0

where:

FIELD=(SIGN,MSGOK,,,,,VFIRST),
       |    |         |
       |    |         +--------> 1st DSLTSV modifier
       |    +------------------> field name
       +-----------------------> routing variable name
```

Testing a Variable Routing Field

The DSLROUTE macro can compare a variable routing field to a literal or to another variable routing field, and cause conditional branching within the routing table.

Name	Operator	Operands
[*label*]	**DSLROUTE**	**TYPE=TEST** **,COND=**({*fieldref* },{*fieldref* },*op*[,*modif*]) {'*literal*'} {'*literal*'} [**,FALSE=***label*] [**,NOTFND=***label*] [**,TRUE=***label*]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE=TEST
Evaluates the logical expression defined in the COND parameter and branch to an appropriate label.

When no label has been specified for the resulting condition, processing continues at the next sequential DSLROUTE macro.

COND
The logical relation between the two operands.

fieldref
The name of a variable routing field previously defined by a DSLROUTE TYPE=DEFINE macro.

literal
An alphanumeric literal enclosed in single quotes. The maximum length of the literal is 32 characters.

op A relational operator. One of the following relational operators must be used:

EQ	Equal
NE	Not equal
GT	First operand greater than second operand
LT	First operand less than second operand
GE	First operand greater than, or equal to, second operand
LE	First operand less than, or equal to, second operand

modif
An additional modifier for the comparison. One of the following modifiers can be used:

No modifier	The two operands are compared with their actual lengths.
AMOUNT	Amount fields are to be compared. An amount field is numeric with one decimal comma.
SHORT	If the operands have different lengths, the shorter length is to be used in the comparison.
LONG	If the operands have different lengths, the longer length is to be used in the comparison. The shorter operand is padded with binary zeros.

FALSE
The label of a DSLROUTE macro where processing is to continue if the result of the comparison is false. This label is also used when the AMOUNT modifier was specified, and adjusting the variable routing fields to the same decimal comma positions leads to an overflow. If the result is false and this parameter has been omitted, processing continues at the next sequential DSLROUTE macro.

NOTFND
The label of a DSLROUTE macro instruction where processing is to continue if a specified variable routing field is not defined. If the variable routing field is not defined and this parameter has been omitted, processing continues at the next sequential DSLROUTE macro.

TRUE
The label of a DSLROUTE macro where processing is to continue if the result of the comparison is true. If the result is true and this parameter has been omitted, processing continues at the next sequential DSLROUTE macro.

Setting the Target List Values

This macro is used to set one target routing function into the routing target list.

Name	Operator	Operands
[*label*]	**DSLROUTE**	**TYPE=SET** **,TARGET=**({'*literal*'}[,{'*literal*'}...]) {*fieldref*}[,{*fieldref*}] [,**GOTO=***label*] [,**NOTFND=***label*]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE=SET
Sets a target routing function in the routing target list. The routing target list can contain up to 12 routing targets, and the next free entry in the list is used.

TARGET
The name of the target queue. The target may specify one literal, or one variable routing field, or a list of up to 8 literals and variable routing fields in any combination. If a variable routing field is not found, no queue name is added to the target list. The values specified in the parameter list are put together, in the same sequence as they are coded, to form a target name of up to 8 characters. Names longer than 8 characters are truncated. Names shorter than 8 characters are padded with blanks.

The resulting target name is checked for validity, that is, it is checked if this function exists in the function table. If the target is not found in the function table, the evaluation of the routing table for this message is stopped, and the processing described for TYPE=FINAL takes place, instead.

literal
An alphanumeric literal enclosed in single quotes. The maximum length of the literal is 8 characters. Note that in a list the length of all literals should not exceed 8 characters.

fieldref
A variable routing field previously defined by a DSLROUTE TYPE=DEFINE macro.

GOTO
The label of a DSLROUTE macro instruction where processing is to continue. If this parameter is omitted, processing continues with the next sequential DSLROUTE macro.

NOTFND
The label of a DSLROUTE macro where processing is to continue if a variable routing field is not found as it is not currently defined. If there is an error and this parameter is omitted, processing continues with the next sequential DSLROUTE macro.

Dropping Variable Routing Fields

The DSLROUTE macro can drop variable routing fields.

Name	Operator	Operands
[*label*]	**DSLROUTE**	**TYPE=DROP** **,FIELD=**({*cccccccc*|**ALL**}) [**,GOTO=***label*]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE=DROP
Drops one or more variable routing fields when new ones are required, and the maximum of 20 definitions should not be exceeded.

FIELD
Either one name of the variable routing field to be dropped, or a list of names separated by commas. ALL specifies that all currently defined variable routing fields are to be dropped.

GOTO
A label for unconditional branching after the DROP has been performed.

Ending the Routing Table and Setting a Default Target Queue

This macro must be the last macro of a routing table. It may include the definition of a default target queue.

Name	Operator	Operands
[*label*]	**DSLROUTE**	**TYPE=FINAL** [**,TARGET=**'*literal*']

Programming Notes:

label
A unique statement label according to assembler language conventions. For DSLROUTE macros that cause branching, this label represents the table exit.

TYPE=FINAL
Shows the end of the routing table. This statement is required and must be the last macro in the table before the assembler END statement.

TARGET
The name of the default target queue. Only one literal can be used specifying a complete function name, and the customer should be careful that this function name really exists in the MERVA ESA function table. This function name is taken if routing errors are detected during the processing of the routing table, and if the final routing function exists and is valid, routing always completes successful and does not interrupt the processing of MERVA ESA applications. If a NEXT function is defined in the function table, routing is first tried to that queue. The errors which led to the use of the final routing function can be inspected using the MERVA ESA routing trace.

Note: Some routing tables of MERVA Link require to not specify the TARGET parameter in TYPE=FINAL (see the *MERVA for ESA Customization Guide* for details).

DSLRTC: Defining the Remote Task Communication

The DSLRTC macro serves the following purposes:

- Starting a receiving task, building control blocks between receiving task and the MERVA ESA nucleus
- Making a receiving task available to receive an instruction
- Retrieving an instruction on the receiving side
- Stopping a receiving task, freeing the control blocks linking the receiving task and the MERVA ESA nucleus
- Sending an instruction via the MERVA ESA nucleus to the receiving task
- Mapping the DSLRTC parameter list

The MERVA ESA (MVS only) remote task communication interface enables one remote task, to pass an instruction via the central service DSLNRTCP program, to another remote task where it is received by the remote task program DSLRRTCP.

Communication between the Receiving Remote Task and MERVA ESA

Name	Operator	Operands

[*label*]	**DSLRTC**	**TYPE=**{**START**|**AVAIL**|**RETRIEVE**|**STOP**}
		,COM={*addr*|(*r*)}
		,ID={*cccccccc*|(*r*)}
		,WORK={*addr*|(*r*)}
		[**,BUF=**{*addr*}|(*r*)]
		[**,MF=(E,**{*addr*|(*r*)})]

Programming Notes: For the address parameters in this macro, the following notations are possible:

addr A symbolic label.

(r) A general register containing the address.

label
A unique statement label according to assembler language conventions.

TYPE
A call to the receiving program of remote task communication DSLRRTCP.

START Inform the MERVA ESA remote task central service program DSLNRTCP that this remote task has started.

AVAIL Inform the MERVA ESA remote task central service program DSLNRTCP that this remote task is now able to receive instructions.

RETRIEVE Retrieve the instruction that the MERVA ESA remote task central service program DSLNRTCP has made available.

STOP Inform the MERVA ESA remote task central service program DSLNRTCP that this remote task has stopped.

COM
COM specifies the label of the MERVA ESA communication area DSLCOM or a general register containing the DSLCOM address.

ID The name identifying the receiving task, or a general register containing the address of an 8-byte field containing the name. If a literal is used, it may be up to 8 characters long.

WORK
WORK specifies the label of the DSLRRTCP communication area or a general register containing the address. The work area of DSLRRTCP is 1KB. It must be the same area for all DSLRTC calls.

BUF
For TYPE=RETRIEVE only, specifies the location of a data buffer for the retrieved instruction. It must be a 4-byte field or general register containing the address of the buffer. The maximum length of this buffer may not be larger than specified in DSLPRM by the MAXBUF parameter of the DSLPARM macro. The size of this buffer can exceed 32KB.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the DSLRTC parameter list, or a general register containing the parameter list address. The default is (E,(1)).

Communication between the Instructing Remote Task and MERVA ESA

Name	Operator	Operands
[*label*]	**DSLRTC**	**TYPE=INSTRUCT** **,ID=**{*cccccccc*|(*r*)} [**,MF=(E,**{*addr*|(*r*)})]

Programming Notes: After using the DSLRTC macro to prepare the parameter list, a DSLNIC macro with TYPE=REQ, NAME=DSLNRTCP and BUF=, is required to request execution of a central service.

label
A unique statement label according to assembler language conventions.

TYPE=INSTRUCT
Instruct the MERVA ESA remote task central service program DSLNRTCP to send an instruction to the remote task identified by the ID= operand.

ID The name identifying the receiving task, or a general register containing the address of an 8-byte field containing the name. If a literal is used, it may be up to 8 characters long.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the DSLRTC parameter list or a general register containing the address of the parameter list. The default is (E,(1)).

Mapping the Parameter List for DSLRTC

The DSLRTC macro may be used to generate a storage description of the DSLRTC parameter list.

Name	Operator	Operands
[*label*]	**DSLRTC**	**MF=L**

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

DSLSRV: Defining Service Requests

The DSLSRV macro serves the following purposes:

- Call the service program DSLSRVP
- Map the parameter list of DSLSRVP

Calling DSLSRVP

The MERVA ESA service program DSLSRVP is used to carry out environment related functions, such as acquiring and freeing storage, loading and deleting program modules, posting ECBs, setting wait intervals, taking dumps or requesting the system date or time.

Name	Operator	Operands

[*label*]	**DSLSRV**	**TYPE=**{**GETMAIN** }
		{**GETMAINA**}
		{**GETMAINS**}
		{**FREEMAIN**}
		{**LOAD** }
		{**RELEASE** }
		{**DELETE** }
		{**WAIT** }
		{**POST** }
		{**DUMP** }
		{**SNAP** }
		{**DATETIME**}
		{**ENQ** }
		{**DEQ** }
		{**COMMIT** }
		{**ROLLBACK**}
		[**,ADSTOR=**{*addr*|(*r*)}]
		[**,DATMASK=**{**'YYDDD '**|'*cccccccc*'|*addr*|(*r*)}]
		[**,DUMPID=**{**127**|*addr*|(*r*)}]
		[,{**ECBLIST**|**ECB**}={*addr*|(*r*)}]
		[**,ERROR=**{*addr*}]
		[**,INCHAR=**{**00**|*xx*}]
		[**,MF=E**]
		[**,MODULE=**{'*cccccccc*'|*addr*|(*r*)}]
		[**,PREFIX=**{**SRV**|*ccc*}]
		[**,QNAME=**{'*cccccccc*'|*addr*|(*r*)}]
		[**,RNAME=**{*addr*|(*r*)}]
		[**,SIZE=**{*nnnnn*|*addr*|(*r*)}]
		[**,TIMMASK=**{**'HH:MM:SS'**|'*cccccccc*'|*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that general register 13 points to a usable save area, that DSLCOM is addressable, and that the field COMSRVPA contains the entry address of the service program DSLSRVP.

For the address parameters in this macro, the following notations are possible:

addr A symbolic label.

(r) A general register containing the address.

label
A unique statement label according to assembler language conventions.

TYPE

The type of service request.

GETMAIN	Acquire storage. The storage address is returned in the field SRVSADDR.
GETMAINA	Acquire storage above the line. The storage is acquired above the 16MB line. The calling program must run in AMODE 31. The storage address is returned in the field SRVSADDR.
GETMAINS	For CICS only, shared storage is acquired. The storage is above the line. The calling program must run in AMODE 31. The storage address is returned in the field SRVSADDR.
FREEMAIN	Release storage.
LOAD	Load a module. The entry point is returned in the field SRVENTRY. The module size is returned in the field SRVSIZE.
RELEASE	Release a module (same as DELETE).
DELETE	Delete a module (same as RELEASE).
WAIT	Wait for completion of an event.
POST	Post completion of an event.
DUMP	Dump a storage area. For CICS the dump is written to the active CICS dump data set. For IMS or MVS Batch programs the dump is written to the DSLSNAP data set.
SNAP	Provide SNAP dump of a storage area. For CICS the dump is written to the active CICS dump data set. For IMS or MVS Batch programs the dump is written to the DSLSNAP data set.
DATETIME	Get date and time. The edited date is returned in the field SRVDATEX. The edited time is returned in the field SRVTIMEX.
ENQ	Enqueue upon a resource. The task issuing this request will get exclusive use of the resource. The task is suspended until the resource is available, when another task has already issued an ENQ requests for the same resource. This request type is not available for programs running under VSE batch.
DEQ	Dequeue for a resource. The task releases the exclusive use of a resource previously enqueued upon using the request type ENQ. When issuing the DEQ request, the resource to be dequeued must be identified by the same method used when enqueuing upon the resource. This request type is not available for programs running under VSE batch.
COMMIT	Perform a COMMIT call. When running under CICS, a CICS SYNCPOINT call is executed. When running under control of DSLNUC in an IMS BMP, an IMS checkpoint call (CHKP) is executed. This request type is not available for programs running under MVS batch or VSE batch.
ROLLBACK	Perform a ROLLBACK call. When running under CICS, a CICS SYNCPOINT ROLLBACK call is executed. When running under control of DSLNUC in an IMS BMP, an IMS rollback call (ROLB) is executed. This request type is not available for programs running under MVS batch or VSE batch.

ADSTOR
Either the label of a storage area, or a general register containing its address.

DATMASK
For TYPE=DATETIME only, one of the following:

- A date mask (in single quotes)
- The label of a field containing the date mask
- A general register containing the address of the date mask field

A date mask can be up to 8 characters long, and contain any combination of the following elements:

YYYY	The 4-digit year (for example, 2001)
YY	The last two digits of the year (00 ... 99)
MM	The month (01 ... 12)
MMM	The name of the month (JAN ... DEC)
DD	The day of the month (01 ... 31)
DDD	The day of the year (001 ... 366)

All other characters or combinations of characters are used as separators. The position of the element determines the position of the corresponding data in the character string. Each element can be used only once. The default mask is '*YYDDD*'.

DUMPID
Either the dump identification as a decimal number, or the label of a halfword or a general register containing the hexadecimal dump identification. A value from 1 to 127 (X'01' to X'7F') can be specified. The default is 127. The parameter list of DSLSRVP (in DSLCOM) shows the dump identifiers used by MERVA ESA.

ECB
Either the label of a fullword to be used as an event control block (ECB), or a general register containing its address. ECB and ECBLIST are mutually exclusive.

ECBLIST
Either a label at the beginning of a list of ECB addresses, or a general register containing the address of the list. ECB and ECBLIST are mutually exclusive.

ERROR
A symbolic address where the program calling DSLSRVP should continue if the return code in register 15 is not zero.

After an error, the DSLSRVP parameter list in DSLCOM contains the return code at label SRVRC and the reason code at label SRVRSNCD.

INCHAR
For TYPE=GETMAIN only, a 1-byte hexadecimal value used to initialize the storage obtained. The default is 00.

MF
The format of the macro. It must be E for execute form. The default is E.

MODULE
For LOAD, RELEASE and DELETE only, specifies either the name of the

module (enclosed in single quotes), or an 8-byte field containing the name of the module, or a general register containing the address of the field that contains the name of the module.

PREFIX
The prefix for the labels of the fields in the parameter list. The prefix must be 3 characters long and must contain only characters that result in valid assembler language symbols. The default is SRV.

QNAME
For ENQ, and DEQ only, specifies either the name of the queue (enclosed in single quotes), or an 8-byte field containing the name of the queue, or a general register containing the address of the field that contains the name of the queue.

The QNAME is only used in MVS batch environment. When the QNAME is not specified, the first 8 bytes of the character string specified with the RNAME parameter are taken.

RNAME
For ENQ, and DEQ only, specifies either the label of a storage area or a general register containing the address. The storage area contains a character string which may be up to 255 bytes in length. The minimum length is 8 bytes. This character string represents the resource. The length of the resource name must be specified using the SIZE= parameter.

SIZE
Either the size of a storage area as a decimal number, or the label of a fullword or a general register containing the size:

- For TYPE=GETMAIN, GETMAINA, ENQ, DEQ, or SNAP, the SIZE parameter is required.
- For TYPE=RELEASE or DELETE, the SIZE parameter is required only when the calling program is running in a VSE batch environment prior to VSE/ESA 1.3.
- For TYPE=FREEMAIN when the SIZE parameter is omitted or SIZE=0 is specified, the size is taken from the first halfword of the storage area specified by ADSTOR.
- For TYPE=ENQ or DEQ, the SIZE parameter specifies the length of the character string used as a resource name. This resource name must have a length between 1 and 255 bytes and is specified with the RNAME parameter.

In a CICS transaction, the maximum size depends on the CICS version or release.

TIMMASK
For TYPE=DATETIME only, one of the following:

- A time mask (in single quotes)
- The label of a field containing the time mask
- A general register containing the address of the time mask field

A time mask can be up to 8 characters long, and contain any combination of the following elements:

HH	The hour (00 ... 23).
MM	The minute (00 ... 59).
SS	The second (00 ... 59).

All other characters or combinations of characters are used as separators. The position of the element determines the position of the corresponding data in the character string. Each element can be used only once. The default mask is '*HH:MM:SS*'.

Mapping the Parameter List of DSLSRVP

The *list* form of the macro DSLSRV may be used to generate a storage description of the DSLSRVP parameter list.

The DSLSRVP parameter list is also included in the MERVA ESA communication area DSLCOM. When calling the program DSLSRVP, DSLCOM is required.

In any program, the macros DSLSRV MF=L and DSLCOM are mutually exclusive because they generate duplicate field names unless the PREFIX parameter is used.

Name	Operator	Operands
[*label*]	**DSLSRV**	**MF=L** [**,DSECT=**{**NO**|**YES**}] [**,PREFIX=**{**SRV**|*ccc*}]

Programming Notes:

label
: A unique statement label according to assembler language conventions.

MF
: The format of the macro. It must be L for list form.

DSECT
: Whether the parameter list is to be mapped as a DSECT. The default is NO.

PREFIX
: The prefix for the labels of the fields in the parameter list. The prefix must be 3 characters long and must contain only characters that result in valid assembler language symbols. The default is SRV.

Parameter Requirements

Figure 2 on page 224 shows for each request type which parameters are mandatory (M) and which are optional (O). Parameters with no indication are ignored for the specific request type. The COMMIT and ROLLBACK request types have no further parameters except TYPE.

Parameter	GETMAIN/GETMAINA/GETMAINS	FREEMAIN	LOAD	RELEASE	DELETE	WAIT	POST	ENQ	DEQ	DUMP	SNAP	DATETIME	MF=L
TYPE	M	M	M	M	M	M	M	M	M	M	M	M	
MODULE			M	M	M								
ADSTOR		M		1	1						M		
ECB						2	M						
ECBLIST						2							
SIZE	M	O		1	1			M	M		M		
DUMPID										3	3		
INCHAR	O												
DATMASK												O	
TIMMASK												O	
QNAME								O	O				
RNAME								M	M				
ERROR	O	O	O	O	O	O	O	O	O	O	O	O	
PREFIX	O	O	O	O	O	O	O	O	O	O	O	O	O
DSECT													O
MF	O	O	O	O	O	O	O	O	O	O	O	O	M

Figure 2. DSLSRV Summary of Parameter Requirements

Note:

M	Stands for mandatory
O	Stands for optional
1	Also optional
2	Mutually exclusive, mandatory
3	Not required in VSE batch programs

DSLSYSP: Defining the Operating System Environment

Setting System Globals

This macro evaluates the Assembler SYSPARM parameter and sets the system globals during the assembly of MERVA ESA programs and MERVA ESA application programs.

Name	Operator	Operands
	DSLSYSP	*cccccccc* [,**AMODE**={**ANY**|**31**|**24**}] [,**RMODE**={**24**|**ANY**}]

Programming Notes: The following statement defines the global parameters and must precede the DSLSYSP macro:

```
COPY  DSLSYSET
```

In MVS the DSLSYSP macro sets SPLEVEL SET=2. The DSLSYSP macro must precede any CSECT, DSECT, or EQU statements. In VSE, a CATALR statement is punched for the object deck.

cccccccc
The name of the program.

AMODE
The address mode of the control section. An assembler AMODE instruction is generated during the assembly of the MERVA ESA program. The default is ANY.

RMODE
The residency mode of the control section. An assembler RMODE instruction is generated during the assembly of the MERVA ESA program. The default is 24.

SYSPARM Requirements

SYSPARM must be specified as follows:

MVS	For MERVA ESA running under MVS.
IMSMVS	For MERVA ESA running in IMS/ESA, which always runs under MVS.
CICSMVS	For MERVA ESA running in CICS/MVS®.
CICSVSE	For MERVA ESA running in CICS/VSE.

SYSPARM is specified during assembly of a MERVA ESA program as follows: In MVS, SYSPARM is specified in the Job Control EXEC statement PARM parameter:

```
//      ...,PARM=(......'SYSPARM(MVS)',......)
```

In VSE, SYSPARM is specified in the Job Control OPTION statement:

