English

LMS V3.4B

Subroutine Interface

Edition June 2013

Comments… Suggestions… Corrections…

The User Documentation Department would like to know your opinion on this manual. Your feedback helps us to optimize our documentation to suit your individual needs.

Feel free to send us your comments by e-mail to:
manuals@ts.fujitsu.com

Certified documentation according to DIN EN ISO 9001:2008

To ensure a consistently high quality standard and user-friendliness, this documentation was created to meet the regulations of a quality management system which complies with the requirements of the standard DIN EN ISO 9001:2008.

cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de

Copyright and Trademarks

Contents

1 Preface

1.1 Brief product description

The Library Maintenance System (LMS) can be called up as a subroutine by a user program. The subroutine offers the user easy-to-use options for processing LMS libraries and their contents, direct from the main program, with LMS being loaded dynamically. This subroutine interface can also be used in the XS (extended system) area.

1.2 Target group

This manual is aimed at LMS users and programmers who want to utilize the various LMS options for particular programs.

Anyone wishing to call LMS as a subroutine should be familiar with Assembler, C or COBOL and with the most important BS2000 commands, in addition to having a good basic knowledge of LMS itself.

1.3 Summary of contents

This manual deals exclusively with the description of the LMS subroutine interface.

Format of the manual

Each chapter contains the following information:

- **LMS as a subroutine**

  Overview of functions and applications.

- **Call preparations**

  Description of the call preparations, return codes and format of the parameter structures.

- **Subroutine functions**

  Description of all functions in alphabetical order with call parameters and return codes.

- **COBOL interface**

  Description of the COBOL parameter structures and example of a COBOL main program.

- **C interface**

  Description of the C parameter structures and example of a C main program.

- **Assembler interface**

  Description of the Assembler parameter structures and example of an Assembler main program.

For a detailed description of LMS, including the various LMS statements and messages, see the User Guide “LMS (BS2000) SDF Format” [1].

README file

The functional changes to the current product version and revisions to this manual are described in the product-specific Readme file.

Readme files are available to you online in addition to the product manuals under the various products at *http://manuals.ts.fujitsu.com*. You will also find the Readme files on the Softbook DVD.

Information under BS2000/OSD

When a Readme file exists for a product version, you will find the following file on the BS2000 system:

```
SYSRME.<product>.<version>.<lang>
```

This file contains brief information on the Readme file in English or German (<lang>=E/D). You can view this information on screen using the `/SHOW-FILE` command or an editor. The `/SHOW-INSTALLATION-PATH INSTALLATION-UNIT=<product>` command shows the user ID under which the product's files are stored.

Additional product information

Current information, version and hardware dependencies, and instructions for installing and using a product version are contained in the associated Release Notice. These Release Notices are available online at *http://manuals.ts.fujitsu.com*.

Terminological note

In the LMS manuals the terms "element" and "member" are used synonymously.

1.4 Notational conventions

Throughout the text, reference literature is quoted using abbreviated titles accompanied by a number in square brackets. The full title of each publication referred to may be found under the appropriate number in the "Related publications" section.

2 LMS as a subroutine

The subroutine interface offers the user a convenient facility for processing program libraries and their members. The LMS functions discussed below are called directly from a user program (COBOL, C, Assembler). Control remains in the user program.

Opening and terminating subroutine access

Subroutine access is opened via the INIT function and terminated via END. Every time INIT is called, a new subroutine access identification is created so that parallel subroutine accesses can be identified.

Functions for the subroutine interface

The following functions can be called via the subroutine interface:

– ADD incorporates files as members in a library.
– CLOSLIB closes a library.
– COPY permits members to be copied.
– COPYLIB copies libraries logically.
– COPYSTR copies delta members with their structure being retained.
– DEL is used to delete members.
– GSYSELEM reads the contents of a system variable, interprets it as a member specification (library, member name, version and type) and converts it to a form that the subroutine interface can handle.
– LOCK prevents members from being modified.
– LST selects members for output to SYSLST.
– MODEA is used to set member attributes.
– MODEP is used to set member protection.
– MODLA is used to set library attributes.
– MODTA is used to set type attributes.
– PROVIDE allows a member to be “borrowed”.
– REN serves to rename non-delta members.
– REORGLIB reduces the amount of disk storage space required for a library.
– RETURN returns a “borrowed” member.
– SEL selects members for output to a file.
– SHOWLA selects library attributes for output.
– SHOWTA permits the type attributes to be displayed.

- UNLOCK readmits modification of members.

Functions for reading or writing members

When a member is to be read or written, it must be opened using one of the three OPEN functions:

- OPENGET opens members for reading
- OPENPUT opens members for writing
- OPENUPD opens members for reading and writing

Simultaneous OPENGET, OPENPUT and OPENUPD calls for the same member are not possible.

All OPEN functions define an access path identification which permits several members to be opened concurrently. This identification must always be specified in subsequent GET and PUT calls.