```
// OPTION SYSPARM='CICSVSE',.....
```

DSLTFD: Defining the Terminal Feature Definition Table

The MERVA ESA terminal feature definition is used by MERVA ESA running in IMS to define the terminal features of screen and printer terminals, and in all MERVA ESA to define the features of the system printer for the DSLSDY program and the terminal printers for the DSLHCP program.

The DSLTFD macro is used to:

- Generate a terminal feature definition table
- Map the terminal feature definition table entry

The terminal feature definition table consists of a sequence of DSLTFD macros. The first instruction should have a label; if it does not, the default label DSLTFDT is used. The last statement of the table must be the END statement.

No assembler language statements except SPACE, EJECT, END, and possibly PRINT must be used in the table definition.

If errors are detected in a specific DSLTFD macro, appropriate MNOTEs are provided.

MERVA ESA provides a sample terminal feature definition table DSLTFDT. The name of the terminal feature definition table to be used in the system has to be specified in the MERVA ESA customizing parameter definition DSLPRM.

Generating a Terminal Feature Definition Table

Name	Operator	Operands
[*label*]	**DSLTFD**	**TYPE=**{**INITIAL**|**FINAL**}

Programming Notes:

label
: The name of the CSECT generated. It must be a unique label according to the assembler language naming conventions. The label must appear in the first DSLTFD statement.

TYPE
: TYPE=INITIAL must be the first terminal feature definition table definition. If the first statement does not contain a label, DSLTFDT is used. In the second and following calls of DSLFDT, the label is disregarded. TYPE=FINAL must be the last terminal feature definition table definition statement.

Generating a Terminal Definition Entry

For MERVA ESA running under IMS the terminal characteristics as specified in the terminal feature definition table are used to prepare a physical page to be sent to the terminal.

For MERVA ESA running under CICS the terminal characteristics are determined by the definition in the CICS terminal definition.

The PAGESIZ parameter is used to determine the page (paper) size for printer terminals. This size specification is interpreted by the MERVA ESA MFS formatting programs as the number of print lines on a page, that is, from perforation to perforation for hardcopy terminals or the maximum number of lines which can be printed between page ejects for system printer devices.

Name	Operator	Operands
[*label*]	**DSLTFD**	[**TYPE=ENTRY**] **,BUFSIZE=**{*nnnnn*} **,LTERM=***cccccccc* [**,FEATURE=**([**COLOR**][**,EXTHIL**][**,SOSI**])] [**,PAGESIZ=**({**24**|*nn*},{**80**|*nn*})] [**,TERMTYP=***typecode*] [**,WRC=**{**WRCEW**|*wrc*}]

Programming Notes:

label
For TYPE=ENTRY calls of DSLTFD, the label is disregarded.

TYPE
TYPE=ENTRY specifies that a terminal definition entry is to be generated.

BUFSIZE
The size of the terminal buffer. The value can be a number from 256 to 16000. The default value is calculated as follows:

- For screen terminals defined with:
  ```
  FEATURE=COLOR
  FEATURE=EXTHIL
  ```
  the buffer is 125 multiplied by the number of rows from PAGESIZ.
- For all other screen terminals, the buffer is 100 multiplied by the number of rows from PAGESIZ.

When the default value exceeds 4096, it is reduced to 4096. The buffer size is rounded up to a multiple of 8.

For printer terminals BUFSIZE is mandatory. For SCS printer terminals the BUFSIZE must be specified according to the NCP/VTAM RUSIZE parameter as a decimal value.

Note: When defining screen terminals in the IMS nucleus, the OUTBUF parameter of IMS must be 500 greater than the BUFSIZE parameter of the DSLTFD macro.

LTERM
The logical terminal name. The name can be from 1 to 8 characters long. This name corresponds to an LTERM name defined in the CICS terminal definition or TERMINAL macro of the IMS nucleus generation. For system printer definitions this name must be DSLSDSY.

cccccccc
A character string of 1 to 8 characters according to the assembler language naming conventions.

FEATURE
The features of a screen terminal. COLOR specifies that color attributes can be sent to the terminal. EXTHIL specifies that extended highlighting attributes can

be sent to the terminal. SOSI specifies that double-byte character set (DBCS) data is supported. In EBCDIC DBCS data is enclosed by a shift-out character X'0E' and a shift-in character X'0F'.

PAGESIZ
The number of rows and columns of a physical page on a screen terminal. For printer terminals the first subparameter (rows) is the length of a logical page. The specified value can be a number from 1 to 9999. The second subparameter is the width of the *device* (columns). The specified value can be a number from 1 to 132. The default value for rows is 24, for columns 80.

TERMTYP
The type of terminal. Allowed values for the parameter are:

3270	327x Screen terminal or other terminals in 327x emulation mode.
3270P	Terminal printer.
SCSP	SCS printer.
SCSPSIM	SCS printer with simplified data stream. There are no Set Vertical Format (SVF) controls used in the generated data stream. One or more New Line (NL) controls are used instead.
SYSP	System printer.

WRC
Which write command should be used. For screen terminals in MERVA ESA running under IMS you can specify:

WRCEW	This stands for “WRite Command Erase Write”. Use it when using the normal screen size. This is the default.
WRCEWA	This stands for “WRite Command Erase Write Alternate”. Use it when using the alternate screen size.

Mapping a Terminal Feature Definition Table Entry

Name	Operator	Operands
[*label*]	**DSLTFD**	**TYPE=MAP** [**,DSECT=**{**NO**|**YES**}] [**,PREFIX=**{**TFD**|*ccc*}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The map of the terminal-feature definition table entry is to be generated (TYPE=MAP). If TYPE=MAP is specified, all other parameters except PREFIX and DSECT are ignored.

DSECT
Whether the map is to be defined as a DSECT. The default is NO.

PREFIX
The 3-character prefix of the variable names of the map. The default is TFD.

DSLTIM: Defining the Timer Service

The DSLTIM macro serves the following purposes:

- Set or cancel a timer request
- Map the DSLTIMP parameter list

Calling the Timer Program

The timer program DSLTIMP is used to request timer services from MERVA ESA.

The calling program requests that MERVA ESA should post an ECB after a specified interval or at a specified time of day. The calling program must supply an ECB, an interval or expiry time, and a unique name to identify the request.

Any number of timer requests may be issued concurrently from the same calling program, but the ECB and name must be unique for each request.

The DSLTIM macro can be used only by programs that are linked to DSLNUC through the nucleus program table DSLNPTT.

Name	Operator	Operands
[*label*]	**DSLTIM**	**TYPE=**{**SET**|**CANCEL**} **,MF=(E,**{*addr*|(*r*)}) **,NAME=**{'*cccccccc*'|*addr*|(*r*)} [**,ECB=**{*addr*|(*r*)}] [**,EXTIM=**{'*hhmmss*'|*addr*|(*r*)}] [**,ITVL=**{'*hhmmss*'|*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that general register 13 points to a usable save area and that DSLCOM is addressable.

For the address parameters in this macro, the following notations are possible:

addr A symbolic label.

(r) A general register containing the address.

label
A unique statement label according to assembler language conventions.

TYPE
The MERVA ESA timer function to be performed:

SET Set up a timer request. A MERVA ESA timer request is created. The request is identified by the name specified in the NAME parameter. If an unexpired request already exists with the same name, it is canceled and replaced by the new request.

CANCEL Cancel a previous timer request. An outstanding timer request is canceled. The request must be identified by the same name as used in the SET request for that ECB. If the ECB is posted by MERVA ESA, the request is automatically canceled at that time.

MF
The format of the macro.

The first parameter must be E for execute form. The second parameter specifies the label of the DSLTIMP parameter list or a general register containing the address of the parameter list.

NAME
A unique name for identification of the timer request. The value can be an 8-character literal, or the label of an 8-byte field containing the name, or a general register containing the address of the name field.

The characters DSL and DWS in the first 3 positions of the request name are reserved for IBM internal use.

ECB
The label of a fullword to be used as an ECB or a general register containing the address of the ECB. The ECB is required for TYPE=SET, it is ignored for TYPE=CANCEL.

EXTIM
The time of day when the ECB is to be posted, in the format HHMMSS, for TYPE=SET.

The parameter may specify a literal (enclosed in single quotes), or the label of a 6-byte field that contains the expiry time in display format, or a general register that points to the field containing the expiry time.

The maximum value for the time of day is the current system time plus 15 hours. The parameters ITVL and EXTIM are mutually exclusive.

ITVL
The timer wait interval in the format HHMMSS for TYPE=SET.

The parameter may specify a literal (enclosed in single quotes), or the label of a 6-byte field that contains the timer wait interval in display format, or a general register that points to the field containing the timer wait interval.

The maximum value for the wait interval is 150000. The parameters ITVL and EXTIM are mutually exclusive.

Mapping the Parameter List of DSLTIMP

The *list form* of the DSLTIM macro is used to generate a storage description of the DSLTIMP parameter list.

Name	Operator	Operands
[*label*]	**DSLTIM**	**MF=L**

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

DSLTRA: Defining the Processing Trace

The DSLTRA macro has two functions, to:

- Call the trace program DSLTRAP
- Map the parameter list of DSLTRAP

Calling the Trace Program DSLTRAP

The MERVA ESA trace program DSLTRAP records significant processing steps in a main storage trace table, and may record the table in the MERVA ESA journal data set.

The trace facility is started by the TRACE parameter of the MERVA ESA customizing macro DSLPARM. The DSLTRA macro calls the trace program.

The trace program uses the nucleus trace table for all programs that are linked to the MERVA ESA nucleus. For programs not linked to the nucleus, the trace program establishes individual trace tables and journal records.

Name	Operator	Operands
[*label*]	**DSLTRA**	[**,DATA=**{'*literal*'|*addr*|(*r*)}]
		[**,ID1=**{*nn*|C'*c*'|X'*nn*'|*addr*|(*r*)}]
		[**,ID2=**{*nn*|C'*c*'|X'*nn*'|*addr*|(*r*)}]
		[**,MF=E**]
		[**,SESSID=**{'*nnnn*'|*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that general register 13 points to a usable save area and that DSLCOM is addressable.

If a parameter is not specified, the corresponding field of the DSLTRAP parameter list remains unchanged. Programs not linked to DSLNUC must allocate a MERVA ESA ICB and provide the DSLNIC parameter list address in DSLCOM, to allow DSLTRAP to write the trace table to the MERVA ESA journal data set (see "DSLNIC: Defining the Intertask Communication" on page 154).

label
A unique statement label according to assembler language conventions.

DATA
The data to be added to the trace table. The data may be a literal up to 24 bytes long enclosed in single quotes, or the label of a 24-byte field that contains the data to be recorded, or a general register containing the address of that field.

ID1
A 1-byte code to identify the caller of the trace program. The code may be a single character (C'*c*'), or a decimal number (*nn*), or a hexadecimal number (X'*nn*'), or the label of an equate statement, or a general register containing a 1-byte hexadecimal code.

Program identifiers used by MERVA ESA are defined in the DSLTRAP parameter list.

Note: In programs not linked to the MERVA ESA nucleus, the last trace request must be ID1=TRAIDTRM to ensure that the last journal record is written.

ID2
A 1-byte extension of the caller's identifier. It may be a single character (C'*c*'), or a decimal number (*nn*), or a hexadecimal number (X'*nn*'), or the label of an equate statement, or a general register containing a 1-byte hexadecimal code.

The extension may be used to identify processing steps within the caller's logic.

MF
The format of the macro. It must be E for execute form. The default is E.

SESSID
A 2-byte session identifier. The parameter may be a decimal number enclosed in single quotes, or a field or general register containing the 2-byte code. The session identifier may be used to identify related processing steps when the caller is servicing more than one user concurrently.

Mapping the Parameter and Substitution List

The *list* form of the macro DSLTRA may be used to generate a storage description of the DSLTRAP parameter list.

Name	Operator	Operands
[*label*]	**DSLTRA**	**MF=L**

Programming Notes: The DSLTRAP parameter list is included in the MERVA ESA communication area DSLCOM. It is not necessary to map the parameter list again when calling the trace program.

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

DSLTSV: Defining the TOF Supervisor Interface

Calling the MERVA ESA TOF Supervisor

The MERVA ESA TOF Supervisor Interface DSLTOFSV is implemented both as a macro interface and as a call interface.

In both cases, a DSLTOFSV parameter list is created to define the TOF request. The parameter list contains the field name and indices of the last field accessed. This defines the current position in the TOF.

The DSLTOFSV request provides positioning information to change the TOF position to the ‘next’ target. One-time modifiers may also be included, which change the basic TOF positioning process.

Positioning in the TOF is done as follows: The parameters of the current TOF position are replaced by any parameters specified for the ‘next’ target. This gives an intermediate position which may be further revised by the one-time modifiers. (This intermediate position is called the ‘actual’ position throughout this macro description.)

The result is a new TOF position, which is called the ‘field reference’. It consists of the parameters: nesting identifier NI, field group FG, repeatable-sequence occurrence number RS, field name FN and data area index DA or option.

Note: In case of a field in a nested repeatable sequence, a repeatable-sequence occurrence number is required for each RS nesting level.

In the parameter list, the current position is calculated by the TOF supervisor. The parameters of the ‘next’ position are not changed unless specified by the request. The one-time modifiers are reset with each request.

Note: TOF position parameters specified for the ‘next’ target are kept in the DSLTOFSV parameter list using a DSLTOFSV request. To make use of the current position returned from the *last successful DSLTOFSV request* the ‘next’ position parameters must be cleared *explicitly*.

Calling DSLTOFSV Using the Macro Interface

Use the DSLTSV macro to create the parameter list and to call the MERVA ESA TOF Supervisor DSLTOFSV. Specifications in the parameter list remain in effect until they are changed by a new specification.

Name	Operator	Operands

[*label*]	**DSLTSV**	**TYPE=**{**ACCESS** }
		{**ADDDA** }
		{**ADDNI** }
		{**CHECK** }
		{**COMPRESS**}
		{**DELETE** }
		{**EXPAND** }
		{**FREE** }
		{**INIT** }
		{**JOIN** }
		{**MERGE** }
		{**READ** }
		{**TOFNEW** }
		{**WRITE** }
		[**,BUFFER=**{*addr*|(*r*)}]

Name	Operator	Operands
		[,**DAINDEX=**{*nn*|*addr*|(*r*)}]
		[,**FDGPIND=**{*nn*|*addr*|(*r*)}]
		[,**FDNAM=**{*addr*|(*r*)}]
		[,**FMODIF=**{**CHECK** }]
		{**DATA** }
		{**DECHECK**}
		{**DEEDIT** }
		{**DELAD** }
		{**DELALL** }
		{**DELAX** }
		{**DELDA** }
		{**DELDAGR**}
		{**DELDAIN**}
		{**DELDAVR**}
		{**DELFN** }
		{**DELNI** }
		{**DELRS** }
		{**EDIT** }
		{**EDITCHK**}
		{**FDSCRPT**}
		{**IGNORE** }
		{**INFO** }
		{**OPTLIST**}
		{((*r*)) }
		[,**MF=(E,**{*addr*|(*r*)})]
		[,**MFSPS=**{*addr*|(*r*)}]
		[,**MODIF=**{(*mod*,...,*mod*)|((*r*))}]
		[,**NESTID=**{*nn*|*addr*|(*r*)}]
		[,**OPTION=**{**YES** }]
		{**NO** }
		{**AFTER** }
		{**BEFORE**}
		{**DATA** }
		{(*r*) }
		[,**PREFIX=**{**TSV**|*ccc*}]
		[,**RSEXT=**{*addr*|(*r*)}]
		[,**RSINDEX=**{*nn*|*addr*|(*r*)|*(rs1,rs2,...,rsn)*}]
		[,**TOF=**{*addr*|(*r*)}]
		[,**WORK=**{*addr*|(*r*)}]

Programming Notes: Before using this macro, you must ensure that DSLCOM is addressable and that the field COMTSVA contains the entry address of the TOF supervisor interface DSLTOFSV.

The TOF must be initialized using a DSLTOFSV TYPE=TOFNEW request before other TOF service requests are issued.

The one-time modifiers (MODIF) and the function modifier (FMODIF) are reset to blanks when not specified; other parameters remain unchanged.

label
: A unique statement label according to assembler language conventions.

TYPE
: The TOF service function to be performed.

The parameter is coded as a value. The values together with their associated function modifiers and options are defined in Table 1 on page 240.

The code is stored in the field TSVPFTYP of the TOF parameter list.

BUFFER
The label of a buffer supplied by the caller, or a general register containing the address of the buffer

- For TYPE=WRITE and TYPE=ADDDA, this buffer contains the data to be written to a TOF data area.
- For TYPE=READ, this buffer receives the specified data from the TOF.
- For TYPE=COMPRESS, the buffer will contain the compressed TOF.
- For TYPE=MERGE, the buffer must contain a valid TOF to be merged into another valid TOF which is specified by the parameter TOF.
- For TYPE=INIT, the buffer contains the field entry from the MCB for the field specified by the field reference. (Refer to the *MERVA for ESA Customization Guide*.)

The buffer is filled by:

- The TOF supervisor if data is to be read from the TOF
- The caller if data is to be written to the TOF

The buffer must have the standard MERVA ESA format, which is described in the *MERVA for ESA Customization Guide*. The size of the buffer depends on the data that is to be stored in it.

DAINDEX
The data area index of the 'next' target. It is a number between 0 and 32767, or the label of a halfword or a general register containing a value from 0 to 32767. A value of 0 does not change the current data area index.

The data area index is stored in the field TSVPNEDA of the TOF parameter list.

FDGPIND
The field group index of the 'next' target. It is a number between 0 and 255, or the label of a 1-byte field or a general register containing a value from 0 to 255. A value of 0 does not change the current field group index.

The field group index is stored in the field TSVPNEFG of the TOF parameter list.

FDNAM
The field name of the 'next' target.

The parameter may be the label of an 8-character field containing the field name, or a general register containing the address of the 8-character field. A value of blanks does not change the current field name.

The field name is stored in the field TSVPNEFN of the TOF parameter list.

FMODIF
A change to the basic TOF function requested by the TYPE parameter. The parameter may be one of the FMODIF values shown in Table 1 on page 240, or a general register that contains the address of a 4-byte field which contains the code shown in the table. The general register must be enclosed in double parentheses.

The code is stored in the field TSVFCMO in the TOF parameter list.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the DSLTOFSV parameter list or a general register containing the address of the parameter list. The default is (E,TSVPARL).

MFSPS
Permanent storage used by Message Format Services (DSLMMFS).

The label of the MFS permanent storage or a general register containing the address of the MFS permanent storage. A storage definition can be generated using the macro DSLMFS MF=PS.

The MFS permanent storage address is stored in the field TSVPENVR in the TOF parameter list.

MODIF
Sets the one-time modifiers.

The parameter may be a list of values, or a general register containing the address of a 5-byte list of modifier codes. The general register must be enclosed in double parentheses.

The list of values may contain from 1 to 5 modifiers. Each value is associated with one of the positioning parameters. The MODIF values are defined below.

For register notation, a modifier code is defined for each value in the table. Each code is a single character; a blank shows no modifier. The list of modifier codes is positional in the following order: nesting indicator, field group, repeatable sequence occurrence, field name, data area.

One modifier may be specified for each positioning parameter. Only logical combinations of modifiers are allowed. This is checked by the TOF Supervisor and a return code is supplied for an illogical combination.

Modifier	Description
NEXTNI	Sets value 'N' in TSVPMONI. The actual nesting identifier is changed with the number of the next logical nesting identifier. This is not necessarily the old nesting identifier plus 1. The actual field group, occurrence number and data area index are changed to the value 1. The actual field name is changed to the first field name of the new nesting identifier.
FIRSTNI	Sets value 'F' in TSVPMONI. The actual nesting identifier is changed with the number of the first logical nesting identifier (NI=0). The actual field group, occurrence number and data area index are changed to the value 1. The actual field name is changed to the first field name of the new nesting identifier.
LASTNI	Sets value 'L' in TSVPMONI. The actual nesting identifier is changed with the number of the last logical nesting identifier in the TOF. The actual field group, occurrence number and data area index are changed to the value 1. The actual field name is changed to the first field name of the new nesting identifier.
PRECNI	Sets value 'P' in TSVPMONI. The actual nesting identifier is changed with the number of the preceding logical nesting identifier. The actual field group and field name are changed to

the value of the exit field on the preceding identifier. The occurrence number and data area index are changed to the value 1.

NEXTRS Sets value 'N' in TSVPMORS (repeatable sequence occurrence). The actual occurrence number is changed with the number of the next occurrence of the actual nesting identifier and field group. The actual data area index is changed to the value 1. The actual field name is changed to the first field name of the repeatable sequence.

FIRSTRS Sets value 'F' in TSVPMORS (repeatable sequence occurrence). The actual occurrence number is changed with the number of the first occurrence (RS=1) of the actual nesting identifier and field group. The actual data area index is changed to the value 1. The actual field name is changed to the first field name of the repeatable sequence.

LASTRS Sets value 'L' in TSVPMORS (repeatable sequence occurrence). The actual occurrence number is changed with the number of the last occurrence of the actual nesting identifier and field group. The actual data area index is changed to the value 1. The actual field name is changed to the first field name of the repeatable sequence.

RSEXT Sets value 'X' in TSVPMORS (repeatable sequence occurrence). This modifier indicates that a repeatable-sequence extension parameter list (RSEXT) is used to supply occurrence numbers or modifiers to access fields in repeatable sequences. In this case the RS parameters are supplied either as a list with the RSINDEX operand or directly in the RSEXT supplied.

NEXTFN Sets value 'N' in TSVPMOFN (field name). The actual field name is changed with the name of the next field. The actual data area index is changed to the value 1. The actual repeatable-sequence occurrence number is not changed. If the nesting identifier and the field group, or both, is changed by this process, a return and reason code is supplied by DSLTOFSV.

NEXTFD Sets value 'S' in TSVPMOFN (field name). The actual field name is changed with the name of the next field. The actual data area index is changed to the value 1.

Note: The actual repeatable-sequence occurrence number may be changed if the actual or next field is in a repeatable sequence. When the actual field is the last in a repeatable sequence, the next field is the first in that sequence with the repeatable-sequence occurrence number incremented by one. If the nesting identifier and the field group, or both, is changed by this process, a return and reason code is supplied by DSLTOFSV.

FIRSTFN Sets value 'F' in TSVPMOFN (field name). The actual field name is changed with the name of the first field of the actual nesting identifier and field group. The actual data area index is changed to the value 1. The actual repeatable-sequence occurrence number is not changed.

LASTFN Sets value 'L' in TSVPMOFN (field name). The actual field name is changed with the name of the last field of the actual nesting identifier and field group. The actual data area index is changed to the value 1. The actual repeatable-sequence occurrence number is not changed.

FIRSTEQ Sets value 'E' in TSVPMOFN (field name). Used to search for a field with the name specified in parameter FDNAM of the DSLTSV macro, starting at the actual position.

The FDNAM parameter is not used in determining the actual position. All other next parameters are considered. Nesting identifier and field group modifiers are applied, then the FDNAM parameter is used to find the required field.

The actual positioning parameters are changed with the positioning parameters of the field found. The actual data area index is changed to the value 1.

Note: The actual repeatable-sequence occurrence number is not changed. If the nesting identifier or field group, or both, are changed by this process, a return and reason code is supplied by DSLTOFSV.

NEXTA Sets value 'A' in TSVPMOFN (field name). Used to search for a field with the name specified in parameter FDNAM of the DSLTSV macro, starting with the field after the actual position. The actual positioning parameters are changed with the positioning parameters of the field found. The actual data area index is changed to the value 1.

Note: The actual repeatable-sequence occurrence number is not changed. If the nesting identifier and/or field group is changed by this process, a return and reason code is supplied by DSLTOFSV.

VFIRST Sets value 'V' in TSVPMOFN (field name). Used to search for a field with the name specified in parameter FDNAM of the DSLTSV macro starting at the very first field in the TOF. The actual positioning parameters are changed with the positioning parameters of the field found. The actual data area index is changed to the value 1.

Note: The actual repeatable-sequence occurrence number is not changed. If the nesting identifier and/or field group is changed by this process, a return and reason code is supplied by DSLTOFSV.

NEXTDA Sets value 'N' in TSVPMODA (data area index). The actual data area index is changed with the number of the next data area. The other positioning parameters are not modified.

FIRSTDA Sets value 'F' in TSVPMODA (data area index). The actual data area index is changed with the number of the first data area (DA=1). The other positioning parameters are not modified.

LASTDA Sets value 'L' in TSVPMODA (data area index). The actual data area index is changed with the number of the last data area. The other positioning parameters are not modified.

NESTID
The nesting identifier of the next target. It is a number between 0 and 255, or the label of a 1-byte field or general register containing a value from 0 to 255. The nesting identifier is stored in the field TSVPNENI of the TOF parameter list.

Note: If the value of NESTID is 0, the indicator TSVPNIRZ in the TOF parameter list is set to 1. Nesting identifier NI=0 is accessed. If NESTID is omitted and TSVPNENI already contains the value 0, TSVPNIRZ is set to 0 and the current nesting identifier is not changed.

OPTION
A further change to the basic TOF function requested by the TYPE parameter. The parameter may be one of the OPTION values shown in Table 1 on page 240, or a general register that contains the equivalent character shown in the table. The equivalent character is stored in the field TSVPNEOM in the TOF parameter list.

PREFIX
The first 3 characters of the field names to be generated in the DSLTOFSV parameter list. The default is TSV.

RSEXT
The label of the repeatable-sequence extension parameter list (RSEXT), or a general register containing the address of the RSEXT.

The RSEXT has the MERVA ESA standard buffer format and contains the parameters required to access fields in repeatable sequences, nested in repeatable sequences.

The address of the RSEXT is stored in the field TSVPRSXA in the TOF parameter list.

RSINDEX
The repeatable-sequence occurrence number of the next target or a list of specifications for fields within nested repeatable sequences. If exactly one index is specified, it must be a number between 0 and 32767 or a label of a halfword or a general register containing a value from 0 to 32767. A value of 0 does not change the current occurrence number.

The repeatable-sequence occurrence number is stored in the field TSVPNERS of the TOF parameter list. When a list of repeatable sequences is specified, each item must be a number between 0 and 32767 or a label of a halfword or a general register containing a value from 0 to 32767, or one of the modifiers FIRSTRS, NEXTRS, or LASTRS. If a list is specified, the modifier RSEXT must be specified also.

TOF
The label of the TOF, or a general register containing the address of the TOF. The address of the TOF is stored in the field TSVPADDR in the TOF parameter list.

Note: For TYPE=MERGE, this TOF must have enough free space to receive all the data contained in the TOF specified by the parameter BUFFER.

WORK
Internal working storage supplied by the caller and used by the TOF Supervisor.

The label of the working storage area, or a general register containing the address of the working storage area. The minimum working storage size is 2KB. A storage definition can be generated using the macro DSLTSV MF=TS.

The working storage address is stored in the field TSVPWORK in the TOF parameter list. The working storage must be in the standard MERVA ESA buffer format. That is, the first halfword contains the length of the working storage.

Table 1. TOF Service Function: TYPE values and Associated Functions Modifiers

Type	Function Description
ACCESS	Code = 'ACC ' in TSVPFTYP. Used to position the TOF to the specified field reference. If the field reference (nesting identifier, field group, RS occurrence number, field name) was found, the field reference is saved in the parameters of TSVPCURR and the parameter TSVPNIEX is filled with the nesting identifier of the next logical nesting identifier if an exit field of the current nesting identifier was accessed.
ADDDA	Code = 'ADDA' in TSVPFTYP. Used to add or append a new data area to a field in the TOF. A data area is appended, if the data area index required in the field reference is greater than an existing data area for this field in the TOF. A data area is inserted according to the option modifier, if the data area required in the field reference was found in the TOF. The field descriptor must be already initialized in the TOF. **FMODIF=DEEDIT** Value in TSVPFCMO='DEED'. De-editing and no checking of data referenced by TSVPBUFF. **FMODIF=CHECK** Value in TSVPFCMO='CHEK'. No de-editing, but checking of data referenced by TSVPBUFF. **FMODIF=DECHECK** Value in TSVPFCMO='DECH'. De-editing and checking of data referenced by TSVPBUFF. **OPTION=AFTER** Value in TSVPNEOM='A'. Insert data area after the data area found in TOF. **Note:** Values 'D' and 'O' are processed like value 'A'. **OPTION=BEFORE** Value in TSVPNEOM='B'. Insert data area before the data area found in TOF.
ADDNI	Code = 'ADNI' in TSVPFTYP. Used to add a new nesting identifier to the TOF. A new nesting identifier is initialized, an exit field is defined by the field reference evaluated from the current (TSVPCURR) and the changing parameters (TSVPMODS) with the field name referenced by FDNAM.
CHECK	Code = 'CHK ' in TSVPFTYP. Used to check of the contents of a field or a data area according to the function modifier FMODIF. A checking routine is called via DSLMFS if specified for this field, otherwise DSLTOFSV basic checking is performed. FMODIF=DATA, value in TSVPFCMO='DATA'. Checking of the data area specified in the field reference.
COMPRESS	Code = 'COMP' in TSVPFTYP. Used to create a compressed TOF into the buffer referenced by TSVPBUFF. Only QUEUE=YES fields are used for compression. There is no free space in the compressed TOF.

Table 1. TOF Service Function: TYPE values and Associated Functions Modifiers (continued)

Type	Function Description
DELETE	Code = 'DELE' in TSVPFTYP. Deletes data from the TOF according to the function modifier. **FMODIF=DELNI** Value in TSVPFCMO='DLNI'. The nesting identifier specified in the field reference and all nesting identifiers logically added to this nesting identifier are deleted. **FMODIF=DELRS** Value in TSVPFCMO='DLRS'. The repeatable-sequence occurrence specified in the field reference is deleted. **FMODIF=DELFN** Value in TSVPFCMO='DLFN'. The field specified in the field reference is deleted. **FMODIF=DELDA** Value in TSVPFCMO='DLDA'. The data area or option as specified in the field reference and option modifier OPTION is deleted. **FMODIF=DELDAGR** Value in TSVPFCMO='DLGR'. The data areas with an index greater than the data area index specified in the field reference are deleted from the field referenced **FMODIF=DELALL** Value in TSVPFCMO='DLAF'. All fields are deleted from the TOF. **FMODIF=DELAX** Value in TSVPFCMO='DLAX'. All fields except those on nesting identifier NI=0 and with the disposition PERM=YES are deleted from the TOF. **FMODIF=DELAD** Value in TSVPFCMO='DLAD'. All data areas of the field referenced by the field reference are deleted. **FMODIF=DELADAIN** Similar to DELDA, but the data area is marked for deletion only and is not removed from the data area chain. This preserves the data area index numbers of a data area chain. **FMODIF=DELDAVR** All data areas which are marked for deletion with a previous DELDAIN request are actually deleted. **OPTION=NO or OPTION=DATA** Value in TSVPNEOM='D'. Delete the data area (FMODIF=DELDA). **Note:** Values 'A' and 'B' are processed like value 'D'. **OPTION=YES** Value in TSVPNEOM='O'. Delete the option (FMODIF=DELDA).
EXPAND	Code = 'EXPA' in TSVPFTYP. Used to expand a field. The MFS expansion exit assigned to the field referenced by the field reference is called.
FREE	Code = 'FREE' in TSVPFTYP. Used to free the TOF Data Part allocated by the dynamic TOF function.
INIT	Code = 'INIT' in TSVPFTYP. Used to initialize the field referenced by the field reference in the TOF. An MCB entry must be supplied in TSVPBUFF by the calling program. (Refer to the *MERVA for ESA Customization Guide*.) A subfield can not be initialized.
JOIN	Code = 'JOIN' in TSVPFTYP. Used to create a compressed TOF into the buffer referenced by TSVPBUFF. All fields are used for compression. There is no free space in the joined TOF.
MERGE	Code = 'MERG' in TSVPFTYP. Used to merge the TOF with a TOF referenced by TSVPBUFF. A valid TOF must be supplied and its content is merged with the TOF referenced by TSVPADDR. This function is used when a TOF is moved from a queue buffer to the TOF.

Table 1. TOF Service Function: TYPE values and Associated Functions Modifiers (continued)

Type	Function Description
READ	Code = 'READ' in TSVPFTYP. Used to read data from the TOF according to the function modifier FMODIF supplied. **FMODIF=EDIT** Value in TSVPFCMO='EDIT'. Editing and no checking of the data area read from the TOF. **FMODIF=CHECK** Value in TSVPFCMO='CHEK'. No editing, but checking of the data area read from the TOF. **FMODIF=EDITCHK** Value in TSVPFCMO='EDCH'. Editing and checking of the data area read from the TOF. **FMODIF=OPTLIST** Value in TSVPFCMO='OPTL'. Read the option list of the field referenced. **FMODIF=FDSCRPT** Value in TSVPFCMO='FLDD'. Read the field descriptor of the field referenced. If the descriptor of a subfield is to be read, the field TSVPMFSR returns the value 35 to indicate that the DSECT of a subfield descriptor has to be used for interpretation. **FMODIF=INFO** Value in TSVPFCMO='INFO'. Read the field DSLRSLEV which contains the maximum level of nesting of repeatable sequences currently initialized in the TOF. This value is returned in the first fullword of the buffer supplied with the BUFFER operand. If the field name is set to *DSLTOF$, the following 5 fullwords are returned in the buffer supplied with the BUFFER operand: **TINFxten** TOF Extension value **TINFmsz** TOF Maximum Size **TINFjbsz** TOF Join Buffer Size **TINFfmt** TOF Format 1 (single TOF) or 2 (split TOF) **TINFdaa** TOF Data Address **OPTION=NO or OPTION=DATA** Value in TSVPNEOM='D'. Read the data area. **Note:** Values 'A' and 'B' are processed like value 'D'. **OPTION=YES** Value in TSVPNEOM='O'. READ the option.
TOFNEW	Code = 'TNEW' in TSVPFTYP. Used to create a new TOF. The buffer referenced by TSVPADDR is formatted as an empty TOF ready for use. This buffer must be supplied and conform to MERVA ESA buffer conventions. This function must be the first DSLTOFSV call before other TOF services can be successfully performed.

Table 1. TOF Service Function: TYPE values and Associated Functions Modifiers (continued)

Type	Function Description
WRITE	Code = 'WRT ' in TSVPFTYP. Used to write data according to the function modifier FMODIF from the buffer referenced by TSVPBUFF to a data area or option field of the TOF. If the data area or option referenced exists in the TOF, it is modified, otherwise it is inserted into the TOF. **Note:** Data is written only if the nesting identifier evaluated in the field reference is initialized in the TOF. **FMODIF=DEEDIT** Value in TSVPFCMO='DEED'. De-editing and no checking of data referenced by TSVPBUFF. **FMODIF=CHECK** Value in TSVPFCMO='CHK'. No de-editing, but checking of data referenced by TSVPBUFF. **FMODIF=DECHECK** Value in TSVPFCMO='DECH'. De-editing and checking of data referenced by TSVPBUFF. **FMODIF=IGNORE** Value in TSVPFCMO='IGN '. Write data area only if no data area with identical contents is already in the TOF for the field referenced. **FMODIF=SVAL** Value in TSVPFCMO='SVAL'. Used to set the expansion values for a Dynamic TOF. The values must be written to the field *DSLTOF$. **TINFxten** TOF extension value **TINFmsz** TOF maximum size **TINFjbsz** TOF join buff size **TINFfmt** TOF format 1 or 2 **TINFdaa** TOF data address **OPTION=NO or OPTION=DATA** Value in TSVPNEOM='D'. Write the data area. **Note:** Values 'A' and 'B' are processed like value 'D'. **OPTION=YES** Value in TSVPNEOM='O'. Write the option.

Mapping the Parameter List and Temporary Storage

The DSLTSV macro can be used to map the DSLTOFSV parameter list and to define a temporary storage work area.

Name	Operator	Operands
[*label*]	**DSLTSV**	[**DSECT=YES,**] **MF=**{**L**|**TS**} [**,PREFIX=**{<u>**TSV**</u>|*ccc*}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

DSECT
When DSECT=YES is specified, the storage area is mapped as an assembler language dummy section. If the parameter is omitted, the storage area is mapped with double word alignment.

MF
The macro format.

L Maps the DSLTOFSV parameter list.

TS
Maps a temporary storage area (see "WORK parameter of DSLTSV MF=E" on page 239).

PREFIX
The first 3 characters of the field names to be generated in the DSLTOFSV parameter list. The default is TSV.

DSLTXT: Defining the Transaction Table

The MERVA ESA transaction table is used by MERVA ESA queue management to trigger transactions for queue events.

The DSLTXT macro is used to:
- Generate a transaction table
- Map the transaction table entry

The transaction table consists of a sequence of DSLTXT macros. The first instruction should have a label; if it does not, the default label DSLTXTT is used. The last statement of the table must be the END statement.

No assembler language statements except SPACE, EJECT, END, and possibly PRINT must be used in the table definition.

If errors are detected in a specific DSLTXT macro, appropriate MNOTEs are provided.

MERVA ESA provides a sample transaction table DSLTXTT.

Generating a Transaction Table

Name	Operator	Operands
[*label*]	**DSLTXT**	**TYPE=**{**INITIAL**|**FINAL**}

Programming Notes:

label
: The name of the CSECT generated. It must be a unique label according to the assembler language naming conventions. The label must appear in the first DSLTXT statement.

TYPE
: TYPE=INITIAL must be the first transaction table definition. If the first statement does not contain a label, DSLTXTT is used. In the second and following calls of DSLTXT, the label is disregarded. TYPE=FINAL must be the last transaction definition statement.

Generating a Transaction Definition Entry

For MERVA ESA the value of the TRAN parameter in a DSLFNT entry refers to the name of an entry in the transaction definition table.

Name	Operator	Operands
[*label*]	**DSLTXT**	[**TYPE=<u>ENTRY</u>**] **,NAME=***cccccccc* [**,METHOD=**{**<u>LOCAL</u>**|**CICSBATCH**|**APPCMVS**}] [**,DELAY=**({**<u>1</u>**|*events*},{**<u>0</u>**|*nnnn*}[,hhmmss])] [**,LTERM=***cccccccc*] [**,LUNAME=***cccccccc*] [**,MODE=***cccccccc*] [**,SYMDEST=***cccccccc*] [**,SYSID=***cccccccc*]

Name	Operator	Operands
		[,**TPNAME=***cccccccc*] [,**TPNMAP=***cccccccc*] [,**USERID=***cccccccc*]

Programming Notes:

label
For TYPE=ENTRY calls of DSLTXT, the label is disregarded.

TYPE
TYPE=ENTRY specifies that a transaction definition entry is to be generated.

NAME
The name of the entry. This name is referred to by the TRAN parameter in the function table entry.

METHOD
The method used to start the transaction.

LOCAL	The transaction is to be started directly via EXEC CICS START when running under CICS. When running in an IMS BMP, the start request is directly inserted in the IMS message queue. This is the traditional method as used in previous MERVA ESA releases. This is the default.
CICSBATCH	The batch EXEC CICS interface is to be used. The SYSID parameter identifies the CICS where the transaction is started. This option is available for MVS only.
APPCMVS	APPC/MVS is to be used to start the transaction. This option is available for MVS only and is also used to start IMS transactions. If this method is specified, the parameters SYMDEST or LUNAME and TPNAME must be specified.

DELAY
The condition for the start of a transaction:

- The first value is a number from 1 to 999 999 999 and specifies the number of events after which the transaction is to be started.
- The second value specifies the maximum delay in seconds. If at least one event has occurred by the time this delay has passed, the transaction will be started, even if the number of events specified for the first value has not been reached. The maximum delay is 86399 seconds.
- The third value is a time of day specified in the form **hhmmss** (hours, minutes, seconds). For example, the time **9:25 PM** would be specified **21:25:00**. If at least one event has occurred by that time, the transaction is started at that time.

A delayed transaction is started only if MERVA ESA is still active after the requested interval or time of day.

If DELAY is not specified, the transaction is started immediately without any delay.

This parameter can be used for tuning purposes to force the transaction programs to process batches of messages rather than a single message in each run, thereby avoiding the overhead associated with repeatedly initializing and terminating the application program.

LTERM

The logical terminal name. The name can be from 1 to 8 characters long. This name corresponds to an LTERM name defined in the CICS terminal definition or TERMINAL macro of the IMS nucleus generation. That is the terminal name for which the transaction is started.

The specified LTERM value overwrites the value in the TUCB. The value is used by the MERVA ESA mapping program in CICS to start the transaction with terminal. If the LTERM is not specified, the value in the TUCB is left unchanged, that is, the value as defined in the function table or specified by a CF (change function) command.

It is a character string of 1 to 8 characters according to the assembler language naming conventions.

LUNAME

The APPC LU name when METHOD=APPCMVS has been specified.

This can be the LU name only (1- to 8-byte character string), or combined network ID and network LU name (two 1- to 8-byte character strings, concatenated by a period).

MODE

The mode name when METHOD=APPCMVS has been specified. The parameter is optional.

It is a character string of 1 to 8 characters according to the assembler language naming conventions.

SYMDEST

The symbolic destination name when METHOD=APPCMVS has been specified. This parameter is optional. This parameter is an alternative for specifying the LUNAME, TPNAME, and the MODE parameter. The TPNAME of the symbolic destination can be overwritten by specifying the TPNAME parameter in the DSLTXT Macro.

It is a character string of 1 to 8 characters according to the assembler language naming conventions.

SYSID

This parameter is required for METHOD=CICSBATCH and identifies the CICS system where the program is started. When running under CICS and using the local method, it is an optional parameter. If specified, the parameter lets you start a remote transaction in a different CICS.

It is a character string of 1 to 8 characters according to the assembler language naming conventions.

TPNAME

The transaction code to be started. For METHOD=LOCAL, the specified value is the name of the transaction to be started. For METHOD=CICSBATCH, this value is passed to the mirror program started via batch DPL. The mirror program in turn starts this specified transaction on the target CICS system. For METHOD=APPCMVS, this name is used as the APPC TP name parameter, and can be up to 64 characters long.

TPNMAP
This value overwrites the transaction code value in the TUCB entry only, but is not directly used to specify the started transaction. For IMS transactions started via APPC/MVS, the TPNAME parameter can specify a MERVA ESA provided mapping transaction.

The mapping transaction is started via APPC/MVS and receives the target transaction name in the TUCB. The mapping transaction inserts the TUCB directly into the IMS message queue of the target transactions. The MERVA ESA transactions do not need such a mapping, because they are able to process both types of TUCBs:

1. A TUCB directly inserted in the IMS message queue, starting with the length field
2. A TUCB received via APPC which is prefixed with the transaction code in the first positions of the buffer

In this case the TPNMAP parameter is not specified.

Customer applications which can only process TUCBs of the first type need to use the MERVA ESA mapping transaction program DSLCIMAP.

USERID
This parameter is optional and used only when running under CICS for METHOD=LOCAL. It lets you start a transaction under a different user ID. CICS performs a security check to verify that the user ID under which MERVA ESA is running is authorized to the specified user ID.

It is a character string of 1 to 8 characters according to the assembler language naming conventions.

Mapping a Transaction Table Entry

Name	Operator	Operands
[*label*]	**DSLTXT**	**TYPE=MAP** [**,DSECT=**{**NO**|**YES**}] [**,PREFIX=**{**TXT**|*ccc*}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

TYPE
The map of the transaction table entry is to be generated (TYPE=MAP). If TYPE=MAP is specified, all other parameters except PREFIX and DSECT are ignored.

DSECT
Whether the map is to be defined as a DSECT. The default is NO.

PREFIX
The 3-character prefix of the variable names of the map. The default is TXT.

DSLUSR: Defining the User File

The DSLUSR macro serves the following purposes:

- Prepare the DSLNUSR parameter list (the EP parameter controls whether the request will be executed as a direct or central service)

  Note: Only the DSP (display) and LST (list) functions can be executed by an application program, either directly using the DSLUSR macro or via the API interface. In the MERVA ESA customizing module DSLPRM, EXDSP=YES must be coded. User file access should be done via DSLAPI functions USRG and USRN.
- Map the parameter list of DSLNUSR
- Map the user file record
- Map the active user table entry

Calling DSLNUSR

Name	Operator	Operands

[*label*]	**DSLUSR**	**TYPE=**{**SIGNON** }
		{**FNT** }
		{**FS** }
		{**SIGNOFF**}
		{**CHGPW** }
		{**CHKPW** }
		{**ONLINE** }
		{**ADD** }
		{**CHG** }
		{**DEL** }
		{**DSP** }
		{**LST** }
		{**LST1** }
		{**UNLK** }
		
		[**,BUF=**{*addr*|(*r*)}]
		
		[**,EP=DSLNUSR**]
		
		[**,FUNC=**{*addr*|(*r*)}]
		
		[**,MF=(E,**{*addr*|(*r*)})]
		
		[**,PREFIX=**{<u>**USRP**</u>|*cccc*}]
		
		[**,PW=**{*addr*|(*r*)}]
		
		[**,UID=**{*addr*|(*r*)}]

Programming Notes:

label
: A unique statement label according to assembler language conventions.

TYPE
: The type of service request:

SIGNON	User signon request
FNT	User function retrieval request
FS	User function selection request

SIGNOFF	User signoff request
CHGPW	Change password
CHKPW	Check password
ONLINE	User file online maintenance request with the following subfunctions:

ADD	Add a user file record
CHG	Change/replace a user file record
DEL	Delete a user file record
DSP	Display a user file record
LST	Display all user IDs, user names, and Origin IDs
LST1	Display one user ID, user name, and Origin ID
UNLK	Unlock a user file record

BUF
The label of a buffer or a general register containing its address. For signon and online functions ADD, CHG, DEL, DSP, the buffer contains a user file record. For online function LST, the buffer contains a list of user IDs, user names and Origin IDs. When function selection is requested, the buffer contains the function table entry of the selected function. When function retrieval is requested, the buffer contains a list of all functions that were found in the function table according to the definition in the user file record.

EP
The only possible value for the EP parameter is DSLNUSR. It specifies that the request will be executed as a direct service.

The EP parameter must be specified by all programs that are linked to DSLNUC. Before issuing the DSLUSR macro with EP=DSLNUSR, the program must ensure that general register 12 contains the address of the MERVA ESA communication area DSLCOM and that general register 13 points to a usable save area.

The parameter EP=DSLNUSR must not be used by programs that are not linked to DSLNUC. After using the DSLUSR macro to prepare the parameter list, a DSLNIC macro with TYPE=REQ and NAME=DSLNUSR is required to request execution of a central service.

If the EP parameter is omitted or invalid, a central service request is assumed.

FUNC
The label of an 8-byte field containing the function name, or general register containing the address of the function name field.

For TYPE=CHGPW (change password) this field must contain the new password in scrambled format.

For TYPE=FNT this field may contain the function name where the search for valid functions is to be started in the function table.

MF
The format of the macro. The first parameter must be E for execute form. The second parameter specifies the label of the DSLNUSR parameter list or a general register containing the address of the parameter list.

The default is (E,(1)).

PREFIX
The prefix for the labels of the fields in the parameter list. The prefix must be from 1 to 4 characters long and must contain only characters that result in valid assembler language symbols. The default is USRP.

This prefix must be the same as the prefix used to generate the parameter list description (see macro DSLUSR MF=L).

PW
The label of an 8-byte field containing the password in scrambled format or a general register containing the address of the field.

UID
The label of an 8-byte field containing the user ID or a general register containing the address of the field.

Mapping the Parameter List

Name	Operator	Operands
[*label*]	**DSLUSR**	**MF=L** [**,DSECT=**{**NO**|**YES**}] [**,FUNC=**{*addr*|(*r*)}] [**,PREFIX=**{**USRP**|*cccc*}] [**,UID=**{*addr*|(*r*)}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be L for list form.

DSECT
Whether the parameter list is to be mapped as a dummy section. The default is NO.

FUNC
The function name. The parameter may be an 8-byte field containing the function name, or a general register containing the address of the function name field. The default is 8 blanks.

PREFIX
The prefix for the labels of the fields in the parameter list. The prefix must be from 1 to 4 characters long and must contain only characters that result in valid assembler language symbols. The default is USRP.

UID
The user ID. The parameter may be an 8-byte field containing the user identification, or a general register containing the address of the identification field. The default is 8 blanks.

Mapping a User File Record

Name	Operator	Operands
[*label*]	**DSLUSR**	**MF=U**
		[**,DSECT=**{**NO**|**YES**}]
		[**,FUNC=**{*addr*|(*r*)}]
		[**,PREFIX=**{**USRU**|*cccc*}]
		[**,UID=**{*addr*|(*r*)}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be U for user file record format.

DSECT
Whether the user file record is to be mapped as a dummy section. The default is NO.

FUNC
The function name, which is generated into the first function field of the user file record. The parameter may be an 8-byte field containing the function name, or a general register containing the address of the function name field. The default is 8 blanks.

PREFIX
The prefix for the labels of the fields in the user file record. The prefix must be from 1 to 4 characters long and must contain only characters that result in valid assembler language symbols. The default is USRU.

UID
The user ID, which is generated into the user field of the user file record. The parameter may be an 8-byte field containing the user identification, or a general register containing the address of the identification field. The default is 8 blanks.

Mapping an Active User Table Entry

Name	Operator	Operands
[*label*]	**DSLUSR**	**MF=A**
		[**,DSECT=**{**NO**|**YES**}]
		[**,FUNC=**{*addr*|(*r*)}]
		[**,PREFIX=**{**USRA**|*cccc*}]
		[**,UID=**{*addr*|(*r*)}]

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be A for active users format.

DSECT
Whether the active user table entry is to be mapped as a dummy section. The default is NO.

FUNC
The function name, which is generated into the function field of the active user table entry. The parameter may be an 8-byte field containing the function name, or a general register containing the address of the function name field. The default is 8 blanks.

PREFIX
The prefix for the labels of the fields in the active user table entry. The prefix must be from 1 to 4 characters long and must contain only characters that result in valid assembler language symbols. The default is USRA.

UID
The user ID, which is generated into the user field of the active user table entry. The parameter may be an 8-byte field containing the user identification, or a general register containing the address of the identification field. The default is 8 blanks.

DSLWTO: Defining the Write-to-Operator Interface

The DSLWTO macro serves the following purposes:

- Prepare the parameter list for the write-to-operator interface DSLWTOP and call DSLWTOP
- Map the write-to-operator interface parameter list

Calling DSLWTOP

The DSLWTO macro is used to display unsolicited messages from MERVA ESA programs on the system console. It is used only by programs that are not linked to DSLNUC.

DSLWTOP may also be used to add the display messages to the display message (DM) table and to the MERVA ESA journal data set.

Name	Operator	Operands
[*label*]	**DSLWTO**	**FROM=**{*addr*|(*r*)} [**,MF=(E,**{*addr*|(*r*)}**)**]

Programming Notes: Before using this macro, you must ensure that general register 13 points to a usable save area and that the MERVA ESA communication area DSLCOM is addressable.

The messages will be added to the display message table and to the MERVA ESA journal data set only if:

- The field COMNICPL contains the address of the DSLNICP parameter list
- A MERVA ESA intertask communication block (ICB) has been allocated

If the MERVA ESA identification is to be added to the display message, the field COMPRMA of DSLCOM must contain the address of DSLPRM.

label
: A unique statement label according to assembler language conventions.

FROM
: The symbolic address of the buffer containing the operator message, or a general register containing the address of the buffer.

MF
: The format of the macro. The first parameter must be E for execute form. The second parameter specifies the address of the DSLWTOP parameter list.

  The default is (E,(1)).

  For the address parameters in this macro, the following notations are possible:

  addr
  : A symbolic label

  (r) A general register containing the address

Mapping the Write-to-Operator Interface Parameter List

Name	Operator	Operands
[*label*]	**DSLWTO**	**MF=L**

Programming Notes:

label
  A unique statement label according to assembler language conventions.

MF
  The format of the macro. It must be L for list form.

Note: The DSLWTOP parameter list contains the definition of the parameter list for DSLNMOP at label WTONMOPL.

DSLZCW: Defining Code Word Table Entry

This macro is used to define the code words in a code word table that is used by the code word checking exit 2002 (DSLZC002).

Generating a Code Word Table

Name	Operator	Operands
[*label*]	**DSLZCW**	**TYPE=**{**INIT**|**FINAL**}

Programming Notes:

label
The name of the CSECT generated. It must be a unique label according to the assembler language naming conventions. The label must be specified in the INIT definition.

TYPE
TYPE=INIT must be the first code word table definition, and the label must be specified. In the second and following calls of DSLZCW, the label is disregarded. TYPE=FINAL must be the last code word table definition statement.

Generating a Code Word Entry

Name	Operator	Operands
	DSLZCW	[**TYPE=<u>ENTRY</u>**] **,LEFT=***nn* **,RIGHT=***nn* **,TEXT=***text*

Programming Notes:

TYPE
TYPE=ENTRY is specified for a code word entry. The default is TYPE=ENTRY.

LEFT
The position of the first character of the code word in the text.

RIGHT
The position of the last character of the code word in the text.

TEXT
Text containing the code word and optional explanatory text. The first 72 characters of this text are displayed in the explanation help panel.

DSLZSC: Associating a Subfield and a Code Word Table

This macro is used to define the code word table that is to be used, to check a subfield, by the checking exit 2002 (DSLZC002).

Generating the Table that Associates a Subfield and a Code Word Table

Name	Operator	Operands
[*label*]	**DSLZSC**	**TYPE=**{**INIT**|**FINAL**}

Programming Notes:

label
The name of the CSECT generated. It must be DSLZSCT. The label must be specified in the INIT definition.

TYPE
TYPE=INIT must be the first subfield definition, and the label must be specified (DSLZSCT). In the second and following calls of DSLZSC, the label is disregarded. TYPE=FINAL must be the last code word table definition statement.

Generating a Subfield Entry

Name	Operator	Operands
	DSLZSC	[**TYPE=ENTRY**] **,SF=***name* **,TABLE=***cccccccc* [**,FG=***nn*] [**,MT=***message type*] [**,OWNER=***owner*] [**,REL=***release*] [**,VER=***version*]

Programming Notes:

TYPE
TYPE=ENTRY is specified for a subfield definition entry. The default is TYPE=ENTRY.

SF
The name of the subfield. This is the name that the checking exit 2002 (DSLZC002) is searching for.

TABLE
The name of the code word table to be used by the checking exit 2002 (DSLZC002). It is a character string of 1 to 8 characters according to the assembler language naming conventions.

FG
A field group number from 1 to 255. The entry is only to be used if the subfield has a field group that matches this number.

MT
A message type of 1 to 8 characters. The entry is only to be used if the message type in the UNH matches this name.

Note: If the UNH has not been filled, the message type is taken from the DSLEXIT field.

OWNER
An owner name of 1 to 2 characters. The entry is only to be used if the owner name in the UNH field matches this name.

REL
A release number of 1 to 3 characters. The entry is only to be used if the release number in the UNH field matches this number.

VER
A version number of 1 to 3 characters. The entry is only to be used if the version number in the UNH field matches this number.

DSLZSS: Defining a Subfield Exception Entry

This macro is used to define subfield exception entries, in the tables (DSLZMST, DSLZMSDT, DSLZNST) used by checking exit 2001 (DSLZC001).

Generating an Exception Table

Name	Operator	Operands
[*label*]	**DSLZSS**	**TYPE=**{**INIT**|**FINAL**}

Programming Notes:

label
: The name of the CSECT generated. It must be DSLZMST, DSLZMSDT or DSLZNST. The label must be specified in the INIT definition.

TYPE
: TYPE=INIT must be the first subfield definition. The first statement must contain a label (DSLZMST, DSLMSDT or DSLZNST). In the second and following calls of DSLZSS, the label is disregarded. TYPE=FINAL must be the last subfield definition statement.

Generating a Subfield Entry

Name	Operator	Operands
	DSLZSS	[**TYPE=ENTRY**] **,SF=***name* [**,FG=***nn*] [**,MT=***message type*] [**,OWNER=***owner*] [**,REL=***release*] [**,VER=***version*]

Programming Notes:

TYPE
: TYPE=ENTRY is specified for a subfield definition entry. The default is TYPE=ENTRY.

SF
: The name of the subfield. This is the name that the checking exit 2001 (DSLZC001) is searching for.

FG
: A field group number from 1 to 255. The entry is only to be used if the subfield has a field group that matches this number.

MT
: A message type of 1 to 8 characters. The entry is only to be used if the message type in the UNH matches this name.

Note: If the UNH has not been filled, the message type is taken from the DSLEXIT field.

OWNER
An owner name of 1 to 2 characters. The entry is only to be used if the owner name in the UNH field matches this name.

REL
A release number of 1 to 3 characters. The entry is only to be used if the release number in the UNH field matches this number.

VER
A version number of 1 to 3 characters. The entry is only to be used if the version number in the UNH field matches this number.

DSLZDEF: Defining a Subfield Default Entry

This macro is used to define the subfield defaults table that is used by the default exit 2001 (DSLZD001).

Generating the Subfield Default Table

Name	Operator	Operands
[*label*]	**DSLZDEF**	**TYPE=**{**INIT**|**FINAL**}

Programming Notes:

label
The name of the CSECT generated. It must be DSLZDEFT. The label must be specified in the INIT definition.

TYPE
TYPE=INIT must be the first subfield definition, and the label must be specified (DSLZDEFT). In the second and following calls of DSLZDEF, the label is disregarded. TYPE=FINAL must be the subfield definition statement.

Generating a Subfield Default Entry

Name	Operator	Operands
	DSLZDEF	[**TYPE=ENTRY**] **,SF=***name* **,TEXT=***text* [**,FG=***nn*] [**,MT=***message type*] [**,OWNER=***owner*] [**,REL=***release*] [**,VER=***version*]

Programming Notes:

TYPE
TYPE=ENTRY is specified for a subfield definition entry. The default is TYPE=ENTRY.

SF
The name of the subfield. This is the name that the checking exit 2002 (DSLZC002) is searching for.

TEXT
Text containing the default value.

FG
A field group number from 1 to 255. The entry is only to be used if the subfield has a field group that matches this number.

MT
A message type of 1 to 8 characters. The entry is only to be used if the message type in the UNH matches this name.

Note: If the UNH has not been filled, the message type is taken from the DSLEXIT field.

OWNER
An owner name of 1 to 2 characters. The entry is only to be used if the owner name in the UNH field matches this name.

REL
A release number of 1 to 3 characters. The entry is only to be used if the release number in the UNH field matches this number.

VER
A version number of 1 to 3 characters. The entry is only to be used if the version number in the UNH field matches this number.

Chapter 4. SWIFT Link: Macros

DWSAUT: Calling the Authentication Services

The macro DWSAUT invokes the SWIFT Link authenticator-key file support DWSAUTP.

Name	Operator	Operands
[*label*]	**DWSAUT**	[**TYPE=**{**AUT**|**UPDATE**}] **,MF=**{**L**|**(E,**{*addr*|(*r*)}**)**} [**,EP=DWSAUTP**] [**,MSG=**{*addr*|(*r*)}]

Programming Notes: You can use one of two notations for the address parameters in this macro.

addr
A symbolic label.

(r) A general register containing the address.

label
A unique statement label according to assembler language conventions.

TYPE
This operand is mandatory only for MF=(E,...). It specifies which authenticator-key file support function to perform:

AUT
Checks whether a SWIFT message is to be authenticated. If so, the authenticator key is retrieved from the file and the message is authenticated. A reason code defines which diagnostic message is displayed, indicating if and how authentication was performed.

UPDATE
Requests an update of the authenticator-key file. When this TYPE is specified, the update function must be set in the DWSAUT parameter list before the DWSAUT macro is executed.

MF
The format of the macro:

L Maps the parameter list of DWSAUTP.

E Shows the execute form. The second subparameter, *addr* or *(r)*, specifies the address of the parameter list.

EP
The only possible value for the EP parameter is DWSAUTP. It specifies that the request will be executed as a direct service.

The EP parameter must be specified by all programs that are linked to DSLNUC. Before issuing the DWSAUT macro with EP=DWSAUTP, the program must ensure that general register 12 contains the address of the MERVA ESA communication area DSLCOM and that general register 13 points to a usable save area.

The parameter EP=DWSAUTP must not be used by programs that are not linked to DSLNUC. After using the DWSAUT macro to prepare the parameter list, a DSLNIC macro with TYPE=REQ and NAME=DWSAUTP is required to request execution of a central service.

If the EP parameter is omitted or invalid, a central service request is assumed.

MSG
If TYPE=AUT, it specifies the address of the message in SWIFT format to be authenticated.

If a generic update function (List, Exchange or Delete) was used and TYPE=UPDATE, it specifies the address of the buffer for the list of processed authenticator-key file records.

DWSCI: Defining the Central Institutes Table

The DWSCI macro is used to define Central Institutes participating in PREMIUM and FIN-Copy services offered by SWIFT. The name of the table can be defined in the SWIFT link customizing parameter module DWSPRM (see "DWSPARM: Generating the DWSPRM Module" on page 275).

If one of these services is used, a full or a partial copy of a financial message is sent to a Central Institute by SWIFT before it is delivered to the receiver. In addition to the message authentication with a MAC trailer the copy of the message to the Central Institute may be authenticated with a PAC trailer.

For this purpose the SWIFT link authentication program DWSAUTP uses the DWSCIT table to check the copy criteria for a central service of each incoming and outgoing SWIFT message. Each financial institution and each message type which participate in one of these services have to be defined in the DWSCIT table.

You can either use the PREMIUM service or a FIN-Copy service for one financial institution. The number of service definitions in the DWSCIT table is unlimited.

Defining a PREMIUM Service

A PREMIUM service can be defined with a single macro statement. It describes the copy criteria of the SWIFT message. If they are met, a PAC is calculated using all message fields for authentication. The copy criteria are as follows:

- The name of the Home LT
- The name of the Central Institute
- The name and the option of the message field containing the name of the Central Institute
- The message identification

Name	Operator	Operands
[*label*]	**DWSCI**	**CI=***name* **,HOME=***name* **,MSG=**(*mtype*[,*mtype*...[,*mtype*]]) [**,ID=**{<u>**W**</u>|*c*}] [**,FIELD=**{<u>**SW53**</u>|*field*}] [**,OPTION=**{<u>**A**</u>|*option*}]

Programming Notes:

label
A unique statement label according to assembler conventions which is used for the CSECT name of the Central Institutes table.

CI
The 8-character name of the Central Institute.

HOME
The 8-character name of the Home LT.

MSG
A list of 8-byte alphanumeric message identifications that are part of the PREMIUM set for this Central Institute.

ID
The 1-character code of the external line format used to generate the PAC trailer. The default is W, because in phase 1 of PREMIUM all fields in a message are used.

FIELD
The 8-character name of the field containing the Central Institute. The default is SW53.

OPTION
The 1-character option for the field containing the Central Institute. The default is A.

Defining a FIN-Copy Service

A FIN-Copy service definition requires the following types of the DWSCI macro:

- TYPE=DEF to define the service characteristics, which contain the copy criteria and the authentication method. A PAC is calculated, if double authentication is specified for the service. The copy criteria are as follows:
  - The name of the FIN-Copy service (the message field 103 must contain this 3-character service code)
  - The name of the Home LT
  - The currency code (optional)
- TYPE=MSG to define a SWIFT message type that participates in the FIN-Copy service. Code one macro statement for each message type.

  The list of message types must be immediately preceded by the FIN-Copy service definition (DWSCI TYPE=DEF).

  For each message type, a list of the fields may be specified which are included in the partial copy to the Central Institute by SWIFT and which are therefore used to calculate the PAC.

Name	Operator	Operands
[*label*]	**DWSCI**	**TYPE=DEF**
		,CI=*name*
		,FINCOPY=*ccc*
		,HOME=*name*
		[**,ACTDATE=**{**19960301**|**YYYYMMDD**}]
		[**,ACTTIME=**{**0001**|**HHMM**}]
		[**,AUTH=**{**1**|**2**}]
		[**,CURR=***ccc*]
		[**,FULLCOPY=**{**Y**|**N**}]
		[**,LIVE=**{**Y**|**N**}]
		[**,VERSION=**{**01**|**nn**}]

Programming Notes:

label
A unique statement label according to assembler conventions which is used for the CSECT name of the Central Institutes table.

CI
The 8-character name of the Central Institute.

FINCOPY
The 3-character name of the FIN-Copy service defined by SWIFT.

HOME
The 8-character name of the Home LT.

ACTDATE
An 8-digit activation date for this FIN-Copy service. The default is 19960301 (March 1st, 1996).

ACTTIME
A 4-digit activation time for this FIN-Copy service. The default is 0001 (one minute after midnight).

AUTH
Whether the service uses single or double authentication. 1 indicates single authentication (MAC trailers only); 2 indicates double authentication (MAC and PAC trailers). The default is 1.

CURR
A 3-character currency code. If a currency code is given, only messages containing this currency will be selected for the FIN-Copy service.

FULLCOPY
Whether the central institute receives the whole message or only a partial copy. Y indicates a full message; N indicates a partial copy. The default is Y.

LIVE
Whether the service is in live or test-and-training mode. Y indicates live mode; N indicates test-and-training mode. The default is Y.

VERSION
A 2-digit version number assigned by SWIFT to the FIN-Copy service. The default is 01.

Name	Operator	Operands
	DWSCI	**TYPE=MSG** **,MT=***mtype* [**,CIFLD=**{<u>**SW53**</u>|*cccccccc*}] [**,CURRFLD=***cccccccc*] [**,FIELDS=**(*list of field tags*)]

Programming Notes:

MT
An 8-byte alphanumeric message identification to be included in the FIN-Copy service. It must be a valid SWIFT message type, as defined in the MERVA ESA message type table DSLMTTT.

CIFLD
The 8-character name of a field or subfield, as defined in the MERVA ESA field definition table DSLFDTT, which contains an identifier for the Central Institute of the FIN-Copy service. The default is SW53.

CURRFLD
The 8-character name of the field or subfield, as defined in the MERVA ESA field definition table DSLFDTT, which contains the currency code of the message. If the currency code is in a subfield, the subfield name must be used.

The CURRFLD parameter is required when the CURR parameter was specified in the FIN-Copy service definition. If the field occurs more than once in the message, the first occurrence will be used to select the message for FIN-Copy.

FIELDS
A list of up to 30 SWIFT field tags, including the option character. An asterisk (*) may be used to indicate all options of the field tag.

The FIELDS parameter is required when FULLCOPY=N (partial messages) was specified in the FIN-Copy service definition. This list specifies the fields from block 4 of the message which will be included in the partial copy sent to the Central Institute.

If FULLCOPY=Y is specified, the FIELDS parameter is ignored.

Mapping a Central Institutes Table Entry

The DWSCI macro may be used to generate a storage description of a Central Institutes table entry.

Name	Operator	Operands
[*label*]	**DWSCI**	**MF=D**

Programming Notes:

label
A unique statement label according to assembler language conventions.

MF
The format of the macro. It must be D to generate a DSECT of a central institutes table entry.

DWSCUR: Defining the Currency Code Table

This macro is used to define the currency codes in the copy code DWSCURT. This copy code is used to generate both of the following:

- The SWIFT Link Currency Code Table (DWSMCCRT), which is used to check the currency codes when the Online Currency Code File is not used
- The SWIFT Link Currency Code Help Panel (DWSHCUR)

The Currency Code table supplied with SWIFT Link defines the following for each currency code:

- The 3-character currency code
- The number of digits after the decimal place for an amount of this currency
- The descriptive name of the currency (for the help panel)
- The names of the countries that use that currency (for the help panel)

When currency codes are changed, added, or deleted in the DWSCURT copy code, DWSMCCRT and DWSHCUR must be assembled, and DWSMU141 and DWSHCUR must be link-edited.

Name	Operator	Operands
[*label*]	**DWSCUR**	[**TYPE=**{**HELP**|**TABLE**}] [**,COM=**{'*currency_name*' }] {('*currency_name*','*country_1*',...,'*country_n*')} [**,COM2=**{'*country_n+1*'|('*country_n+1*',...,'*country_n+m*')}] [**,CUR=***currency_code*] [**,NUM=***number*]

Programming Notes:

TYPE
This parameter is used only in the first DWSCUR macro of the DWSMCCRT and DWSHCUR modules:

- TYPE=HELP indicates that the help MCB DWSHCUR is to be generated.
- TYPE=TABLE indicates that the currency code table (DWSMCCRT) for checking is to be generated.

COM
The first value defines the descriptive name of the currency; the other (optional) values define the names of the countries that use this currency. These names are used by the help MCB. This parameter is required when the TYPE parameter is not specified.

COM2
The names of countries in addition to those specified in the COM parameter that use this currency. These names are used by the help MCB.

CUR
The 3-character currency code, as defined in the S.W.I.F.T. Directory book. This parameter is required when the TYPE parameter is not specified.

NUM
The number of digits that can follow the decimal place of an amount for this currency, as defined in the S.W.I.F.T. Directory book. This parameter is required when the TYPE parameter is not specified.

DWSLT: Generating the Logical Terminal Table

This macro:

- Maps the logical terminal table header and the logical terminal table entry
- Generates the logical terminal table

The logical terminal table is used to define the logical terminals, their synonyms, and their applications with the attributes for routing modules, ready queues, and the technology used for login and select: paper tables or secure login/select as defined by SWIFT. It also is used by DWSDGPA for controlling the line operations associated with a master logical terminal.

Mapping the Logical Terminal Table Header and Entry

Name	Operator	Operands
	DWSLT	**MF=D**

Programming Notes:

MF=D
Maps the logical terminal table header and the logical terminal table entry. No other parameter can be specified in this macro call.

Generating the Logical Terminal Table

Name	Operator	Operands
[*label*]	**DWSLT**	[**,TYPE=**{<u>**MASTER**</u>|**SYNONYM**|**AP**}] **NAME=***name* [**,DELSUB=**(*cccccc*[,*cccccc*,...])] [**,LINE=**{<u>**1**</u>|*nn*}] [**,READYQ=**(*name1*[,*name2*,...])] [**,ROUTIN=**{<u>**DWSRTIN**</u>|*name*}] [**,ROUTOUT=**{<u>**DWSRTOUT**</u>|*name*}] [**,ROUTSK=***name*] [**,SKEYQ=***name*] [**,TFLAG=**{<u>**PT**</u>|**SLS**}] [**,USENAME=***name*] [**,WINDOW=**{<u>**10**</u>|*nnn*}]

Programming Notes:

label
The label of the first DWSLT macro defines the name of the logical terminal table. This name must be specified with the LTT parameter of the DWSPARM macro. The default is DWSLTT.

TYPE
The type of logical terminal table entry:

MAS or MASTER
For a master logical terminal. This is the default.

A master logical terminal can be the subject of a **login**, **logout**, **abortlt**, **setlt** or **dl** command, and it has its own sequence of input sequence numbers (ISN), output sequence numbers (OSN), and login sequence numbers (LSN). The master logical terminal also determines if paper tables or secure login/select is used.

SYN or SYNONYM
For a synonym logical terminal. A synonym logical terminal shares the general purpose application (GPA) and the financial application (FIN) with the preceding master logical terminal.

AP
The financial application (FIN) for the preceding master logical terminal and its synonyms. A FIN application can be the subject of a **select**, **quit**, or **abortap** command, and it has its own sequence of input sequence numbers (ISN), output sequence numbers (OSN), and select sequence numbers (SSN).

NAME
For TYPE=MASTER and TYPE=SYNONYM, this specifies the name of the master or synonym logical terminal. It must have 9 characters, be unique, and it must follow the naming conventions specified in the *S.W.I.F.T. User Handbook*.

For TYPE=AP, this specifies the name of the financial application of the previously defined master logical terminal. It must be a name from 3 to 6 characters starting with **FIN**.

A maximum of 255 logical terminal table entries can be specified in one logical terminal table.

DELSUB
The list of mnemonic names of the delivery subsets as defined to SWIFT for the FIN application. Each delivery subset mnemonic must be 6 characters long, and you can specify up to 30 delivery subset mnemonics separated by commas.

This list of delivery subset mnemonics is used in the SWIFT select message in field 338 when selecting this special application for a logical terminal. The list of delivery subset mnemonics can be changed with the **dds** command or with the **select** command.

LINE
For TYPE=MASTER, specifies the line number, from 1 (the default) to 30, on which messages of this logical terminal and its FIN application are sent to and received from the SWIFT network.

A line description module must be available for each specified line. For example, DWSLIN1 may be the name of a line module for line number 1, DWSLIN2 for line number two, and so on. Use the DWSVLINE macro to define an X.25 data communication line to SWIFT.

The line number can be changed with the **setlt** command.

READYQ
The names of up to four MERVA ESA queues, defined in the MERVA ESA function table, from which messages of this master logical terminal or FIN application are sent to the SWIFT network. The READYQ parameter is only valid for TYPE=MASTER or TYPE=AP.

The ready queues defined with the READYQ parameter are processed by the SWIFT Link in the order they are specified; that is, for all master destinations logged in for input on the same line, the queues defined in the first subparameter are processed until they are all empty, then the queues defined in the second subparameter are processed until they are all empty, and so on. To skip a queue position, leave a subparameter position empty by coding only a comma. For example, to use only the third and fourth queues, which have names RQ3 and RQ4, code READYQ=(,,RQ3,RQ4).

This logic is started from the beginning each time when the SWIFT Link looks for a message to send.

Note: The ready queues defined for one logical terminal table entry must not be used for another logical terminal table entry.

ROUTIN
The name of the routing module for messages that are read from the ready queues (SWIFT input messages), and for the APDUs that are generated by DWSDGPA. By default ROUTIN=DWSRTIN is used. This parameter can be used for master logical terminals and special applications, but not for a synonym logical terminal.

ROUTOUT
The name of the routing module for messages that are received from the SWIFT network. By default ROUTOUT=DWSRTOUT is used. This parameter can be used for master logical terminals and special applications, but not for a synonym logical terminal.

ROUTSK
For TYPE=MASTER and TFLAG=SLS, the name of the routing module that is used for the routing of the request-for-session-key messages to a MERVA Link send queue. From this queue, MERVA Link will send these messages to the USE workstation.

SKEYQ
For TYPE=MASTER, AP, and SLS, the name of a MERVA ESA queue that is used for the storing of pregenerated session keys for the login of the master logical terminal or the select of the FIN application. The session keys are received from the USE workstation and loaded into the session key queue by the program DWSDLSK. A session key queue can be shared by several master logical terminals and FIN applications, or each master logical terminal and FIN application can have its own session key queue.

If a session key queue is defined, the SWIFT Link tries to get the session key from this queue during the execution of a **login** or **select** command, or during automatic login and select.

The contents of this queue can be maintained using the MERVA ESA queue key list at a MERVA ESA display station without displaying the session keys.

TFLAG
For TYPE=MASTER, the technology flag used for the login of the master logical terminal and for the select of the FIN application. The following values can be specified:

PT Indicates that paper tables are used for login and select. The 4-digit random and response numbers must be entered with the **login** and **select** commands, or DWSLOG2 is called to supply these numbers. This is the default value.

SLS Indicates secure login/select. In this case, the SWIFT Link gets the session key in one of the following ways:

- The operator enters the session key and the check value with the **login** and **select** commands (unconnected mode)
- The session key is retrieved from the queue of preloaded session keys defined with the SKEYQ parameter (preload is initiated at the USE workstation)
- The session key is retrieved from the card reader via the USE workstation (connected mode)

The TFLAG value can be changed with the **setlt** command. The SWIFT Link keeps the TFLAG value until it is changed again with a **setlt** command.

USENAME
For TYPE=MASTER, the 1- to 9-character name of the USE workstation that is used for secure login and select (SLS) for this master and its FIN application. If less than 9 characters are specified, USENAME is padded with the character X to obtain a 9-character USENAME. The name is used in the destination field of the application header and should therefore contain only the characters A to Z in the first six positions to avoid checking errors. The USENAME is used during routing of the request-for-session-key message to a MERVA Link send queue. From this queue, MERVA Link will send this message to the appropriate USE workstation. The default value is LT.

After having started the SWIFT Link, the USENAME value can be changed with the **setlt** command. The SWIFT Link keeps the USENAME value until it is changed again with a **setlt** command.

WINDOW
For TYPE=MASTER and TYPE=AP, the default window in the login or select message sent to the SWIFT network in field 110. The window size can be changed with the **login** or **select** command. The default value is 10.

DWSPARM: Generating the DWSPRM Module

The DWSPARM macro generates the customization parameter module for the SWIFT Link (DWSPRM module).

Name	Operator	Operands
[*label*]	**DWSPARM**	[,**AUTH**={**80**|*authenticator*}]
		[,**AUTHAGE**={**100**|*size*}]
		[,**AUTHENT**={**YES**|**NO**}]
		[,**AUTHMAN**={**MERVA2**|**MERVAESA**}]
		[,**AUTHUP**={**UNLOAD**|*password*}]
		[,**CHECK**=([{**NO**|**YES**}][,{**NO**|**YES**}])]
		[,**CIT**=*citname*]
		[,**FORMAT**={**TOF**|**ELF**}]
		[,**FORM2**={**W**|*formid*}]
		[,**INTACK**={**600**|*nnnn*}]
		[,**INTASS**={**10**|*nnnn*}]
		[,**INTRES**={**300**|*nnnn*}]
		[,**INTSKR**={**300**|*nnnn*}]
		[,**LINENAM**={**DWSLIN**|*linename*}]
		[,**LOG2**={**DWSLOG2**|*logname*}]
		[,**LSKQUE**={**SLSLOAD**|*cccccccc*}]
		[,**LTT**={**DWSLTT**|*lttname*}]
		[,**LTTQUE**={**SLSLTT**|*cccccccc*}]
		[,**MF=D**]
		[,**NET**={**S**|*netid*}]
		[,**PDU**={**512**|*size*}]
		[,**RSKQUE**={**SLSRECV**|*cccccccc*}]
		[,**SWIN**={**10000**|*size*}]
		[,**SWOUT**={**11000**|*size*}]
		[,**TLSTOR**={**200**|*size*}]

Programming Notes:

label
Defines the name of the customization parameter module, or with the MF=D parameter, the name of the DSECT. If you omit the label, DWSPRM is used.

AUTH
The indicator bit that is set on for those message types which require

authentication (NETSPEC parameter of the DSLMTT macro). The default is 80; that is, the first bit in this byte indicates that authentication is required.

AUTHAGE
The number of entries in the aging table used by DWSAUTP to store authenticator keys in main storage to reduce the accesses to the authenticator-key file. The default is 100. SIZE is a number from 0 to 625. If 0 is specified, no aging table is used by DWSAUTP.

AUTHENT
Whether SWIFT Link is to authenticate SWIFT input and output messages. The default is YES.

If AUTHENT=NO is specified, no authenticator-key file need be defined during the installation of the SWIFT Link, and the program DWSAUTIN (SWIFTAUT) must be removed from the copy code DWSNPTTC before generating DSLNPTT. Then the online maintenance of the authenticator-key file cannot be used.

AUTHMAN
If AUTHMAN=MERVAESA is specified, manual keys in the authenticator-key file will not be updated by BKE keys which are sent from the USE workstation in a special MT 999 to the queue BKEUPDT.

The default is MERVA2; that is, if manual keys are in a record, they will be overwritten by BKE keys.

AUTHUP
The 8-character password for scrambling authenticator keys before they are unloaded to a sequential file. The default is 'UNLOAD ' (note the two trailing padding blanks).

CHECK
Whether SWIFT Link checks the SWIFT messages:

- The first subparameter specifies whether the messages received from the SWIFT network are to be checked. The default is NO. When CHECK=(NO) is specified, checking could be performed in a later processing step by the MERVA ESA checking and expansion transaction DSLCXT. This can be invoked by routing the messages to the appropriate function queue.
- The second subparameter specifies if the messages sent to the SWIFT network are to be checked. The default is NO. When CHECK=(,NO) is specified, checking should be done before routing the messages to the ready queues in order to avoid negative acknowledgments sent by SWIFT in response to erroneous messages.

CIT
The name of the Central Institutes table for the SWIFT PREMIUM and FIN-Copy services.

FORMAT
The format in which messages received from the SWIFT network or system messages created by the SWIFT Link application are to be stored:

TOF	Tokenized form (this is the default)
ELF	External line format

There is a performance advantage in storing messages in external line format, because no mapping process is needed and checking is not performed. If during later processing the message is needed in tokenized form, the MERVA ESA mapping transaction DSLCXT can be used to perform the transformation.

In external line format the fields contained in the message are not directly accessible during routing. The header and trailer fields can be accessed as subfields using a different name from the tokenized format name. The impact on installed routing modules must be evaluated carefully when considering the external line format for the SWIFT Link. Refer to the *MERVA for ESA Customization Guide* for details.

Messages in ready queues are processed in the format in which they are stored regardless of the FORMAT specification. If a message is available in external line format, this format is used.

FORM2
The format identification which is used as FORMID parameter when mapping messages for the SWIFT network using the Message Format Service. The default is W.

INTACK
The time interval in seconds after which DWSDGPA is to notice when the acknowledgment for a message sent to the SWIFT network does not arrive. According to the SWIFT protocol, DWSDGPA must perform abort processing for this logical terminal or FIN application. The default is 600 seconds.

INTASS
The time interval in seconds after which the SWIFT Link is to notice when the confirmation for an association, disassociation, suspension, or resumption request does not arrive. According to the SWIFT protocol, the SWIFT Link must perform abort processing for this logical terminal or special application. The default is 10 seconds.

INTRES
The time interval in seconds after which the SWIFT Link starts a resumption if there is a message available for sending. The interval is counted from the operator message