- GET enables a record to be read
- PUT enables a record to be written
- CLOSE must be used to close the member explicitly after it has been processed

Functions for searching for members

Three TOC functions provide information on member entries:

- TOCPRIM searches for a member in the primary directory
- TOCSEC searches for a member in the secondary directory
- TOC searches for further members

TOCPRIM and TOCSEC are used to define the search criteria for a member and to output the member entries of the first member satisfying these criteria. At the same time, a TOC identification must be specified for both functions. If the search criteria are to be used to look for further member entries, the TOC function must be called. The TOC function continues TOCPRIM or TOCSEC. The TOC identification must be specified in the TOC call in order to determine which of the two functions is to be continued.

Functions for member protection

Member protection can be set by means of the following functions:

– MODEP sets member protection for a specific member.
– MODTA sets the initial member protection for all new members of a specific member type.
– MODLA sets the initial member protection for all new members of a library.

Member protection can be displayed by means of the following functions:

– The TOC functions display the member protection for a specific member.
– SHOWTA displays the type attributes of a specific member type.
– SHOWLA displays the library attributes of a library.

Functions for version automation

The following two functions apply to version automation:

– MODTA to set the convention.
– SHOWTA to display the valid convention.

STD-SEQUENCE, MULTI-SEQUENCE and STD-TREE are available as conventions.

Functions for the support of extended character sets in LMS

The following functions support extended character sets in LMS (see the “XHCS” manual [2]):

– MODEA to set and modify a character set name
– the TOC functions to display the character set name assigned to a member.

Functions for managing permissible storage modes

Versions of data can be stored either in full or in delta (incremental) form. The following functions are available for managing the permissible storage mode for a given library or type:

– MODLA to set the permissible storage mode for the members of a library.
– MODTA to set the permissible storage mode for the members of a type.
– SHOWLA to check the permissible storage mode for the members of a library.
– SHOWTA to check the permissible storage mode for the members of a type.

The storage mode for a member is selected when the member is created.

Functions for supporting a borrowing mechanism

– MODLA to set the write control for a library and to define the initial borrowing privilege for all new members of that library.
– MODTA to set the write control for a type and to define the initial borrowing privilege for all new members of that type.
– MODEP to set the borrowing privilege for a member.
– MODEA to set the state for a member (FREE or INHOLD).
– PROVIDE to reserve a member of a source library and then copy it to an output library.
– RETURN to return a member of a source library to an output library, provided that the base specified for the target version has been reserved by the user in the output library. RETURN also deletes the member from the source library and cancels the reservations in the output library.
– SHOWLA to check the write control and initial borrowing privilege of a library.
– SHOWTA to check the write control and initial borrowing privilege of a type.
– The TOC functions to check the state (FREE or INHOLD) and borrowing privilege of a member.

Functions for supporting make functionality

– MODEA for updating the modification date.

3 Call preparations

In order for the main program to be linked with LMS, the caller must permanently link in module LMSUP1 from the SYSLNK.LMS.034 library.
The installation location of this library is freely selectable via IMON.

The installation location of SYSLNK.LMS.034 can be determined using the builtin function INSTALLATION-PATH:

```
/SET-VARIABLE LIBRARY-NAME =INSTALLATION-PATH-
                            (LOGICAL-ID = ´SYSLNK´-
                            ,INSTALLATION-UNIT = ´LMS´-
                            ,VERSION = ´3.4´-
                            ,DEFAULT-PATH-NAME = ´$.SYSLNK.LMS.034´)
```

Furthermore, as of BS2000/OSD V2.0, if more than one LMS version is installed, it is possible via SELECT-PRODUCT-VERSION to specify a specific LMS version to be dynamically loaded. Otherwise, the highest LMS version is always loaded.

Main programs wanting to call LMS as a subroutine may execute either in 24-bit or 31-bit addressing mode.

3.1 Call preparations and return information

Supplying the call parameters

Before a function is called, the caller must supply certain fields of the parameter structures, depending on the function to be called. In the description of each function, the fields to be supplied are listed in the table of call parameters. The sequence of the fields in this table must be adhered to.

Return parameters

After a function has been called, return parameters (return codes, MSG codes) are written to certain fields of the parameter structures, depending on the function called. The relevant fields are listed in the table of return parameters in the descriptions of the individual functions.

Return codes

The return code comprises one character and states whether a function has been executed successfully.

Detailed information can be obtained from the DMS, LMS and PLAM message codes. These message numbers are hex codes and must be interpreted as such. The DMS codes are displayed in decimal code already and can be evaluated directly, whereas the LMS and PLAM codes still have to be converted from hexadecimal to decimal for evaluation by /HELP.

The following return codes are passed at the subroutine interface:

Return code	Meaning
X'00'	The function has been successfully executed.
X'04'	The record buffer defined in GET is too small (maximum record length 32 Kbytes or 256 Kbytes for records of format B).
X'08'	End of member/TOC reached.
X'0C'	Function aborted due to a user error.
X'14'	Call parameters errored or incomplete.
X'18'	Illegal call sequence.
X'1C'	Internal LMS error.

3.2 Function and format of the parameter structures

The parameter structures are the central facility for handling the entire data transfer from the main program to LMS and vice versa. The parameter structures must be defined in the main program. The following features are available for this purpose:

- for COBOL, appropriate COPY members (see page 143ff)
- for C, an appropriate INCLUDE member (see page 175ff)
- for Assembler, appropriate macros (see page 207ff).