```
DWS661I CBT suspension is complete on line n
```

If the MERVA ESA operator enters a SWIFT Link command that is related to this line (for example, a **login** or **logout** command), the resumption is started immediately. The default is 300 seconds.

INTSKR
When secure login/select is used, the time interval in seconds after which the SWIFT Link is to notice when the response for a session key request sent to the USE workstation does not arrive. The default is 300 seconds.

LINENAM
The first 6 characters of the names of the line description modules. The default is DWSLIN. The LINENAM parameter must correspond to the first 6 characters of the *label* of each DWSVLINE macro defining a line to the SWIFT network. The SWIFT Link uses these 6 characters and the 1- or 2-digit numbers of the line as seventh character and eighth character when loading the line definition module. For example, for LINENAM=DWSLIN and line number 1, the module DWSLIN1 is loaded.

LOG2
The name of the table for login and select random and response codes for

SWIFT network when paper tables are used. The default is DWSLOG2. If NO or 0 is specified, no such table is included.

LSKQUE
When secure login/select is used, the name of the MERVA ESA queue that is used for receiving pregenerated session keys for login and select from the USE workstation. From this queue, the session keys are read by the program DWSDLSK and loaded into the queues defined with the SKEYQ parameter of the DWSLT macro.

The default is SLSLOAD.

LTT
The name of the logical terminal table DWSLTT generated by DWSLT macros. The default is DWSLTT.

LTTQUE
The name of the MERVA ESA queue that is used for storing information of the master logical terminals and their FIN applications between a SWIFT Link termination and the next startup:

- The technology flag PT (paper tables) or SLS (secure login/select)
- The next login sequence number (LSN) or select sequence number (SSN)
- The last session number
- The last input sequence number
- The last output sequence number

Only if secure login/select (SLS) is used:

- The session key for the next LSN/SSN if available
- The card reader name
- The ICC parameters (whitelist flag, ICC kernel version, ICC set number)

The information contained in this queue cannot be displayed at a MERVA ESA display station.

The default is SLSLTT.

MF=D
Maps a DSECT of the SWIFT Link parameter module with the DSECT name of the name field.

NET
The external network identification which is used in the MSGID parameter when mapping messages for the SWIFT network using the Message Format Service. The default is S.

PDU
The maximum size of a transport layer data unit (TPDU) and of a link interface data unit (NSDU). The default is 512. The SWIFT network accepts a maximum of 512 for the TPDU. A lower value degrades the performance.

RSKQUE
When secure login/select is used, the name of the MERVA ESA queue that is used for receiving the responses for session key requests from the USE workstation. From this queue, the session keys are read by the program DWSDGPA and stored in the pertinent entry of the DWSLTT if they match the actual LSN or SSN.

If parallel SWIFT Link servers are used, each server needs its own queue for receiving the responses for session key requests. A SWIFT Link server with descriptive name SWIFTIIx builds the name of the queue by appending the letter x to the specified queue name. Refer to the *MERVA for ESA Customization Guide* for details.

The default is SLSRECV.

SWIN
The maximum size of a message to be sent to the SWIFT network. The default is 10000. For each SWIFT message type a maximum length is specified in the MERVA ESA message type table (DSLMTTT). If a formatted message is longer than specified there, the message is not sent to SWIFT.

SWOUT
The maximum size of a message to be received from the SWIFT network. The default is 11000. SWIFT allows input messages of up to 10000 bytes. To allow for additional header and trailer fields in an incoming (SWIFT output) message, 1000 bytes are added.

TLSTOR
The buffer size for the transport layer in kilobytes. The required size depends on the size of the messages to be processed. The value should be in the range of 150 to 200. The default is 200, resulting in 200KB storage above the 16MB line being required for the transport layer for each line subtask.

DWSVLINE: Defining MERVA Extended Connectivity Controlled Lines

The DWSVLINE macro is used to define the data communication lines of the SWIFT Link via X.25. Each line is defined as a separate load module with one DWSVLINE macro. Up to 30 lines can be defined.

The name of a line description module consists of the 6-character *linename* specified in the DWSPARM macro and a number from 1 to 30. The line description modules are loaded when referenced by the SWIFT Link operator commands **login** and **setlt**. They must be available in the load library. For MERVA ESA running under CICS they must be defined to CICS. The names DWSLIN1 to DWSLIN9 are supplied in the CICS program definition sample supplied by MERVA ESA. If other names are used, they must be defined to CICS.

In the SWIFT Link Logical Terminal Table (DWSLTT) a line number is assigned to each master logical terminal. This line number corresponds to the number that follows the 6-character line name.

The line numbers are used in the SWIFT Link commands **abortli**, **close**, **diva**, **dl**, **setlt**, and **xtrace**.

Generating the MERVA Extended Connectivity Parameters

Name	Operator	Operands
label	**DWSVLINE**	[**TYPE=MODULE**]
		,LNSAPNM=*cccccccccccc*
		,PLUNAME=*cccccccc*
		,SLUNAME=*cccccccc*
		[**,CUD=***cccccccccccccc*]
		[**,LDTEADR=***ccccccccccccccc*]
		[**,LINETYP=**{**LEASED**|**SWITCHED**}]
		[**,LOGMODE=***cccccccc*]
		[**,LPHONE=***nnnnnnnnnnnnnnnnnnnn*]
		[**,PHONE=**{((*num1*[,{**S**|**D**}])[,...][,(*num9*[,{**S**|**D**}])])}] {(({ **S**|**D** }[,...][,{ **S**|**D**}]) }
		[**,RDTEADR=**{*address* }] {(address1,...,address9)}
		[**,RETRY=**{**0**|*n*}]
		[**,RRUSIZE=**{**256**|*nnnn*}]
		[**,RSDELAY=**{**40**|*nnn*}]
		[**,RSDUR=**{**10**|*nn*}]
		[**,RSRETRY=**{**0**|*nn*}]
		[**,SRUSIZE=**{**256**|*nnnn*}]
		[**,TRACE=**([{**N**|**Y**}][,...])]

Programming Notes:

label
The name of the parameter table module. The name must have a length of 7 or 8 characters. The name consists of a 6-character prefix followed by a number between 1 and 30. It is mandatory.

TYPE
The usage of the parameter table. A CSECT with a name determined by *label* is generated. The default is MODULE.

LNSAPNM
The name of the local network service access point (NSAP).

This name must consist of 10 or 13 alphanumeric characters. When 10 characters are specified, the parameter value is extended by three numeric characters '000' to a value consisting of 13 characters. The 13 characters are translated into a sequence of 26 digits.

This parameter must be coded according to the SWIFT specification (Symbolic Address) for this X.25 connection.

PLUNAME
The primary logical unit (LU) name. This name must match an APPL definition in the VTAM definition library.

SLUNAME
The secondary logical unit (LU) name. This name must match a virtual LU definition in the generated MERVA Extended Connectivity.

CUD
The variable part of call user data (CUD). It consists of up to 14 hexadecimal characters (0-9, A-F).

MERVA ESA generates the first 8 characters of the CUD (fixed part) as defined by SWIFT. If this parameter is specified, additional CUD can be defined in hexadecimal notation.

This parameter must not be specified for a leased line or a switched line.

LDTEADR
The local data terminal equipment (DTE) address. It consists of up to 15 characters.

If this parameter is specified, it must be done according to the SWIFT defined possible values dependent on the physical communication line.

LINETYP
The type of the line used.

LEASED	The line is a leased line. This must also be specified for a PSPDN line. This is the default.
SWITCHED	The line is a switched line.

LOGMODE
The VTAM logon mode name. If this parameter is specified, the session parameters of the VTAM logon mode are used. The specified VTAM logon mode must meet the requirements of a communication between MERVA ESA and MERVA Extended Connectivity as described in the *MERVA for ESA Customization Guide*. The parameters RRUSIZE and SRUSIZE are ignored when LOGMODE is specified.

LPHONE
The local phone number for call back when connecting a shared PSTN port at SWIFT. It consists of up to 20 digits. The first digit must be a 9 followed by a complete telephone number according to the SWIFT specifications (that is, 9 + country code + national number).

This parameter is optional for a switched line. It is ignored for a leased line.

PHONE
A list of dial-out phone numbers, and indicators as to whether the ports at the SWIFT regional processor are to be shared or dedicated. This parameter is mandatory only for a switched line; it is ignored for a leased line.

- **For an automatic dial mode of the 37xx controller**, at least one dial-out number is required. A dial-out number consists of up to 20 characters. You can specify up to 9 dial-out numbers. If you specify more than one, they must be separated by commas and enclosed in parentheses.

  Each dial-out number can be followed by one of the following indicators:

  S For a shared PSTN port. Any part of the word SHARED (SH, SHA, SHAR, SHARE, or SHARED) can be specified instead of S. This is the default.

  D For a dedicated PSTN port. Any part of the word DEDICATED (DE, DED, DEDI, ...) can be specified instead of D.

  The dial-out number and the indicator must be separated by a comma and be enclosed in parentheses.
- **For a non-automatic dial mode of the 37xx controller**, up to 9 indicators can be specified without a dial-out number. In this case the modem or an operator has to dial. A single indicator need not be enclosed in parentheses. At least one indicator is required.

Specifications both for automatic and for non-automatic dial mode can be mixed.

The following are some examples of valid specifications (for better readability, the dial-out numbers are shown as three digit numbers.