The same COPY members, INCLUDE members or macros (format L) can be used to initialize the parameter structures, i.e. to define suitable presettings. The parameter structures of the subroutine interface are:

- the control block CB
- the descriptors ED, FD, LD, RD and TD
- the TOC parameters EI, EM and TID
- the record buffer ER
- the library attributes LA, LI and PA
- the type attributes TA and TI
- the member attributes EA.

Fields of the parameter structures

Numeric fields:	2 or 4 bytes
Indicators:	1 byte. Symbolic values exist for processing these fields.
String fields:	All strings must be entered left-justified in the appropriate fields. The strings that are entered are evaluated up to the first blank, but not beyond field length.

CB (Control Block)

The CB identifies the interface version (SCBVERSION), the appropriate access (ACC) and the desired function (FUNCTION). In addition a subcode (SUBCODE) can be specified. For all functions, a return code (RETURNCODE) and error codes for LMS, DMS and PLAM (LMS-MSG, DMS-MSG, PLAM-MSG) will be returned. Following the INIT call, the LMSVERSION field contains the version number of the LMS installation. The processing operands are set to the default values in the INIT call.

The CB control block is required for all functions. It must always be the first parameter in the parameter list.

If the CB is translated with a new version, all other parts of the program which use LMSUP structures must also be translated.

Field	**Meaning**	**Length in bytes**	**Contents after initialization [after INIT]**
SCBVERSION	Interface version	2	X'F0F4'
FUNCTION	Function code	1	1 x X'01'
SUBCODE	Subcode	1	1 x X'40'
ACC	Subroutine access identification	4	4 x X'FF'
RETURNCODE	Return code class	1	1 x X'00'
	reserved	1	1 x X'00'
LMS-MSG	LMS message code	2	2 x X'00'
DMS-MSG	DMS message code	2	2 x X'00'
PLAM-MSG	PLAM message code	2	2 x X'00'
LMSVERSION	LMS version	12	12 x X'40'
	reserved	4	4 x X'00'
****************	LMS parameters ********************		
DESTROY	Physical overwriting Y: Overwrite data with X’00’ N: Do not overwrite data	1	1 x X'40' [N]
FCB	File attributes S: As stored, or default value I: ISAM Q: SAM C: As described in catalog	1	1 x X'40' [S]

Field	**Meaning**	**Length in bytes**	**Contents after initialization [after INIT]**
RKEY	ISAM key and file attributes Y: Store ISAM key and file attributes N: Do not store ISAM key and file attributes	1	1 x X'40' [N]
OVERWRITE	Logical overwriting Y: Target member/file is overwritten (YES) N: Target member/file is not overwritten (NO) O: Target member/file must exist (ONLY) E: Target member/file extended (EXT) A: All target versions for the specified name are overwritten (NAME) (only applies to COPY, error in all other cases)	1	1 x X'40' [N]
COLUMN-P	Number of columns per line: 132/80	2	H'0' [132]
LINE-P	Number of lines per page LINE-P= 60; 21≤LINE-P≤255 0: title not output	2	H'0' [60]
PROT-IND	PROTECTION INDICATOR S: Target members/files have standard protection (STD) M: Target members/files have same protection as source (SAME)	1	1 x X'40' [S]
ATTR-IND	ATTRIBUTES INDICATOR (COPY-LIB) S: The target library is created with standard protection (STD) M: The target library takes the same attributes as the library file (source) (SAME)	1	1 x X'40' [N]
INFO	Member subarea to be processed S: Process standard area (STD) X'01': Process text proper (record type 1) (TXT) X'02': Process comment/documentation (record type 2) (COM)	1	1 x X'40' [S]

Field	**Meaning**	**Length in bytes**	**Contents after initialization [after INIT]**
LD-RETURN	The NAME field in LD is used as the return parameter for the full DMS file name of the library: N: name is only written if library is specified via link name Y: full library name with catalog ID and user ID after ACS conversion, or hit library in the case of library lists	1	1 x X‘40‘ [N]
	reserved	4	4 x X'40'

EA (Element Attributes)

EA is used to set the member attributes and update the modification date.

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
USER-DATE	User-specified date	14	14 x X'40'
USER-TIME	User-specified time of day	8	8 x X'40'
CCS-NAME	Coded Character Set name	8	8 x X'40'
HOLD-STATE	Hold flag: state assigned to the member ' ': UNCHANGE '-': FREE 'H': INHOLD	1	1 x X'40'
	reserved	8	8 x X'40'
MOD-DATE-IND	MODIFICATION DATE INDICATOR controls modification date updating 'O': OLD 'S': NEW (SYSTEM DATE)	1	1 x X'40'
	reserved	56	56 x X'40'

The descriptor EA is required for the MODEA function.

Notes for input:

– USER-DATE
  The first 10 characters of the date must have the format `YYYY-MM-DD`. However, the date is only checked syntactically for the format `zzzz-zz-zz`. The remaining 4 characters are not checked, but should have the format `dddB`, where ddd stands for the current day of year and B for blank.

– USER-TIME
  The time of day is not syntactically checked. It should have the format `HH:MM:SS`.

– CCS-NAME
  The specified values are not checked for system permissibility. If a member is not to be assigned a character set, the keyword '*NONE_' with a trailing blank must be entered. If only blanks are specified, the existing CCS name of the relevant member is retained.

- HOLD-STATE
  State assigned to the member. Possible values:

FREE	Only the HOLDER and the owner of the library are allowed to change the state to FREE.
INHOLD	The new state of the member is INHOLD. The user ID of the person changing the state from FREE to INHOLD is then recorded as HOLDER. If the state was already INHOLD, the subroutine call is rejected. Only a user with borrowing privilege for the member may become HOLDER.

- MOD-DATE-IND
  Controls whether the modification date (EI: MODIFI-TIME and MODIFI-DATE) is updated or left as is. Possible values:

NEW	The current time of day and the current date are assigned to the member as the time of last modification.
OLD	The time of last modification is left as it is.

ED (Element Description)

The ED descriptor identifies a member. There are restrictions for the following functions:

– OPENGET

TYP Member types C and L are allowed only if the EXTRA subcode is set in the OPENGET call.

– OPENPUT, OPENUPD

TYP Member types C and L are not allowed, as members of these types contain records which cannot be processed.

– COPY, COPYSTR, DEL, OPENGET, PROVIDE, REN, RETURN, SEL

The member designation of the input member is not checked for conformance with LMS conventions (exception: member type for COPY, PROVIDE, REN, RETURN).

– ADD, COPY, COPYSTR, LOCK, OPENPUT, OPENUPD, PROVIDE, REN, RETURN, UNLOCK

The following rules apply for the member designation of the output member:

TYP The member type must be a valid standard LMS type (S, M, P, D, J, X, R, C, L, H, F, U) or derived from one of the above (see "LMS (BS2000) SDF Format" [1]). The standard types (R, C, L, H, F, U) and types derived from them are not permitted for delta members.

NAME
The name must meet the standards described in [1].

VERSION
The version must meet the standards described in [1]. @ is not a valid version designation. If the user wishes to generate the highest possible version, X’FF’ followed by at least one blank must be entered explicitly. The string Vn. (n=0-9) is not converted to V0n.

– ADD, COPY, OPENPUT, OPENUPD, PROVIDE, REN, RETURN

USER-DATE
The first 10 characters of the date must always have the format YYYY-MM-DD (although only a syntax check for zzzz-zz-zz is performed). The remaining 4 characters are not checked; they should have the format dddB, where ddd stands for the current day of year and B for blank.

USER-TIME
The time of day is not checked for syntax errors and should be specified in the form HH:MM:SS.

– ADD, COPY, OPENPUT, PROVIDE

STORE-FORM
The storage mode value must either STD, DELTA or FULL.

– ADD, COPY, OPENPUT, PROVIDE, REN, RETURN

The following rules apply to the member description of the output member:

Due to version automation, the ED version field for the output member contains the following, depending on the subcode:

Subcode	Contents
INCP	Prefix [1]
HIGP	Prefix [1]
INCB	Base

[1] The prefix may also consist of the empty version (blanks) and has a maximum length of 23 characters.

If the subcode is not a blank, the base ED for delta members (ED2 or ED3 for COPY) is ignored.

If the target ED has a member type for which the convention STD-TREE is valid, abbreviated versions (e.g. 2.3) are converted into the internal format (002.003).

When version automation is in use, the named fields are likewise used to write the written target version back. Specified prefix and base entries are overwritten during this process.

Field	Meaning	Length in bytes	Contents after initialization
TYP	Member type	8	8 x X'40'
NAME	Member name	64	64 x X'40'
VERSION	Member version	24	24 x X'40'
STORE-FORM	Storage mode of member ´S´: STD: (FULL or DELTA) ´D´: DELTA: delta member ´V´: FULL: non-delta member	1	C’V’
USER-DATE	Date specified by user	14	14 x X'40'
USER-TIME	Time of day specified by user	8	8 x X'40'

The descriptor ED is required in the following functions:

ADD, COPY, COPYSTR, DEL, GSYSELEM, LOCK, LST, MODEA, MODEP, OPENGET, OPENPUT, OPENUPD, PROVIDE, REN, RETURN, SEL, UNLOCK.

With the ADD, COPY, DEL, LOCK, LST, MODEA, MODEP, OPENGET, OPENUPD, OPENPUT, PROVIDE, REN, RETURN, SEL and UNLOCK functions, the EDVERS field in the source and target member specification in ED may contain the string *HIGH followed by a blank. This allows the user to specify the highest version for the given member name without first having to use the TOC function to look it up. LMS automatically looks up the highest version and returns it in the EDVERS field, overwriting the string *HIGH. The user has to set up a loop to keep passing the value *HIGH to the EDVERS field.

EI (Element Information)

The parameter EI provides member information for the TOC functions. It is a return parameter. Without extensions, EI is identical to the format of ED. The library contents can be listed using TOC and the resulting member information can be immediately used for processing the member. When TOCPRIM is called, extensions 1 and 3 are supplied; when TOCSEC is called, extensions 2 and 3 are supplied (if subcode=LONG is set). The USER-DATE and USER-TIME fields are only given values in the case of subcode=LONG. CSECT names which exceed 32 characters are truncated to this length for output in the SEC-NAME field.