```
PHONE=145
PHONE=(213)
PHONE=((367))
PHONE=((489,D),(588,SHARED),643,721,(823,DEDI))
PHONE=(SHARED,DEDICATED,D,S)
PHONE=((965,S),S,(102,D),D)
```

RDTEADR
The remote data terminal equipment (DTE) address or addresses. These are the addresses an X.25 connection is established to (the *called DTE addresses*).

You can specify up to 9 addresses. Each address can consist of up to 15 characters. If you specify more than one, they must be separated by commas and enclosed in parentheses. The default address is 1.

Do not specify this parameter for a switched line.

RETRY
The number of times the list of dial-out phone numbers will be retried if the first try is unsuccessful. It can be a number from 0 to 9. The default is 0. This parameter is ignored for a leased line.

RRUSIZE
The request unit (RU) size for receiving. It must be one of the following:
- 256 (this is the default)
- 512
- 1024
- 2048

This parameter is ignored, if a VTAM® logon mode name was specified.

RSDELAY
The delay in seconds before an attempt is made to reestablish a failed CBT-SWIFT connection (‘RP resynchronization’). It must be a number between 20 and 300. The default is 40, provided that the RSRETRY parameter value is greater than 0.

RSDUR
The maximal duration in minutes for the consecutive attempts to reestablish a failed CBT-SWIFT connection (‘RP resynchronization’). It must be a number between 1 and 60. The default is 10, provided that the RSRETRY parameter value is greater than 0.

RSRETRY
The number of consecutive attempts to reestablish a failed CBT-SWIFT connection (‘RP resynchronization’).

nn is a number between 0 and 40. If RSRETRY=0 is specified, RP resynchronization is not supported on that line. The default is 0.

SRUSIZE
The request unit (RU) size for sending. It must be one of the following:
- 64
- 128
- 256 (this is the default)
- 512
- 1024
- 2048
- 4096
- 8192

If a VTAM logon mode name was specified, this parameter is ignored.

TRACE
The list of trace flags.

The flags are used to switch on different traces for debugging purposes. The flags are intended to be used only by IBM service level. When MERVA ESA is running they can be switched on or off using the command XTRACE described in the *MERVA for ESA Operations Guide*.

Up to 10 flags can be specified, separated by commas and enclosed in parentheses. Flags can be omitted from within the list. This is indicated by two consecutive commas or by a list starting with a comma (the first flag is omitted). For a single trace flag, the parentheses can be omitted. The flags can be specified as follows:

N	Indicates that a trace flag is not set. This is the default.
Y	Indicates that a trace flag is set.

Mapping the MERVA Extended Connectivity Parameters

Name	Operator	Operands
[*label*]	**DWSVLINE**	**TYPE=DSECT**

Programming Notes:

label
A unique statement label according to assembler language conventions. If the label is omitted, no statement label is used.

TYPE
TYPE=DSECT must be specified for this form of the macro. All other parameters are ignored.

Chapter 5. MERVA Link: Macros

EKAASFx: Supporting the ASF Development

MERVA ESA provides a set of 3 assembler macros for MERVA Link to facilitate the development of an Application Support Filter (ASF) in the CICS and IMS environment. These macros are called:

EKAASFB	Supports the ASF begin
EKAASFC	Supports the call of the “next program”
EKAASFE	Supports the ASF end.

The CICS or IMS environment is determined by the SYSPARM parameter that must be passed to the ASSEMBLER when these macros are used. SYSPARM(CICSVSE), SYSPARM(CICSMVS), and SYSPARM(IMSMVS) identify the CICS/VSE, CICS/MVS, and the IMS environment, respectively.

With the help of these macros, an ASF without specific functions can be written in five statements as shown in Figure 3. It is, however, an executable ASF.

An ASF, running in a CICS environment, must pass the CICS command language preprocessor before it is assembled. This also applies when no CICS commands are used in the ASF.