Fields with no current output relevance contain their initial values.

Extension 3 contains the PA fields (except for the passwords), the CCS name, the member state (FREE/INHOLD), the holder’s user ID and the ACCESS-DATE, ACCESS-TIME and ELEMENT-SIZE fields. The possible entries for the individual display fields of the protection attributes are described under PA (see page 41).
Extension 3 permits protection attributes to be displayed. Passwords are not displayed. However, the user is informed if a password has been allocated. In addition, the stored character set name can be output for each member. The output ’*NONE’ means that “no code” is output.
If the member state is INHOLD, the user ID of the holder is output in the HOLDER field (the holder being the one who changed the member state from FREE to INHOLD); otherwise a blank is output. The ELEMENT-SIZE field displays the number of PAM pages (2-K units) the member requires in the memory.

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
TYP	Member type	8	8 x X'40'
NAME	Member name	64	64 x X'40'
VERSION	Member version	24	24 x X'40'
STORE-FORM	Storage mode of member ´D´: DELTA: delta member ´V´: FULL: non-delta member	1	1 x X'40'
USER-DATE	Date specified by user	14	14 x X'40'
USER-TIME	Time of day specified by user	8	8 x X'40'
	*********** Extension 1 ***		
CREATION-DATE	Date of member generation	14	14 x X'40'
CREATION-TIME	Time of member generation	8	8 x X'40'
MODIFI-DATE	Date of last update	14	14 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
MODIFI-TIME	Time of last update	8	8 x X'40'
	*********** Extension 2 ***		
SEC-NAME	Reference name	32	32 x X'40'
SEC-ATTRIBUTE	Reference attribute	8	8 x X'40'
	reserved	5	5 x X'40'
	*********** Extension 3 ***		
P-TIND-READ	Protection type indicator for read	1	1 x X'40'
P-READ-OWN	Read indicator for owner	1	1 x X'40'
P-READ-GRP	Read indicator for group	1	1 x X'40'
P-READ-OTH	Read indicator for others	1	1 x X'40'
P-READ-PIND	Read password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-WRIT	Protection type indicator for write	1	1 x X'40'
P-WRIT-OWN	Write indicator for owner	1	1 x X'40'
P-WRIT-OTH	Write indicator for others	1	1 x X'40'
P-WRIT-PIND	Write password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-EXEC	Protection type indicator for execute	1	1 x X'40'
P-EXEC-OWN	Exec indicator for owner	1	1 x X'40'
P-EXEC-GRP	Exec indicator for group	1	1 x X'40'
P-EXEC-OTH	Exec indicator for others	1	1 x X'40'
P-EXEC-PIND	Exec password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-GUARD-READ	Read guard indicator	18	18 x X'40'
P-GUARD-WRIT	Write guard indicator	18	18 x X'40'
P-GUARD-EXEC	Exec guard indicator	18	18 x X'40'
CCS-NAME	Coded Character Set name	8	8 x X'40'
P-TIND-HOLD	Borrowing privilege indicator	1	1 x X'40'
P-HOLD-OWN	Borrowing privilege indicator for owner	1	1 x X'40'
P-HOLD-GRP	Borrowing privilege indicator for group	1	1 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
P-HOLD-OTH	Borrowing privilege indicator for others	1	1 x X'40'
P-HOLD-PIND	Borrowing password indicator	1	1 x X'40'
	reserved	4	4 x X'40'
P-GUARD-HOLD	Borrowing guard indicator	18	18 x X'40'
HOLD-STATE	Hold flag: state assigned to the member '-': FREE 'H': INHOLD	1	1 x X'40'
HOLDER	User ID of the holder	8	8 x X'40'
ACCESS-DATE	Date of last access to member	14	14 x X'40'
ACCESS-TIME	Time of last access to member	8	8 x X'40'
	reserved	1	1 x X'40'
ELEMENT-SIZE	Member size indicator	4	4 x X‘00‘
	reserved	40	40 x X‘40‘

The TOC parameter EI is required in the following functions:

TOC, TOCPRIM, TOCSEC.

EM (Element Mask)

The parameter EM controls member selection for the TOC functions. EM is a call parameter.

Extension 3 contains the PA fields (except for the passwords). The possible entries for the individual fields of the protection attributes are described under PA. Extension 3 also permits selection through the CCS names, the member status (FREE / INHOLD), the user ID of the holder, the ACCESS-DATE and ACCESS-TIME fields, and the E-SIZE-MIN and E-SIZE-MAX fields. Selection is therefore possible through specified protection attributes, but not through specified passwords. If a blank is entered or if X‘00000000‘ is entered for E-SIZE-MIN and X‘FFFFFFFF‘ for E-SIZE-MAX, any value can be used for selection.
In the case of passwords, selections can be made on the basis of the indicators (existence of a password).

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
TYP	Member type	20	20 x X'40'
NAME	Member name	132	132 x X'40'
VERSION	Member version	52	52 x X'40'
STORE-FORM	Storage mode of member	6	6 x X'40'
USER-DATE	Date specified by user	32	32 x X'40'
USER-TIME	Time of day specified by user	20	20 x X'40'
	*********** Extension 1 ***		
CREATION-DATE	Date of member generation	32	32 x X'40'
CREATION-TIME	Time of member generation	20	20 x X'40'
MODIFI-DATE	Date of last update	32	32 x X'40'
MODIFI-TIME	Time of last update	20	20 x X'40'
	*********** Extension 2 ***		
SEC-NAME	Reference name	68	68 x X'40'
SEC-ATTRIBUTE	Reference attribute	20	20 x X'40'
	reserved	14	14 x X'00'
	*********** Extension 3 ***		
P-TIND-READ	Protection type mask for read	1	1 x X'40'
P-READ-OWN	Read mask for owner	1	1 x X'40'
P-READ-GRP	Read mask for group	1	1 x X'40'

Field	Meaning	Length in bytes	Contents after initialization
P-READ-OTH	Read mask for others	1	1 x X'40'
P-READ-PIND	Read password mask	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-WRIT	Protection type mask for write	1	1 x X'40'
P-WRIT-OWN	Write mask for owner	1	1 x X'40'
P-WRIT-GRP	Write mask for group	1	1 x X'40'
P-WRIT-OTH	Write mask for others	1	1 x X'40'
P-WRIT-PIND	Write password mask	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-EXEC	Protection type mask for exec	1	1 x X'40'
P-EXEC-OWN	Exec mask for owner	1	1 x X'40'
P-EXEC-GRP	Exec mask for group	1	1 x X'40'
P-EXEC-OTH	Exec mask for others	1	1 x X'40'
P-EXEC-PIND	Exec password mask	1	1 x X'40'
	reserved	4	4 x X'00'
P-GUARD-READ	Read guard mask	40	40 x X'40'
P-GUARD-WRIT	Write guard mask	40	40 x X'40'
P-GUARD-EXEC	Exec guard mask	40	40 x X'40'
CCS-NAME	Coded Character Set name	20	20 x X'40'
P-TIND-HOLD	Borrowing privilege mask	1	1 x X'40'
P-HOLD-OWN	Borrowing privilege mask for owner	1	1 x X'40'
P-HOLD-GRP	Borrowing privilege mask for group	1	1 x X'40'
P-HOLD-OTH	Borrowing privilege mask for others	1	1 x X'40'
P-HOLD-PIND	Borrowing password indicator mask	1	1 x X'40'
	reserved	4	4 x X'40'
P-GUARD-HOLD	Borrowing guard mask	18	18 x X'40'
HOLD-STATE	Hold flag: state assigned to the member ' ': ANY '-': FREE 'H': INHOLD	1	1 x X'40'
HOLDER	User ID of the holder	8	8 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
ACCESS-DATE	Date of last access to member	14	14 x X'40'
ACCESS-TIME	Time of last access to member	8	8 x X'40'
	reserved	3	3 x X'40'
E-SIZE-MIN	Lower limit for member size selection (PAM pages, 2-K unit)	4	X‘00000000‘
E-SIZE-MAX	Upper limit for member size selection (PAM pages, 2-K unit)	4	X‘FFFFFFFF‘
	reserved	64	64 x X‘40‘

The TOC parameter EM is required in the following functions:

TOCPRIM, TOCSEC.

Permissible wildcards for all string fields:

Wildcard	**Meaning**
*	Replaces any string (also a null string).
/	Replaces one arbitrary character.
<s1:s2>	Replaces a string with the following criteria: – Minimum length = min(L’s1,L’s2) – Maximum length = max(L’s1,L’s2) – Located between "s1" and "s2" (inclusive) in alphabetical order – "s1" and/or "s2" may also be empty.
<s1:s2,...>	List form of the type "s1:s2" – The above rules apply for each range specified. – The wildcard list replaces all strings to which one of the range entries applies (ORing). – The length attributes are paired, i.e. they apply to one range entry at a time The angle brackets (’<’ and ’>’) designating a set must always be present in pairs. The characters ’*’, ’/’, ’<’, and ’>’ must not occur within sets.
-s	Replaces all strings not matching the wildcard. The minus sign may only appear at the beginning of the wildcard string.

Notes for input:

– When entering a range for USER-, CREATION- and MODIFICATION-TIME, hours must be entered as follows: `<HH:HH>*`

  When entering USER-, CREATION- and MODIFICATION-DATE, dates must be entered in the following form: `YYYY-MM-DD*`
  A date range must be entered in the following form: `<YYYY-MM-DD:YYYY-MM-DD>*`

  Entries should always end with the asterisk (*), as this stands for the Julian date. LMS searches for both the specified date and the Julian date which is automatically created at member generation.

  If the asterisk is omitted, LMS inserts 4 blanks instead of the Julian date. As a result, errors occur when the member mask EM is analyzed.

– A search for members to which “no code” is assigned can be conducted by entering the keyword ’*NONE’ with a trailing blank in the CCS-NAME field. If blanks are entered, members with any CCS name will be selected.

– CSECT names which exceed 32 characters are truncated to this length before the wildcard comparison, so there is no point in specifying longer wildcard strings in the SEC name field.

ER (Element Record)

The ER record buffer transfers member records between caller and LMS in either direction. Members consist of variable-length records, i.e. a 4-byte record header is always prefixed to the data area.