```
EKAAF000 EKAASFB                GENERATE ASF BEGIN CODE         00001
EKAAF000 CSECT                  START ASF APPLICATION CODE      00002
         EKAASFC                CALL APPLICABLE NEXT PROGRAM    00003
         EKAASFE                GENERATE ASF END CODE           00004
         END                                                    00005
```

Figure 3. ASF without Specific Functions

EKAASFB: ASF Begin

The EKAASFB macro generates the code to start an Application Support Filter (ASF). It must be the first noncommentary statement in the source deck used to generate an ASF in assembler language.

In addition to ASF start code, it generates a set of equate symbols for the general registers 0 to 15. It also generates data definitions (DSECTs) for the MERVA Link data areas:

EKAMTPL	The MERVA Link message transfer (MT) parameter list including equate symbols for all service primitive identifiers. EKAMTPL is based on RMTP (general register 5).
EKAPT	The Partner Table. EKAPT is based on RPT (general register 6).
EKAPDUD	The MERVA Link PDU definitions. EKAPDUD is based on RPDU (general register 4).

Finally, this macro generates data definitions for fields in the ASF work area. These fields are used by the code generated by the ASF development support macros. Additional work area fields can be defined by adding the data definition statements for these fields following the EKAASFB macro statement. The end of the work area definition statements is specified by a CSECT statement that starts the specific ASF code.

The following general registers have been set by the ASF begin code generated by EKAASFB, when the specific ASF code starts:

R4	Pointer to the Body Part Data Segment
R5	Pointer to the EKAMTPL
R6	Pointer to the Partner Table header
R9	CICS EIB pointer (if applicable)
R12	ASF base register
R13	Pointer to the RSA followed by the ASF work area

The ASF work area, which can be extended by user-defined fields, is identified by "*ASFWA*" in the field WAID at displacement X'F8' from the beginning of the RSA (dynamic storage in the CICS environment). The pointer to the EKAMTPL is saved in the field WAMTPL of the ASF work area.

Note: The EKAASFB macro statement can be specified only once in a source deck.

Name	Operator	Operands
[*label*]	**EKAASFB**	[**PRINT=**{**ON**|**OFF**}]

Programming Notes:

label
An optional assembler label for this statement. If it is specified, an assembled CSECT statement with the specified label is generated. If the label is not specified, the CSECT statement must have been coded earlier in the source deck.

PRINT
Whether the code for the ASF begin should be printed:

ON	The begin code and the data definitions are to be printed. This is the default.
OFF	The begin code and the data definitions are not printed.

EKAASFC: Call the Next Program

The EKAASFC macro generates code to evaluate the name of the next program and to call it.

The ASF must identify its load module name to the code generated by EKAASFC. The load module name can be different from the source module name or the CSECT name. By default, the first 7 characters of the CSECT name are used as the name of the ASF. If the default is not used, the ASF load module name must be explicitly specified in parameter MYNAME.

Note: The EKAASFC macro statement can be specified only once in a source deck.

Name	Operator	Operands
	EKAASFC	[**MYNAME=***asfname*]

Programming Notes:

MYNAME
The load module name of this ASF as it is also specified in the applicable Partner Table (PT) ASP entry. This name is used by the generated code to find

the position of this ASF in the ASF list in the PT ASP entry. With the help of this position and the current service primitive identifier, it can evaluate the name of the next program. The default is a name consisting of the first 7 characters of the CSECT name.

EKAASFE: ASF End

The EKAASFE macro generates code to define the end of the ASF work area, and generates code to return to the caller.

Note: The EKAASFE macro statement can be specified only once in an ASF source deck.

Name	Operator	Operands
	`EKAASFE`	

Programming Notes: None.

EKAPT: Defining the Partner Table

The Partner Table (PT) is defined by coding EKAPT statements. The EKAPT statements generate the PT header, PT entries, the PT trailer, and a data description of all these parts of a PT. The EKAPT statement can have the following types:

TYPE=INITIAL	Defines the Partner Table header.
TYPE=ASP	Defines an Application Support Process (ASP).
TYPE=MTP	Defines a Message Transfer Process (MTP).
TYPE=ASPGS	Defines an ASP group boundary.
TYPE=SCP	Defines a System Control Process (SCP).
TYPE=SCPGS	Defines an SCP group boundary.
TYPE=FINAL	Defines the end of the table.
TYPE=DSECT	Generates an assembler DSECT of the PT.

The EKAPT macro has parameters or subparameters that are not intended for customer use. Only those EKAPT parameters and subparameters described in the following should be used to request the services of MERVA Link.

The EKAPT statements are assembler macros. Therefore, the coding rules for assembler macros apply. The following describes the parameters of the different EKAPT statement types.

Generating the PT Header

The EKAPT TYPE=INITIAL statement generates the PT header, which contains global MERVA Link information. It must be the first statement in the source deck used to generate the PT.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=INITIAL** **,NODE=***nodename* [**,MONITOR=**(*amid*[,{**5**|*tiv*}])] [**,OPMSG=**{**DSLMSGT**|*opmsgtab*}] [**,SP=**{**1**|*spnr*}] [**,TEST=**{**OFF**|**EXCEPTION**|**ON**}] [**,TRACE=**([*dsid*][,{**FULL**|**WEAK**}])]

Programming Notes:

label
An optional assembler label for this statement. It has no meaning for the table header.

TYPE=INITIAL
The table definition begins and the table header must be generated.

NODE
The name of the local MERVA Link node. This name must be unique within

the MERVA Link Message Transfer System (MTS). Together with an ASP name it builds the unique address of an application within the MERVA Link Message Handling System.

MONITOR
Controls the MERVA Link ASP Monitor:

- The first subparameter (*amid*) specifies the CICS transaction identifier of the ASP Monitor. There is no default value provided for the ASP Monitor transaction identifier. The MERVA Link sample transaction identifier for the ASP Monitor is EKAM. The name of the ASP Monitor program is EKAAM10.
- The second subparameter (*tiv*) specifies the ASP Monitor restart time interval in minutes (00 to 59). ASP Monitor restart time interval 00 means that the ASP Monitor is not restarted. The default value for the restart time interval is 05 minutes if the ASP Monitor transaction identifier is specified and 00 if it is omitted.

Note: The MERVA Link ASP Monitor is not supported in an IMS environment.

OPMSG
The name of the Operator Message Table that contains the operator messages issued by the MERVA Link programs executing in the Message Transfer Layer. The messages issued by the MERVA Link programs executing in the Application Support Layer must be contained in the active MERVA ESA Message Table. The name of the message table is specified in the DSLPRM module. The default is DSLMSGT.

SP An MVS main storage subpool number. A subset of the main storage areas used by MERVA Link programs in the MERVA IMS environment is acquired from the specified subpool, and released at the end of a MERVA Link task by freeing the entire subpool.

The subpool number can be a number between 1 and 127. Refer to the MVS documentation for restrictions concerning storage subpools that are eligible for subpool FREEMAIN. Subpool number 2 can be used, for example, if the default subpool 1 must not be used. The default is 1.

This parameter is not applicable to the CICS environment.

TEST
Whether the MERVA Link runs in a test environment. In the MERVA Link test environment a MERVA Link task is terminated with a storage dump. This allows detailed problem analysis in situations considered by MERVA Link to be normal or exceptional. For error conditions a storage dump is issued regardless of this parameter:

OFF	The test environment is not activated. This is the default.
EXCEPTION	The test environment is activated only for conditions considered by MERVA Link as exceptional.
ON	The test environment is activated.

TRACE
Controls the MERVA Link CICS Conversation Trace:

- The trace identifier (*dsid*) applies to all MTPs. For a sending MTP, the trace identifier can be overridden in a specific EKAPT TYPE=MTP entry, or the conversation trace can be disabled for a specific sending MTP. For details refer to the TRACE parameter in the EKAPT TYPE=MTP macro.

For the MERVA Link running under CICS, *dsid* specifies the identifier of an extrapartition transient data queue that represents the conversation trace data set. It can consist of up to 4 characters and must also be defined in a DFHDCT TYPE=EXTRA entry in the CICS destination control table DFHDCT. This DFHDCT entry must point to a DFHDCT TYPE=SDSCI entry that describes the actual conversation trace data set.

- The second subparameter specifies the conversation trace type. It applies to all sending and receiving MTPs:

FULL	Both control and application data must be traced. This is the default.
WEAK	Only control data must be traced and application data is not shown in the conversation trace.

If the TRACE parameter is omitted, there is no default conversation trace data set for any sending MTP, and no conversation trace is written for any receiving MTP.

Note: The MERVA Link Conversation Trace is not supported in an IMS environment.

Defining an Application Support Process

This statement describes an ASP for both outgoing messages or reports and incoming messages or reports.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=ASP** **,CONTROL=**(*ctlq*[,*window*]) **,DEST=**(*nodename*,*aspname*) **,MTP=***mtpname* **,NAME=**(*aspname*[,'*appl free-form name*']) **,SENDQC=**([*hiprtyq*][,*normalq*][,*loprtyq*]) [**,ACCEPT=**{**OK**|**WARNING**}] [**,CONFIRM=**{**NO**|**NON**|**ALL**}] [**,DSLPRM=**{**DSLPRM**|*prmname*}] [**,FILTER=**(*filter1*[,*filter2*[,*filter3*]][,{**ASF**|**ALL**}])] [**,FORMAT=**{([**NET**][,{**DSLEXIT**|*mtypefield*}][,{*****|*nfid*}])}] {(**MCB**,*mcbname*[,{*****|*nfid*}]) } {**QUEUE** } [**,IMROUTE=**(**RT**,*queue*)] [**,IPRECOV=**{**MANUAL**|**AUTO**}] [**,IRROUTE=**{(**ACK**,*queue*[,**CTLQ**])}] {(**RT**,*queue*) } [**,JOURNAL=**([**OM**][,**IR**][,**IM**])] [**,MFSEXIT=***exitno*] [**,OMROUTE=**{(**Q**,*queue*) }] {(**RT**,*queue*)} {**DELETE** } [**,SECKM=***modifier*] [**,SECURE=**{**NO**|**AUT**|**ENCRYPT**}] [**,START=**([{**AUTO**|**OPERATOR**|**RETRY**}][,{**ENABLED**|**DISABLED**}])]

Name	Operator	Operands
		[,**TRAN**={**EKAS**|*tranid*}]

Programming Notes:

label
An optional assembler label for this statement. It has no meaning for this table entry.

TYPE=ASP
An ASP definition begins and an ASP entry must be generated.

CONTROL
Message Integrity Protocol (MIP) control information:

- *ctlq* specifies the name of a MERVA ESA queue that is used by this ASP as application control queue to support MIP. The application control queue must be unique with respect to all other application control queue names and all send queue names.
- *window* specifies the window size to be used by this ASP when a *last confirmed control message* is generated by MERVA Link. It determines the maximum number of messages that are transmitted to the receiving application before a confirmation of the transfer process is requested. The window size can be a number from 1 to 999, and can be changed by a MERVA Link operator. The default is 10.

  The window size specification is ignored if this ASP is associated with an MTP of type ISC, because an ISC connection does not support message windows. This means a message transfer confirmation is requested for every single message.

DEST
The destination application for all outgoing messages of this application.

The destination has two parts: *nodename* is the destination node name, *aspname* is the destination ASP (application) name. Both names can have a maximum length of 8 characters.

MTP
The internal name of an MTP as defined in a PT entry of TYPE=MTP in this PT. This MTP is used to transfer the message to its destination.

This parameter connects the ASP and the specified MTP for outgoing messages, and for incoming messages that do not specify the recipient MTP name.

NAME
The name of the ASP and, optionally, its free-form name.

The ASP name is also considered as the name of the application as it is known by the MERVA Link Message Handling System. It must be agreed between the originating and receiving applications. The application name is part of any originator or recipient address used by the MERVA Link MTS to transfer a message from its originator to the intended recipient. The ASP name must be unique with respect to all other ASP names, all send queue names, and all internal MTP names in this PT.

aspname consists of up to 8 characters starting with a letter. To avoid conflicts with MERVA Link internal names, do not start an ASP name with the letters EKA.

appl free-form name specifies the free-form name of the application. If it is specified, the free-form name is conveyed as part of the originator descriptor in the message heading to the recipient. The receiver should understand the meaning of the free-form name.

appl free-form name can consist of up to 60 characters. All characters, including blanks, are valid.

SENDQC
The names of the queues that build the MERVA Link send queue cluster. The send queue names must be unique with respect to all other send queue names, all application control queue names, and all ASP names in this PT.

A sending ASP started by MERVA ESA obtains the name of a send queue from MERVA ESA in the Terminal User Control Block (TUCB). The applicable ASP entry is determined by matching this queue name with one of the queue names in this parameter.

This ASP processes the messages in all specified send queues in the specified sequence (all messages in queue 1, the next message in queue 2, if there is no message in queue 1, and the next message in queue 3, if there is no message in queue 1 and 2).

The queue names may consist of up to 8 characters as specified in the MERVA ESA function table (DSLFNTT). At least one of the three queue names must be specified.

ACCEPT
Whether this ASP accepts incoming application messages with a warning (WARNING), or whether processing of an incoming message must be fully correct (OK). The default is OK. A warning can be issued when an incoming message is formatted from net to TOF format, or by the user exit which is called before an incoming message is routed to the target queue or queues.

A message that is not accepted is reported as undeliverable to the sending ASP. Automatic delivery error handling may apply at the sending side to recover an ASP from an undeliverable message (see IPRECOV=AUTO). A message that is accepted is reported as delivered without error to the sending ASP. The warning information is not returned to the sending application; however, it is passed to the receiving application.

CONFIRM
Whether receipt confirmation must be requested from the recipient application.

FMT/ESA uses this parameter to decide where to generate the SWIFT acknowledgment and whether to generate a SWIFT delivery notification message.

NO Neither receipt confirmation nor nonreceipt confirmation is requested. This is the default.

For FMT/ESA:

- The SWIFT acknowledgment is generated in the message sending MERVA ESA.
- A SWIFT delivery notification is not generated.

NON Only nonreceipt confirmation is requested.

For FMT/ESA:

- The SWIFT acknowledgment is generated in the message sending MERVA ESA.

- A SWIFT delivery notification is generated in the receiving MERVA ESA if it is requested by the received SWIFT input message.

ALL Receipt confirmation and nonreceipt confirmation is requested.

For FMT/ESA:

- The SWIFT acknowledgment is generated in the message receiving MERVA ESA.
- A SWIFT Delivery Notification is generated in the receiving MERVA ESA if it is requested by the received SWIFT input message.

The request for receipt confirmation specified in this parameter is conveyed to the receiving application as control information. It does not imply, however, that the receiving application sends a receipt confirmation.

DSLPRM

The name of the MERVA ESA parameter table which must be used by this ASP when it receives messages. A sending ASP always uses the MERVA ESA parameter table named DSLPRM.

The MERVA ESA parameter table specified in this operand may describe a MERVA ESA system which is different from the MERVA ESA system described by DSLPRM. The effect is, that messages from one MERVA ESA system are sent to the partner, and messages from that partner are delivered to the other MERVA ESA system.

The default is DSLPRM, which means that the ASP processes messages from and delivers messages to the same MERVA ESA system.

FILTER

The names of Application Support Filters (ASF), customer-written programs located at the boundary between the Application Support Layer and the Message Transfer Layer. The ASF named by *filter1* is called by an ASP when a SUBMIT.Request is processed. *filter1* calls *filter2*, and so on. The last ASF in this list calls the Message Transfer Service Processor.

The last ASF in the list is called by the Message Transfer Service Processor when a DELIVER.Indication is processed. This ASF calls the preceding ASF, and so on. The first ASF in the list calls the applicable ASP. The name of the applicable receiving Application Support Process is EKAAR10 by default.

If the subparameter ALL is specified, the ASFs are activated for every event (primary or secondary service primitive) that crosses the MTL boundary, not just for SUBMIT.Requests and DELIVER.Indications.

Examples of tasks that may be performed by an ASF are:

- Authentication of the message using a proprietary algorithm
- Encryption/decryption of the message text
- Compression/expansion of the message text

If this parameter is omitted, the MERVA Link Message Transfer Service Processor is directly called by the sending ASP, and the receiving Application Support Processor is directly called by the Message Transfer Service Processor.

filter1 to *filter3* may consist of up to 8 characters starting with a letter. The corresponding programs must be accessible by an ASP and by the Message Transfer Service Processor. For MERVA Link under CICS, these programs must be defined in a CICS PROGRAM definition.

The subparameters *filter2* and *filter3* are ignored, if subparameter *filter1* is not specified. The subparameter *filter3* is ignored if subparameter *filter2* is not specified.

FORMAT

The transfer format of the messages that are processed by this ASP.

NET	MERVA ESA line (network) format. The message type is specified in a control field in the MERVA ESA TOF. This is the default. The subparameter *mtypefield* is the name of the control field. The default is DSLEXIT (this is the standard MERVA ESA control field that contains the message type).
MCB	Messages are transferred in MERVA ESA line (network) format. *mcbname* is the name of the MCB (in the form of a message type) used to format the message to external line format.
QUEUE	Messages are transferred in MERVA ESA queue format.

The subparameter *nfid*, which applies to types NET and MCB, specifies the line-format identifier to be used when a MERVA ESA message is formatted to a MERVA ESA external line format. It is a single character of the set {A,..., Z, 0,..., 9, *, #}, where:

- The character * specifies that the first letter of the message type must be used as line-format identifier. This is the default.
- The character # specifies that the line-format identifier contained in the MERVA Link control field EKANETID must be used. If this field is not defined or empty, the first letter of the message type is used as line-format identifier.

The specified external line format must be defined in all MERVA ESA MCBs that describe the messages that are processed by this ASP.

IMROUTE

Local routing parameters for incoming application messages. Incoming application messages are routed to one or two received message queues to deliver these messages to the receiving application.

The first subparameter must be RT (for ″routing table″). It defines that the message must be routed as specified by a MERVA ESA routing table. *queue* is the name of a queue associated with that routing table.

The routing table must specify the application control queue as one of the twelve possible target queues.

If this parameter is omitted, the message is routed as specified by the MERVA ESA routing table associated with the application control queue (see parameter CONTROL).

IPRECOV

How this ASP should be recovered from an undeliverable message:

MANUAL

Manually. Manual recovery can be performed via the MERVA Link operator command **iprecov**. This is the default.

AUTO
: Automatically. Automatic recovery of an ASP from an undeliverable message requires specific information from the receiving partner system in the non-delivery report.

MERVA Link of MERVA ESA as receiving partner system provides information for automatic IP message recovery in the following error situations:

- Formatting an incoming message from net to TOF format is not successful.
- The user exit that is called before the incoming message is routed to the destination queue(s) reports an error.

In all other error situations, automatic IP message recovery is not applicable. Also, other receiving systems can have their own rules to provide information for automatic IP message recovery.

If automatic recovery is not specified, an ASP, which fails to get a message delivered to the destination application, becomes inoperable. In that situation, the partner systems must be modified to allow a successful delivery of the message, or the sending ASP must be manually recovered.

IRROUTE
Local routing parameters for incoming status reports. An incoming status report results in the local routing of an acknowledgment message, or in the local routing of an acknowledged message (reported message with status information).

The first subparameter is one of the following values:

RT The report must be routed as specified by a MERVA ESA routing table. *queue* is a queue associated with the routing table.

The routing table must specify the application control queue as one of the twelve possible target queues.

ACK The report must be correlated with the reported message in the ACK wait queue specified by *queue*. The message must be updated with status information from the report, and the updated message must then be routed.

If the CTLQ subparameter has not been specified, the updated message is routed as specified by the MERVA ESA routing table associated with that ACK wait queue.

If the CTLQ subparameter has been specified, the updated message is routed as specified by the MERVA ESA routing table associated with the application control queue (see parameter CONTROL).

The routing table must specify the application control queue as one of the twelve possible target queues.

If this parameter is omitted, the report is routed as specified by the MERVA ESA routing table associated with the application control queue (see parameter CONTROL).

JOURNAL
Whether any of the three message categories of outgoing message, incoming report, and incoming application message must be written to the MERVA ESA journal:

OM	Indicates that outgoing messages (application messages as well as acknowledgment messages) must be written to the MERVA ESA journal.
IR	Indicates that incoming reports (acknowledgment messages as well as delivery reports) must be written to the MERVA ESA journal.
IM	Indicates that incoming application messages must be written to the MERVA ESA journal.

If any of the subparameters is omitted, messages of the corresponding category are not written to the MERVA ESA journal.

MFSEXIT

The number of a MERVA ESA MFS user exit program.

The MERVA ESA MFS user exit program is called to:

- Check whether a "ready-to-send" message should be processed
- Identify an outgoing message as an application message or acknowledgment message
- Handle an incoming application message
- Handle an incoming acknowledgment message
- Handle a confirmed outgoing message
- Handle a recovered message

exitno must be a number from 1 to 32767. Corresponding definitions must be provided in the MERVA ESA MFS program table.

MFS user exits must be numbered according to rules established by MERVA ESA. If this parameter is omitted or if *exitno* is 0, no user exit is called.

OMROUTE

Local routing parameters for outgoing messages. Outgoing messages are locally routed when the confirmation of the successful delivery to the intended receiving application has been received.

The first subparameter must be one of the following values:

Q	MERVA ESA queue: The message must be routed to a MERVA ESA queue, specified as *queue*.
RT	The message must be routed as specified by a MERVA ESA Routing Table. *queue* is a queue associated with that routing table.
DELETE	The message is not routed to another local queue, but deleted.

The *queue* subparameter can consist of up to 8 characters.

If the parameter is omitted the message is routed as specified by the MERVA ESA routing table associated with the application control queue (see parameter CONTROL).

SECKM

The message text security key modifier that must be used by the MERVA Link message text encryption and authentication functions. If a security key modifier is specified, the same modifier must be specified for the partner ASP. Otherwise, the partner ASP cannot decrypt and authenticate the message text. MERVA Link does not transmit the security key modifier to the partner system.

modifier is a string of one to eight alphanumeric characters. This set of characters is not restricted if the partner system is also a MERVA ESA system. If the partner system is MERVA AIX or MERVA OS/2, the set of characters that can be used for the security key modifier is restricted to 0 - 9 and upper- and lowercase A - I.

If this parameter is omitted, the message text security key is not modified. The security key modifier can be used only if the partner MERVA Link system supports MERVA Link security key modification.

SECURE
The message text security provisions to be made by MERVA Link for all messages processed by this application:

NO	No security provisions made. This is the default.
AUT	An outgoing message must be authenticated using the MERVA Link proprietary authentication algorithm, and the authenticity of incoming messages must be checked.
ENCRYPT	An outgoing message must be authenticated and encrypted using MERVA Link proprietary algorithms, and the authenticity of incoming messages must be checked as soon as they have been decrypted.

START
ASP start options for a sending and for a receiving ASP:

- The first subparameter specifies how the ASP starts processing of outgoing messages:

AUTO	Process outgoing messages automatically when a message has been routed to one of its send queues. The ASP must be operable.
RETRY	Process outgoing messages automatically when a message has been routed to one of its send queues. If the ASP is inoperable, it tries to become operable again; however, it does this only once per minute.
OPERATOR	Process outgoing messages only if a MERVA ESA operator command **sf** or a MERVA Link operator command **astart** or **kickoff** is entered.

- The second subparameter specifies whether the ASP must process incoming messages when MERVA ESA is started:

ENABLED	Incoming messages are processed.
DISABLED	Incoming messages are not processed.

  This status can be changed in the CICS environment using the MERVA ESA operator commands **enable** and **disable**. These commands may not have the desired effect in the IMS environment.

The default is (AUTO,ENABLED).

TRAN
The transaction identifier of the sending ASP. The name of the sending Application Support Program is EKAAS10. It is recommended to specify this parameter in the IMS environment because the sending ASP transaction code must be unique for each ASP defined in this PT.

For outbound message processing, the sending ASP task is started by MERVA ESA using the transaction identifier specified in the MERVA ESA function table (DSLFNTT) entries for all send queues of this ASP (see parameter SENDQC).

tranid can consist of up to 8 characters. However, for MERVA Link under CICS, only the first 4 characters are significant. The default is EKAS.

In a CICS environment, the default transaction identifier can be used for all MERVA Link applications. However, in an IMS environment, you must assign a unique transaction identifier for each sending ASP. This parameter is therefore mandatory in the IMS environment for all ASPs associated with an ISC type MTP. The uniqueness of the sending ASP transaction identifiers is checked if SYSPARM=IMSMVS is specified for the assembly of the PT. The default (TRAN=EKAS) is not included in this check for uniqueness. The TRAN parameter can be omitted in the definition of ASPs associated with APPC type MTPs in the IMS environment, however this is not recommended.

The transaction identifier specified in this parameter must match the transaction identifier specified in the MERVA ESA function table (DSLFNTT) entries for all send queues of this ASP (see parameter SENDQC).

Defining a Message Transfer Process

This statement describes a Message Transfer Process (MTP). It applies to both a sending and a receiving MTP. The MTP characteristics specified in this PT entry include the relevant identifiers of the remote partner.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=MTP**
		,ASP=*aspname*
		,NAME=(*mtpname*[,*extmtp*])
		[,**CONNECT=**([*r1*][,*r2*][,*r3*])]
		[,**DEST=**{**BTB** }]
		{(*sys_idnt*[,*tp_id*][,*profile*])}
		{(*sym_dest*[,*tp_id*][,*logmode*])}
		[,**LINK=**{**(APPC,***sys_idnt*[,*profile*])}]
		{**(APPC,***lu_name*[,*logmode*]) }
		{**BTB** }
		[,**LOCLU=***lu_name*]
		[,**MTPCI=**(,*mtprname*)]
		[,**PARTNER=**([*extmtp*][,*tp_id*])]
		[,**TRACE=NONE**|*dsid*]

Programming Notes:

label
An optional assembler label for this statement. It has no meaning for this table entry.

TYPE=MTP
An MTP definition begins and an MTP entry must be generated.

ASP
The name of an ASP as defined in a PT entry of TYPE=ASP. All incoming messages and reports are delivered to this ASP. This parameter connects the MTP and the specified ASP for incoming messages that specify the recipient MTP name.

NAME
The internal and the external name of the MTP:

mtpname
The internal MTP name. It can consist of up to 8 characters starting with a letter.

The internal MTP name is specified by an ASP to identify the MTP that handles the message transfer. It must be unique with respect to all other internal MTP names and all ASP names in this PT.

extmtp The external MTP name. It can consist of up to 8 characters.

If specified, the external MTP name is passed to the partner MTP for originator identification and verification, and must be received from the partner MTP for recipient identification and verification. The name or its absence must be agreed between the cooperating MTPs.

CONNECT
Requests specific control functions from a sending MTP concerning an APPC connection in the CICS environment. This parameter is applicable to the CICS APPC environment only. It has no effect in the APPC/IMS and MERVA Link back-to-back environments.

The values **ACQ**, **QUEUE**, and **REL** can be specified for *r1*, *r2* and *r3*.

ACQ This value asks an MTP to acquire a connection to the partner system if the connection is not already available. The CICS System Programming command issued by the MTP reads **EXEC CICS SET CONN(***system***) ACQUIRED**. Processing depends on the specification of the QUEUE value:

- If ACQ is specified and QUEUE is not specified, the MTP checks the status of the connection to the partner system and issues an ALLOCATE command (with the NOQUEUE option) independent of the connection status. If the ALLOCATE session command fails, the MTP issues the CICS command to acquire a connection to the partner system, and reports the ALLOCATE command error. The ALLOCATE command may be successful when the MTP is called the next time.
- If both ACQ and QUEUE values are specified, the MTP checks the status of the connection to the partner system and, if the connection is not available, it issues the CICS command to acquire a connection to the partner system and waits up to 10 seconds to allow CICS to satisfy the request. An ALLOCATE session command (without the NOQUEUE option) is issued after this, independent of the connection status. If the ALLOCATE session command fails, the MTP reports the ALLOCATE command error.

QUEUE
The ALLOCATE session command is issued without the NOQUEUE option when this value is specified. If all available sessions are busy, CICS queues the task until a session becomes available to satisfy the

ALLOCATE request. CICS does not, however, bind any session as response to an ALLOCATE command (with or without the NOQUEUE option).

When this value is not specified, the ALLOCATE command is issued with the NOQUEUE option. A session is allocated only if a contention winner session is immediately available.

When the QUEUE value is specified with the ACQ value, the function of the ACQ option is modified.

REL After a FREE conversation command, the MTP issues the CICS commands **EXEC CICS DELAY INTERVAL(5)** and **EXEC CICS SET CONN(***system***) RELEASED** and terminates. An ALLOCATE command may fail when the MTP is called the next time. So you are recommended to specify the REL value with the ACQ and QUEUE values.

The delay of five seconds allows a sending MTP in the partner system to start a conversation for message transmission. When the release connection command is issued, all unused sessions are immediately released. A session which is used by a conversation is released by CICS as soon as the conversation ends.

Either of these values can be specified in the CONNECT parameter, or all three values can be specified in any sequence.

The **EXEC CICS SET** command is a CICS SP (system programming) command. CICS security checking applies for CICS SP commands. Additional security requirements may therefore apply to the MERVA Link Send Task (sample transaction identifier is EKAS) if you specify the CONNECT parameter in an MTP definition. For more information refer to the *CICS/ESA System Programming Reference*.

If the CONNECT parameter is omitted, an APPC session is allocated with the CICS NOQUEUE option, and an APPC connection is not explicitly acquired nor is it released by the MTP.

DEST

Provides for specifying the parameters of a synchronous connection to a partner system. The DEST operand provides an alternative to the LINK operand. The LINK and DEST operands are mutually exclusive.

DEST=BTB

This identifies the MTP as a synchronous back-to-back MTP in the CICS and IMS environments. The partner system is the local system. Messages handled by this MTP are synchronously sent and received by the local MERVA Link node. If both the DEST and the LINK operands are missing from an EKAPT TYPE=MTP statement, this is the default.