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
	Record header	2	Record length max 32 K (incl. header)
		1	reserved
		1	Record type
	Record area (see GET)	variable	Current record

When a record is read, the record header in ER is supplied by LMS. The record length value is derived from the minimum record length of the record read and the buffer length specified by the user in RD. Record types 1-159, 163 and 164 can be read (record types 1-159 may be freely selected by the user).

When a record is written (PUT), the first 4 bytes in ER are reserved for the record header. In PUT, however, the record header in ER is not analyzed.

The record buffer ER is required in the GET and PUT functions.

Exception

Format B records (record type 160)
Format B records do not have a record header. The record length is a multiple of 2K, to a maximum of 256K. The size of the record buffer (ER) must be chosen accordingly (also refer to OPENGET and GET).

FD (File Description)

The descriptor FD identifies a file. If a link name (LINK) exists, the associated file is processed. Any additional file name (NAME) specified for FD is ignored. LMS enters in the NAME field the file name associated with LINK. If no link name exists, the file identified by NAME is processed.

Field	Meaning	Length in bytes	Contents after initialization
PASSWORD	Password as per PASSWORD command	4	4 x X'00'
LINK	Link name	8	8 x X'40'
NAME	File name	54	54 x X'40'

The descriptor FD is required in the ADD and SEL functions.

LA (Library Attributes)

The descriptor LA permits an administration privilege to be allocated for a library and the allowable storage mode to be set for its members. A library-wide borrowing mechanism can be initialized.

With WRITE-CTRL=DEACT there are no additional checks on creating or overwriting a version.

With WRITE-CTRL=ACTIV a version cannot be written unless the user ID of the person wishing to write it is recorded as the holder of the base version (which is always defined, either explicitly or implicitly) and either a new version is being created or the base version is being overwritten.
There is no base for the first version of a name; this version must be created by a user with the ADMIN privilege.
When a version is created or overwritten, a record of record type 2 is automatically created, recording the writer as HOLDER plus the DATE and TIME of the transaction. In addition the HOLD-STATE and HOLDER attributes and all privileges are applied to the new version unless the current operation calls for other values to be defined.

ACCESS-DATE=KEEP allows the date of last access to be recorded for members of the library.

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
P-TIND-ADMI	Protection type input field for administration N: no special protection Y: set standard protection G: set protection by guard Blank: the current setting remains unchanged	1	1 x X'40'
P-ADMI-OWN	Administration input field for owner N: no administration indicator Y: administration indicator Blank: the current setting remains unchanged	1	1 x X'40'
P-ADMI-GRP	Administration input field for group N: no administration privilege Y: administration privilege Blank: the current setting remains unchanged	1	1 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
P-ADMI-OTH	Administration input field for others N: no administration privilege Y: administration privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-ADMI-PIND	Administration password input field N: no password allocated Y: password allocated Blank: the current setting remains unchanged	1	1 x X'40'
P-ADMI-PSWD	Administration password	4	4 x X'00'
P-GUARD-ADMI	Administration guard	18	18 x X'40'
STORE-FORM	Storage mode for library ´N´: NONE: (same effect as STD) ´ ´: UNCHANGED ´S´: STD: (FULL or DELTA) ´V´: FULL: member ´D´: DELTA: member	1	1 x X'40'
WRITE-CTRL	Write control for library ´N´: NONE: (same effect as ´D´) ´ ´: UNCHANGED ´A´: activate write control for library ´D´: no write control (deactivate)	1	1 x X'40'
ACCESS-DATE	Record date of last access ´ ´: UNCHANGED ´N´: do not record date of access (NONE) ´K´: record date of access (KEEP)	1	1 x X'40'
	reserved	34	34 x X'40'

The descriptor LA is required for the function MODLA.

LD (Library Description)

The descriptor LD identifies a library. If a link name (LINK) exists, the associated library is processed. Any additional library name (NAME) specified for LD is ignored. LMS enters in the NAME field the library name associated with LINK. If no link name exists, the library identified by NAME is processed. The length of the library name is determined by the first blank. However, if a smaller value is entered in the MAX-NAME-LEN field, this value is assumed as the length of the library name. The library name must not exceed the length (1-54) entered in MAX-NAME-LEN. The value MAX-NAME-LEN=54 is recommended.

The NAME field is also used as a return parameter, depending on the LD-RETURN field of CB:

LD-RETURN=NO	If the library is specified via a link name, LMS enters the library name belonging to LINK in the NAME field. Otherwise the NAME field remains unchanged.
LD-RETURN=YES	LMS enters in the NAME field the full DMS file name of the library with catalog ID and user ID after ACS conversion, or the hit library in the case of library lists.

If LD-RETURN=YES is specified, note that the NAME field of LD must be supplied with the library name before each subroutine call if the same library is to be addressed.

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
PASSWORD	Password as per PASSWORD command	4	4 x X'00'
LINK	Link name	8	8 x X'40'
	reserved	8	8 x X'00'
MAX-NAME-LEN	Maximum length of library name	2	Length field
NAME	Library name	54	54 x X'40'

The descriptor LD is required in the following functions:

ADD, CLOSLIB, COPY, COPYLIB, COPYSTR, DEL, GSYSELEM, LOCK, LST, MODEA, MODEP, MODLA, MODTA, OPENGET, OPENPUT, OPENUPD, PROVIDE, REN, REORGLIB, RETURN, SEL, SHOWLA, SHOWTA, TOCPRIM, TOCSEC, UNLOCK.