DEST=*sys_idnt*

In the MERVA Link **CICS** environment, an APPC connection can be identified by one of the following:

- If *sys_idnt* consists of 1 to 4 characters, it is the remote system identification as specified in the CONNECTION parameter of a CICS CONNECTION definition. Subparameter *tp_id* is the identifier of the transaction program (TP) that handles inbound MERVA Link messages in the partner system. The subparameter *profile* is the name of an APPC profile defined in CICS. An APPC profile can be specified to select a specific mode set for the APPC session to the partner system.

- If *sys_idnt* consists of more than 4 characters, it is a symbolic destination name as specified in the PARTNER parameter of a CICS PARTNER definition. The subparameters *tp_id* and *profile* are ignored by an MTP in the CICS environment if a symbolic destination name is specified. Instead, they are retrieved from the CICS PARTNER definition.

DEST=*sym_dest*
In the MERVA Link **IMS** environment, an APPC connection can be identified by the symbolic destination name of the partner system, as defined in an APPC/MVS side information profile. The partner LU name, the SNA mode name, and the partner TP name are contained in an APPC/MVS side information profile (SI Profile). The SNA mode and TP names specified in the SI profile can, however, be overwritten by the subparameters *logmode* and *tp_id*, respectively.

A *tp_id* subparameter specified in a PARTNER operand of an EKAPT TYPE=MTP statement overwrites a *tp_id* subparameter specified in a DEST operand of the same statement.

LINK
Defines the connection to the partner system. The first subparameter defines the communication method. It can be **APPC** for a synchronous LU 6.2 connection or **BTB** for a synchronous back-to-back connection. The support of an asynchronous LU 6.1 connection **LINK=(ISC,..)** has been dropped.

APPC In the MERVA Link **CICS** environment, the *sys_idnt* and *profile* subparameters are applicable. For more information about these subparameters, refer to the corresponding subparameters of the **DEST** parameter in the CICS environment.

In the MERVA Link **IMS** environment, the *lu_name* and *logmode* subparameters are applicable. Both subparameters consist of up to 8 characters. The subparameter *lu_name* is the name of the APPC LU in the remote system. The optional *logmode* subparameter identifies a VTAM LOGMODE table entry.

BTB A synchronous back-to-back MTP does not require the specification of a partner system. The partner system is the local system. CICS or IMS data communication services are not used for this type of MERVA Link connection.

LOCLU
The name of an APPC/IMS logical unit (LU) that must be used for outbound conversations. It can consist of up to 8 characters, and must identify an APPC/MVS LU that is accessible from an IMS Message Processing Region (MPR).

This parameter is applicable to the APPC/IMS environment only. It must be specified only if the APPC/IMS Base LU that is associated with an IMS MPR must not be used. If this parameter is omitted, the APPC/IMS Base LU (an APPC/MVS Base LU that is associated with the APPC/IMS transaction scheduler) is used for outbound conversations.

MTPCI

MTP Control Information. This parameter is required only if the default standard MERVA Link sending Message Transfer Program must not be used by this MTP.

, The first subparameter of this operand is ignored. It must, however, be represented by a comma.

mtprname
The name of a sending Message Transfer Program. It can consist of up to 8 characters starting with a letter.

The default Sending Message Transfer Program name depends on MTP characteristics:

- **EKATM10** is used for a back-to-back MTP in all DC environments.
- **EKATPO1** is used for an APPC MTP in the APPC/IMS environment.
- **EKATS10** is used for an APPC MTP in the CICS environment.

PARTNER

Information about the partner MTP. All subparameters consist of up to 8 characters. The PARTNER subparameters are:

extmtp The external name of the partner MTP. All external partner MTP names must be unique within this PT.

If specified, the external MTP name is passed to the partner MTP for recipient identification and verification, and must be received from the partner MTP for originator identification and verification. This name or its absence must be agreed between the cooperating MTPs.

tp_id The APPC partner transaction program identifier. It is used in the CICS CONNECT PROCESS command or in the APPC/MVS ALLOCATE request to connect the front end and the back end processes. Only 4 characters are significant if the partner process executes under CICS, as *tp_id* is a CICS transaction code for the remote receiving MTP. The receiving MTP transaction code in the MERVA Link sample is EKAR.

For APPC/MVS and APPC/IMS receiving environments *tp_id* is a TPNAME in an APPC/MVS TP-Profile definition. The MERVA Link sample TPNAMEs in these environments are EKAR and EKARI410, respectively.

If the partner system is MERVA Link OS/2, the name EKAOSVR must be specified as *tp_id*. The sample *tp_id* for a MERVA Link AIX partner system is EKAR1.

The *tp_id* subparameter does not apply to a **BTB** connection.

TRACE

The identifier of the conversation trace data set that must be used with this MTP when it sends messages to its partner MTP. This identifier overrides the identifier specified in the PT header, if any, and does not apply to the MTP when it receives messages from its partner.

For MERVA Link running under CICS *dsid* specifies the identifier of an extrapartition transient data queue that represents the conversation trace data set. It can consist of up to 4 characters. This identifier must also be defined in a DFHDCT TYPE=EXTRA entry in the CICS destination control table DFHDCT. This entry points to a DFHDCT TYPE=SDSCI entry that describes the actual conversation trace data set.

TRACE=NONE defines that no conversation trace is written for the sending MTP.

Note: The MERVA Link conversation trace is not supported in an IMS environment.

Generating an ASP Group Separator

This statement defines the boundary between two groups of ASP entries in this PT. ASP groups may be defined to structure the ASP list displayed by the MERVA System Control Facility.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=**{**ASPGS**|**DUMMY**} [**,GRPSEP=**'*c*']

Programming Notes:

label
: An optional assembler label for this statement. It has no meaning for this table entry.

TYPE={ASPGS|DUMMY}
: An ASP group separator (formerly called a PT dummy entry) must be generated, that defines the boundary between two ASP groups. For compatibility reasons, TYPE=DUMMY is still supported.

GRPSEP
: The character used to form an ASP group separator line in the ASP list displayed by the MERVA System Control Facility.

If this parameter is omitted, a line consisting of dashes is displayed.

Defining a System Control Process

This statement describes an SCP for both active and passive partner MERVA system control.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=SCP** **,NAME=**([*ps-name*][,*nodename*]) [**,DEST=**{([*sys_idnt*][,*tp_id*][,*profile*])}] {([*sym_dest*][,*tp_id*][,*logmode*])} [**,DSLTERM=**{<u>**NO**</u>|**YES**}] [**,LINK=**{(,*sys_idnt*[,*profile*])}] {(,*lu_name*[,*logmode*]) } [**,LOCLU=***lu_name*] [**,LOPER=**{**ALL**|*name1*|(*name1*,*name2*[,*name3*])}] [**,PARTNER=**(,{<u>**EKAC**</u>|*tp_id*})] [**,POPER=**{**ALL**|*name1*|(*name1*,*name2*[,*name3*])}]

Programming Notes:

label
An optional assembler label for this statement. It has no meaning for this table entry.

TYPE=SCP
An SCP definition begins and an SCP entry must be generated.

NAME
The name(s) of the partner system. It is either only the partner system node name, or a local nickname for the partner system and the partner system node name. Either of these names can be used when the partner system must be identified in a **node** command.

ps-name and *nodename* consist of up to 8 characters starting with a letter. To avoid conflicts with MERVA Link internal names, you must not start an SCP name with the letters EKA.

DEST
Provides for specifying the parameters of a connection to a partner system. The DEST operand provides an alternative to the LINK operand. The LINK and DEST operands are mutually exclusive.

DEST=*sys_idnt*
In the MERVA Link **CICS** environment, the meaning of *sys_idnt* depends on its length:

- If *sys_idnt* consists of 1 to 4 characters, it is the remote system identification as specified in the CONNECTION parameter of a CICS CONNECTION definition. Subparameter *tp_id* is the identifier of the transaction program (TP) that handles inbound MSC requests in the partner system. Subparameter *profile* is the name of an APPC profile defined in CICS. An APPC profile can be specified to select a specific mode set for the APPC session to the partner system.
- If *sys_idnt* consists of more than 4 characters, it is a symbolic destination name as specified in the PARTNER parameter of a CICS PARTNER definition. Subparameter *tp_id* is the identifier of the transaction program (TP) that handles inbound MSC requests in the partner system. It, or the default TP ID **EKAC**, overwrites the Partner TP information in the CICS PARTNER definition. Subparameter *profile* is the name of an APPC profile defined in CICS. If it is specified, it overwrites the Profile information in the CICS PARTNER definition.

DEST=*sym_dest*
In the MERVA Link **IMS** environment, a connection can be identified by the symbolic destination name of the partner system defined in an APPC/MVS side information profile. The partner LU name, the SNA mode name, and the partner TP name are contained in an APPC/MVS side information profile (SI Profile). The SNA mode and TP names specified in the SI profile can, however, be overwritten by the subparameters *logmode* and *tp_id*, respectively.

If both the DEST and the LINK operands are missing from an EKAPT TYPE=SCP statement, the partner system node name specified for the NAME parameter is the default. A default value for the *tp_id* subparameter is not provided for the DEST operand in the MERVA Link IMS environment. It is assumed to be specified in the applicable APPC/MVS SI profile. The default partner TP ID in the CICS environment is **EKAC**.

A *tp_id* subparameter specified in a PARTNER operand of an EKAPT TYPE=SCP statement overwrites a *tp_ip* subparameter specified in a DEST operand.

DSLTERM
Whether this MERVA system can be terminated by an authorized operator in the partner MERVA system, or whether the MERVA system termination commands must be rejected:

NO A **cancel** and a **terminat** command received from the partner MERVA system is rejected by the local MERVA system. This is the default.

YES An authorized operator in the partner system can terminate the local MERVA system via the **cancel** and **terminat** commands.

LINK
Defines the connection to the partner system. In the MERVA Link **CICS** environment, the *sys_idnt* and *profile* subparameters apply. For more information about these subparameters, refer to the corresponding subparameters of the **DEST** parameter in the CICS environment.

In the MERVA Link **IMS** environment, the *lu_name* and *logmode* subparameters apply. Both subparameters consist of up to 8 characters.

The subparameter *lu_name* is the name of the APPC LU in the remote system. The default value for this parameter is the partner system name *ps-name* specified in the NAME parameter. The optional *logmode* subparameter identifies a VTAM LOGMODE table entry.

LOCLU
The name of one of the local APPC/MVS logical units (LU). It can consist of up to 8 characters. The named LU must be used by MERVA Link for outbound conversations to the partner system specified in the LINK and PARTNER parameters of this macro.

The LOCLU parameter is applicable in the APPC/MVS environment only. Local LU selection is supported by APPC/MVS of MVS/ESA 4.3 (or a later upward compatible release). The LOCLU parameter is ignored in all other environments.

If the LOCLU parameter is omitted, the local LU name field contains blanks. In the APPC/MVS environment, the System Base LU is used for outbound conversations in this case.

In the IMS/ESA APPC/IMS environment the APPC/IMS Base LU that is associated with the APPC/IMS scheduler must be specified as the local LU. The APPC/MVS Base LU cannot be used in the APPC/IMS environment.

LOPER
The names of up to three local MERVA operators who are authorized to operate this partner MERVA system. If LOPER=ALL is specified, all local MERVA master and MERVA Link operators are authorized to operate this partner MERVA system. If this parameter is omitted, no local MERVA operator is authorized to operate this partner MERVA system.

PARTNER
The transaction code of the MERVA System Control Facility program in the partner system. The default value for this parameter is EKAC if the Link parameter was specified. A default value for this parameter is not applicable if the DEST parameter was specified.

POPER
The names of up to three MERVA operators in the partner system who are authorized to operate the local MERVA system. If POPER=ALL is specified, all MERVA master and MERVA Link operators in the partner MERVA system are authorized to operate the local MERVA system. All users of the partner system need an entry with the appropriate authorization in the user file at the local system. If this parameter is omitted, no MERVA operator in the partner MERVA system is authorized to operate the local MERVA system.

Generating an SCP Group Separator

This statement defines the boundary between two groups of SCP entries in this PT. SCP groups may be defined to structure the SCP list displayed by the MERVA System Control Facility.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=SCPGS** [,**GRPSEP=**'*c*']

Programming Notes:

label
An optional assembler label for this statement. It has no meaning for this table entry.

TYPE=SCPGS
An SCP group separator must be generated, that defines the boundary between two SCP groups.

GRPSEP
The character used to form an SCP group separator line in the SCP list displayed by the MERVA System Control Facility.

If this parameter is omitted, a line consisting of dashes is displayed.

Ending the Partner Table

This statement defines the end of the table. It is a mandatory statement and must be followed by an assembler END statement.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=FINAL**

Generating a DSECT of the Partner Table

This statement generates a DSECT of the PT in assembler language. This DSECT describes all parts of a PT, the PT header, an ASP entry, an MTP entry, and the PT trailer. It also provides equate symbols for flags and identifiers.

Name	Operator	Operands
[*label*]	**EKAPT**	**TYPE=DSECT**

EKAUXS: Starting the MFS User Exit

The MERVA Link provides the EKAUXS macro for an MFS user exit.

This start macro EKAUXS establishes the MERVA ESA environment for an MFS user exit used with MERVA Link. It should be the first noncommentary statement in the source deck used to generate a user exit in assembler language.

Name	Operator	Operands
[*label*]	**EKAUS,**	[**CICS=**{<u>**NO**</u>|**YES**}] [**,NUM=**{<u>**7010**</u>|*number*}] [**,PRINT=**{<u>**ON**</u>|**ALL**|**OFF**}]

Programming Notes:

label
An optional assembler label for this statement. If it is specified, an assembled CSECT statement with the specified label is generated. If the label is not specified, the CSECT statement must have been coded earlier in the source deck.

CICS
Whether the code for the CICS environment setup should be generated:

NO The CICS environment setup code is not generated. CICS commands cannot be issued in this user exit. This is the default.

YES The CICS environment setup code is generated. CICS commands can be issued in this user exit. The fields UXDFHEIS, UXDFHEIB, UXEKACPL, and UXDSLTOF must be defined in the MFS Temporary Storage work area of the user exit if this parameter is specified.

NUM
The number of the user exit. This number is mandatory. It must be within the range of 7000 to 7999. The default number is 7010.

PRINT
Whether the code for the environment setup should be printed:

ON The setup code for the MERVA ESA environment, a MERVA Link interface description, and the MERVA Link DSECTS for the EKAXCPL and the Partner Table are to be printed. The MERVA ESA DSECTS are not printed with this specification. This is the default.

ALL In addition to the printout obtained with PRINT=ON, the MERVA ESA DSECTS are printed.

OFF No setup code is printed.

EKAUXS Error Information

The code generated by the macro EKAUXS checks for valid parameters of the DSLMFS user exit call. These parameters are the pointer to the EKAXCPL communication area and the pointer to the Partner Table. If either of these pointers is invalid, the user exit returns to the caller with MFS return code 12 (X'000C'). The MFS reason code is 700 (X'02BC') if the pointer to the EKAXCPL is invalid. The MFS reason code is 701 (X'02BD') if the pointer to the Partner Table is invalid.

A MERVA Link MFS user exit is called by MERVA Link programs, only. Either of these errors, therefore, indicates a MERVA Link programming error.

User-exit user code also reports errors via the MFS reason code. The reason code values should be within the range of 750 to 799.

Note: You must not use reason code values 700 to 749. They are reserved for MERVA Link only.

Chapter 6. FMT/ESA with MERVA Link: Macro

EKASPARM: Generating the Customization Information

The EKASPARM macro has the following functions:

- Generate the customization parameter table
- Map the customization parameters

Generating the Customization Parameter Table

Name	Operator	Operands
[*label*]	**EKASPARM**	[**TYPE=**{**MODULE**|**INLINE**}]
		,ISNCQ=*queue_name*
		,OSNCQ=*queue_name*
		[**,AUTHENT=**([{**YES**|**NO**}][,{**YES**|**NO**}])]
		[**,CHECK=**{**YES**|**NO**}]
		[**,JIDRCVD=**{**61**|*journal_id*}]
		[**,JIDSENT=**{**60**|*journal_id*}]

Programming Notes:

label
A unique statement label according to assembler language conventions. The default is EKASPRM.

TYPE
The usage of the parameter table.

MODULE A CSECT with a name determined by *label* is generated. This type of parameter table is a module of its own and must be link-edited to or loaded by any module which refers to it. The parameter table module named EKASPRM must be link-edited to the FMT/ESA user exit EKAMU044. This is the default.

INLINE The parameter table and some MERVA ESA MFS interface code is generated.

This type of parameter table is embedded into executable code. It may be used to call the FMT/ESA user exit EKAMU044 from another MERVA Link MFS user exit with modified parameter values. You find more information in the *MERVA for ESA Customization Guide*.

ISNCQ
The name of the ISN Control Queue. The queue must be defined in the MERVA ESA function table.

OSNCQ
The name of the OSN Control Queue. The queue must be defined in the MERVA ESA function table.

AUTHENT
Whether the FMT/ESA is to perform the message authentication:

YES SWIFT messages are authenticated.

The first subparameter specifies that the messages to be sent (SWIFT input messages) are authenticated (AUTHENT=(YES)). This is the default.

The second subparameter specifies that the messages to be received (SWIFT output messages) are authenticated (AUTHENT=(,YES)). This is the default.

Note: The SWIFT authentication must have been started either automatically or by the command 'start', which starts the authentication initialization program defined in the MERVA ESA Nucleus program table DSLNPTT.

NO SWIFT messages are not authenticated.

The first subparameter specifies that the messages to be sent (SWIFT input messages) are not authenticated (AUTHENT=(NO)).

The second subparameter specifies that the messages to be received (SWIFT output messages) are not authenticated (AUTHENT=(,NO)).

CHECK
Whether the FMT/ESA is to perform message checking:

YES SWIFT input messages will be checked by the MFS (Message Format Services) of MERVA ESA. Checking is done before SWIFT input messages are being sent to the partner MERVA Link. This is the default.

NO SWIFT input messages will not be checked before being sent.

JIDRCVD
The identifier of the MERVA ESA journal record that is written for a message received from the partner MERVA Link. Received messages are:

- SWIFT output message
- SWIFT input message acknowledgment

Note: This acknowledgment is not always received from the partner MERVA Link. It may have been generated by the FMT/ESA in the local MERVA ESA. It is written to the MERVA ESA journal in order to be compatible with the MERVA ESA SWIFT Link journal entries.

This identifier consists of one or two hexadecimal characters. The default is 61.

JIDSENT
The identifier of the MERVA ESA journal record that is written for a message sent to the partner MERVA Link. Sent messages are:

- SWIFT input message
- SWIFT output message acknowledgment

Note: In the FMT/ESA, this acknowledgment is not sent. It is written to the MERVA ESA journal in order to be compatible with the MERVA ESA SWIFT Link journal entries.

This identifier consists of one or two hexadecimal characters. The default is 60.

Mapping the Customization Parameters

Name	Operator	Operands
[*label*]	**EKASPARM**	**TYPE=DSECT**

Programming Notes:

label
: A unique statement label according to assembler language conventions. The default is EKASPRM.

TYPE
: TYPE=DSECT must be specified for this form of the macro. All other parameters are ignored.

Chapter 7. Telex Link: Macros

ENLPARM: Generating the ENLPRM Module

The ENLPARM macro is used to generate the customizing parameter module ENLPRM for the Telex Link. You must modify ENLPRM according to your environment and requirements, then assemble and link-edit it before starting the Telex Link.

Name	Operator	Operands
[*label*]	**ENLPARM**	[**AUTO={YES**|**NO**}] [**,BUFSIZE=**{**4000**|*nnnn*}] [**,CMDQ=**{**TX2TLC**|*name*}] [**,CTLQ=**{**TX2CTL**|*name*}] [**,JIDR=**{(**40,' RECEIVE '**)|(*nn*,'*cccccccccccccc*')}] [**,JIDS=**{(**41,' SEND '**)|(*nn*,'*cccccccccccccc*')}] [**,JRN1=**{**YES**|**NO**}] [**,LFMID=**{**T**|*x*}] [**,LINES=**{**120**|*nnn*}] [**,LRQ=**{**TXSTPLR**|*name*}] [**,NRMQ=**{**TXNRM**|*name*}] [**,OPNAM=**{**MAS**|*ccc*}] [**,PDEQ=**{**TXSTPPDE**|*name*}] [**,RECEIVE=**{**TXHCFRCV**|*name*}] [**,RTIM=**{**120**|*nnn*}] [**,RTLST=**(*terminal name list*)] [**,RTRAN=**{**ENLR**|*code*}] [**,RTRCV=**{**ENLRTHCF**|*name*}] [**,RTSND=**{**ENLRTHCF**|*name*}] [**,SEND=**{**TXHCFSND**|*name*}] [**,STATION=**{**TELEX**|*name*}] [**,STRAN=**{**ENLS**|*code*}] [**,TESTKEY=***progname*] [**,TKPPCB=**{**3**|*nn*}] [**,URGQ=**{**TXURG**|*name*}] [**,WAITQ=**{**TXWAIT**|*name*}] [**,WERRQ=**{**TXNAK**|*name*}]

Programming Notes:

AUTO
Whether the session with Headoffice Telex on a fault-tolerant system is to be signed on automatically when the Telex Link (program ENLSTP) is started:

YES	The session is started automatically. This is the default.
NO	The session is not started automatically.

BUFSIZE
The size of the buffer used for the communication with Headoffice Telex on a fault-tolerant system, that is, the maximum number of characters that a telex message can have. The default is 4000.

The carriage-return and line-feed characters after each line are included in this number, but not the shift characters (alphabetic to numeric and numeric to alphabetic).

The maximum number is 30640, but you must not specify a size which is greater than Headoffice Telex on a fault-tolerant system can handle.

Note: If you use a buffer size that is greater than 6000 (as contained in the sample ENLPRM), you must also check the values for the NICBUF, JRNBUF, and TOFSIZE parameters in the MERVA ESA customizing parameter module DSLPRM accordingly. The NICBUF parameter value must be at least 2000 characters more than the BUFSIZE parameter value, and you must increase the TOFSIZE parameter by the same value as the NICBUF parameter. The record size defined for the journal data sets must be greater than the BUFSIZE parameter value unless the journal is specified with segmentation in DSLPRM.

CMDQ
The 1- to 8-character name of the queue where the last received command response of workstation based telex functions is stored. The sample function table definitions of the workstation based telex functions contain the queue name TX2TLC (this is the default).

CTLQ
The 1- to 8-character name of the control queue where the control message of workstation based telex functions is stored. The sample function table definitions of the workstation based telex functions contain the queue name TX2CTL (this is the default). For details refer to the functional description of workstation based telex functions sequence numbering in *MERVA Workstation Based Functions*.

Note: The function table definition for the sequence number control queue must specify **KEY1=(EKAASP,8)**.

JIDR
The identification of journal records for messages received from Headoffice Telex on a fault-tolerant system:

- The first subparameter specifies the journal identifier as two hexadecimal digits.
- The second subparameter specifies a descriptive text of up to 15 characters, which must be enclosed in single quotes.

The default is (40,’ RECEIVE ’).

JIDS
The identification of journal records for messages sent to Headoffice Telex on a fault-tolerant system:

- The first subparameter specifies the journal identifier as two hexadecimal digits.
- The second subparameter specifies a descriptive text of up to 15 characters, which must be enclosed in single quotes.

The default is (41,’ SEND ’).

JRN1
Whether telex messages are journaled:

YES The telex messages are journaled immediately before they are sent to

Headoffice Telex on a fault-tolerant system, or immediately after they have been received from Headoffice Telex on a fault-tolerant system. This is the default.

NO The telex messages are not journaled.

LFMID
The 1-character line-format identification to be used first when formatting a telex message for transmission:

T SWIFT messages are formatted using the MCBs supplied by the SWIFT Link. This is the default. To ensure better readability, field tags can be extended by modifying the MCBs as described in the *MERVA for ESA Installation Guide* and *MERVA for ESA Customization Guide*.

S SWIFT messages are formatted using the MCBs supplied by the SWIFT Link without modification. The SWIFT message then looks exactly as if sent to the SWIFT network, particularly if the field tags are the same.

LINES
The maximum number of text lines that an outgoing telex message can have. The maximum is 9999; the default is 120. The same number of lines must be defined in the Message Control Block (MCB) for the message type TELEX, the message type for outgoing telex messages.

Note: In the **signon** command for the session with Headoffice Telex on a fault-tolerant system, the Telex Link specifies 9999 lines for received telex messages. Therefore, the Message Control Block for the message type TRCV (received telex message) specifies a data area count (DACNT) of 9999 for the text of the received telex message.

LRQ
The 1- to 8-character name of the queue that contains the last message received from Headoffice Telex on a fault-tolerant system. The sample function table definitions of the Telex Link contain the queue name TXSTPLR (this is the default). You must not change any parameter of this queue except its name.

NRMQ
The 1- to 8-character name of the queue that contains the telex messages that are ready for transmission to Headoffice Telex on a fault-tolerant system and do not have the telex type U (urgent), that is, they have the telex type N (normal), T (timed), or P (print only). Telex messages contained in this queue are only sent to Headoffice Telex on a fault-tolerant system when the queue for urgent telex messages (defined with the URGQ parameter) is empty. The sample function table definitions of the Telex Link contain the queue name TXNRM (this is the default).

OPNAM
A 3-character identification of the MERVA ESA operators who are authorized to use the restricted Telex Link commands. The user IDs of these operators must start with these 3 characters. Operators whose user IDs start with the 3 characters MAS (the default) are always allowed to use the restricted Telex Link commands.

If you specify OPNAM=///, MERVA ESA does not check the first 3 characters of the user ID. Instead, MERVA ESA checks whether the MERVA ESA operators contain the authorization for the restricted Telex Link commands in their user file records.

Note: If you want the user ID specified in parameter APIUID to get the authorization for the restricted Telex Link commands, this user ID must start with the same 3 characters as specified in OPNAM. That is, the APIUID user ID must start with /// when OPNAM=/// is specified.

An operator is **always** authorized if the user record of this operator indicates it.

PDEQ
The 1- to 8-character name of the queue that contains the last message sent to Headoffice Telex on a fault-tolerant system until the logical acknowledgment is received. The sample function table definitions of the Telex Link contain the queue name TXSTPPDE (this is the default). You must not change any parameter of this queue except its name.

RECEIVE
The 1- to 8-character name of the queue where messages received from Headoffice Telex on a fault-tolerant system are stored by the Telex Link Receive transaction ENLHCF1. The sample function table definitions of the Telex Link contain the queue name TXHCFRCV (this is the default). You must not change any parameter of this queue except its name.

RTIM
A numeric time interval in seconds. The default is 120. If a telex message has been sent to Headoffice Telex on a fault-tolerant system, and Headoffice Telex on a fault-tolerant system does not send the logical acknowledgment within the time specified here, an authorized operator can use the command **txdisp recover** to resend the telex message to Headoffice Telex on a fault-tolerant system.

RTLST
For CICS only, a list of the logical terminal names to which error messages are sent if MERVA ESA is not ready when data is received from Headoffice Telex on a fault-tolerant system. The specified names must be enclosed in parentheses and separated by commas.

RTRAN
The 1- to 4-character transaction code used for receiving data from Headoffice Telex on a fault-tolerant system. You must also specify this transaction code:

- In Headoffice Telex on a fault-tolerant system
- Under CICS, in the transaction definition and terminal definition
- Under IMS, in the transaction definitions for the application ENLHCF1

The default is ENLR.

RTRCV
The 1- to 8-character name of the MERVA ESA routing table used for routing telex messages and invalid data received from Headoffice Telex on a fault-tolerant system. The Telex Link provides a sample with the name ENLRTHCF (this is the default).

RTSND
The 1- to 8-character name of the MERVA ESA routing table used for routing telex messages sent to Headoffice Telex on a fault-tolerant system, for which the transmission acknowledgment was received from Headoffice Telex on a fault-tolerant system. The Telex Link provides a sample with the name ENLRTHCF (this is the default).

SEND
The 1- to 8-character name of the queue where the message is stored for

sending to Headoffice Telex on a fault-tolerant system by the Telex Link Send transaction program ENLHCF1. The sample function table definitions of the Telex Link contain the queue name TXHCFSND (this is the default).

STATION
The 1- to 8-character station name that is used in the transmission headers of all messages sent to or received from Headoffice Telex on a fault-tolerant system. The default is TELEX.

STRAN
The 1- to 4-character transaction code used for sending data to Headoffice Telex on a fault-tolerant system. You must also specify this transaction code:

- In the MERVA ESA function table, in the queue with the name defined with the SEND parameter
- Under CICS, in the transaction definition
- Under IMS, in the transaction definitions for the application ENLHCF1

The default is ENLS.

TESTKEY
The 8-character module name of the test-key processing program to be invoked for the **testkey** command:

- Under CICS, control is given to this program using an EXEC CICS LINK command.
- Under IMS, this program is loaded and control given to it using a BALR assembler instruction.

TKPPCB
For IMS only, the position of the first PCB for databases used by a test-key processing program in the PSB of the MERVA ESA program DSLEUD. The default is 3.

URGQ
The 1- to 8-character name of the queue that contains the telex messages that are ready for transmission to Headoffice Telex on a fault-tolerant system and have the telex type U (urgent). Telex messages contained in this queue are sent to Headoffice Telex on a fault-tolerant system before any telex message contained in the queue for normal telex messages (defined with the NRMQ parameter) is sent. The sample function table definitions of the Telex Link contain the queue name TXURG (this is the default).

WAITQ
The 1- to 8-character name of the queue that contains the telex messages sent to Headoffice Telex on a fault-tolerant system, and for which logical acknowledgment has been received, but not the transmission acknowledgment. The sample function table definitions of the Telex Link contain the queue name TXWAIT (this is the default).

WERRQ
The 1- to 8-character name of the queue that contains the telex messages for which a negative logical or transmission acknowledgment has been received from Headoffice Telex on a fault-tolerant system. The sample function table definitions of the Telex Link contain the queue name TXNAK (this is the default).

ENLPARM: Mapping the Telex Link Customizing Parameters

Code the following ENLPARM macro statement to use the default customizing parameters.

Name	Operator	Operands
[*label*]	**ENLPARM**	**MF=D**

Programming Notes:

MF=D
Maps the Telex Link customizing parameters. If the MF parameter is specified, the value D is assumed and all other parameters are ignored.

ENLTKREQ: Defining the Test-Key Requirements Table

The ENLTKREQ macro generates the test-key requirements table (ENLTKRQT). The test-key requirements table specifies, for specific message types and groups of message types, whether messages of these types must be protected by the addition of a test key.

To generate the test-key requirements table you must enter a sequence of macros.

The first macro must be an ENLTKREQ TYPE=INITIAL macro:

Name	Operator	Operands
	ENLTKREQ	**TYPE=INITIAL**

You can specify the test-key requirements for a particular message type or for a group of message types using the following macro:

Name	Operator	Operands
	ENLTKREQ	[**TYPE=ENTRY**] **,MT=***nnnn* **,TK=**{**YES**|**NO**}

Programming Notes:

TYPE
ENTRY defines one entry in the test-key requirements table. The defalut is ENTRY.

MT
Defines a message type or a group of message types of 4 characters. Any alphanumeric character can be used. Asterisks can be used in any of the four positions and are considered equal to the same position in a specific message type.

To specify the test-key requirement for all message types not specifically listed in the table, you must code the following value:

```
MT=****
```

Enter this definition as the last ENLTKREQ TYPE=ENTRY macro in the table.

MERVA ESA allows for 8-character message types, but only the first 4 characters are tested by the Telex Link.

TK
Whether a test key is required for this message type or group of message types.

The last macro in the sequence must be an ENLTKREQ TYPE=FINAL macro:

Name	Operator	Operands
	ENLTKREQ	**TYPE=FINAL**

This macro must be followed by the END assembler statement.

ENLTKRQT: Programming Rules

When creating a telex message, the test-key requirements table is searched sequentially for a matching specification, that is, where the particular message type has the same character in all four positions as the message type specification in the table.

An asterisk in any position in the table will match any character found in the same position of the message type.

The TK value specified in the table determines the test-key requirement for all message types that satisfy that specification. Therefore, if specific message types and groups of message types overlap, the more specific one must be coded first in the sequence.

Example: Enter specific message types first, followed by more general definitions such as:

```
ENLTKRQT ENLTKREQ TYPE=INITIAL
         ENLTKREQ MT=S0**,TK=NO
         ENLTKREQ MT=S100,TK=YES
         ENLTKREQ MT=S2**,TK=YES
         ENLTKREQ MT=S3**,TK=NO
         ENLTKREQ MT=****,TK=NO
         ENLTKREQ TYPE=FINAL
         END
```

Refer to the *MERVA for ESA Customization Guide* for more information.

Appendix A. Macro List

The macros identified in this appendix are provided as programming interfaces for customers by MERVA ESA.

Attention:

Do not use as programming interfaces any MERVA ESA macros other than those identified in this appendix.

The following macros are provided as Product-sensitive Programming Interfaces:

DSLCOM	DSLCWA	DSLDSFDT
DSLDSMCB	DSLECOFN	DSLEISPA
DSLEPT	DSLFLT	DSLFLV
DSLFNT	DSLGRP	DSLJRN
DSLKPROC	DSLLCOND	DSLLDEV
DSLLDFLD	DSLLEXIT	DSLLFDT
DSLLFLD	DSLLGEN	DSLLGRP
DSLLMCB	DSLLMFLD	DSLLNFLD
DSLLSUBF	DSLLUEND	DSLLUNIT
DSLMFS	DSLMLIT	DSLMPFK
DSLMPT	DSLMSG	DSLMTT
DSLNCM	DSLNIC	DSLNMO
DSLNPA	DSLNPT	DSLNSV
DSLNTR	DSLOMS	DSLPARM
DSLQMG	DSLROUTE	DSLRTC
DSLSRV	DSLTFD	DSLTIM
DSLTRA	DSLTSV	DSLTXT
DSLUSR	DSLWTO	DSLZCW
DSLZDEF	DSLZSC	DSLZSS
DWSAUT	DWSCI	DWSCUR
DWSLT	DWSPARM	DWSVLINE
EKAASFB	EKAASFC	EKAASFE
EKAPT	EKASPARM	EKAUXS
ENLPARM	ENLTKREQ	

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

Trademarks

- 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.

ASF. (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 Version 4.1 System Programming Reference*, SC33-1171
- *High Level Assembler Language Reference*, SC26-4940
- *MQSeries for MVS/ESA System Management Guide*, SC33-0806
- *MVS/ESA SP V5 Writing TPs for APPC/MVS*, GC28-1471
- *MVS/ESA SP V5 Writing Servers for APPC/MVS*, GC28-1472
- *MVS/ESA SP V5 Initialization and Tuning Reference*, SC28-1452.

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

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.

Send the fax to:

```
To: MERVA Development, Dept. 5640              Fax Number: +49-7031-16-4881
    Attention: Gerhard Stubbe                  Internet address:
                                               mervareq@de.ibm.com
    IBM Deutschland Entwicklung GmbH
    Schoenaicher Str. 220
    D-71032 Boeblingen
    Germany
```

MERVA Requirement Request

```
To: MERVA Development, Dept. 5640              Fax Number: +49-7031-16-4881
    Attention: Gerhard Strubbe                 Internet address:
                                               mervareq@de.ibm.com
    IBM Deutschland Entwicklung GmbH
    Schoenaicher Str. 220
    D-71032 Boeblingen      Germany

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
Macro Reference
Version 4 Release 1

Publication No. SH12-6377-01

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
SH12-6377-01

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

SH12-6377-01

Cut or Fold
Along Line

Program Number: 5648-B29

SH12-6377-01

Spine information:

IBM MERVA for ESA Macro Reference Version 4 Release 1