LI (Library Information)

The descriptor LI describes the administration privilege and the initial values for member protection. Passwords are not displayed. However, the user is informed if a password is allocated. For allocation of the display fields see page 41.
The library size, the free 2-K pages, the library format and the UPAM protection are output as well. The permissible storage mode for library members and the indicators for the borrowing mechanism and for member access date recording are displayed.

Field	Meaning	Length in bytes	Contents after initialization
P-TIND-ADMI	Protection type indicator for administration	1	1 x X'40'
P-ADMI-OWN	Administration indicator for owner	1	1 x X'40'
P-ADMI-GRP	Administration indicator for group	1	1 x X'40'
P-ADMI-OTH	Administration indicator for others	1	1 x X'40'
P-ADMI-PIND	Administration password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-GUARD-ADMI	Administration guard indicator	18	18 x X'40'
STORE-FORM	Storage mode for library ´N´: NONE: (same effect as STD) ´S´: STD: (FULL or DELTA) ´V´: FULL: member ´D´: DELTA: member	1	1 x X'40'
WRITE-CTRL	Write control for library ´N´: NONE: (same effect as ´D´) ´A´: write control for library activated ´D´: no write control (deactivated)	1	1 x X'40'
ACCESS-DATE	Record date of last access ´N´: do not record date of access (NONE) ´K´: record date of access (KEEP)	1	1 x X'40'
	reserved	24	24 x X'40'
LIB-FORM	Library format ´2´: NK2 format ´4´: NK4 format	1	1 x X'40'

Field	Meaning	Length in bytes	Contents after initialization
UPAM-PROT	UPAM protection indicator ´Y´: library is UPAM-protected ´N´: library is not UPAM-protected	1	1 x X'40'
FILE-SIZE	Library size in 2-K pages	4	4 x X'00'
FREE-SIZE	Number of free 2-K pages	4	4 x X'00'
P-TIND-READ	Protection type indicator for read	1	1 x X'40'
P-READ-OWN	Read indicator for owner	1	1 x X'40'
P-READ-GRP	Read indicator for group	1	1 x X'40'
P-READ-OTH	Read indicator for others	1	1 x X'40'
P-READ-PIND	Read password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-WRIT	Protection type indicator for write	1	1 x X'40'
P-WRIT-OWN	Write indicator for owner	1	1 x X'40'
P-WRIT-GRP	Write indicator for group	1	1 x X'40'
P-WRIT-OTH	Write indicator for others	1	1 x X'40'
P-WRIT-PIND	Write password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-EXEC	Protection type indicator for execute	1	1 x X'40'
P-EXEC-OWN	Exec indicator for owner	1	1 x X'40'
P-EXEC-GRP	Exec indicator for group	1	1 x X'40'
P-EXEC-OTH	Exec indicator for others	1	1 x X'40'
P-EXEC-PIND	Exec password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-GUARD-READ	Read guard indicator	18	18 x X'40'
P-GUARD-WRIT	Write guard indicator	18	18 x X'40'
P-GUARD-EXEC	Exec guard indicator	18	18 x X'40'
P-TIND-HOLD	Borrowing privilege indicator	1	1 x X'40'
P-HOLD-OWN	Borrowing privilege indicator for owner	1	1 x X'40'
P-HOLD-GRP	Borrowing privilege indicator for group	1	1 x X'40'
P-HOLD-OTH	Borrowing privilege indicator for others	1	1 x X'40'
P-HOLD-PIND	Borrowing password indicator	1	1 x X'40'
	reserved	4	4 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
P-GUARD-HOLD	Borrowing guard indicator	18	18 x X'40'
	reserved	68	68 x X'40'

The descriptor LI is required for the function SHOWLA.

PA (Protection Attributes)

The descriptor PA defines member protection and the group of users with borrowing privileges, plus the presettings for these attributes.

Field	Meaning	Length in bytes	Contents after initialization
P-TIND-READ	Protection type input field for read N: no special protection Y: set standard protection G: set protection by guard Blank: the current setting remains unchanged	1	1 x X'40'
P-READ-OWN	Read input field for owner N: no read privilege Y: read privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-READ-GRP	Read input field for group N: no read privilege Y: read privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-READ-OTH	Read input field for others N: no read privilege Y: read privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-READ-PIND	Read password input field N: no password allocated Y: no password allocated Blank: the current setting remains unchanged	1	1 x X'40'
P-READ-PSWD	Read password	4	4 x X'00'
P-TIND-WRIT	Protection type input field for write N: no special protection Y: standard protection G: protection by guard Blank: the current setting remains unchanged	1	1 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
P-WRIT-OWN	Write input field for owner N: no write privilege Y: write privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-WRIT-GRP	Write input field for group N: no write privilege Y: write privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-WRIT-OTH	Write input field for others N: no write privilege Y: write privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-WRIT-PIND	Write password input field N: no password allocated Y: password allocated Blank: the current setting remains unchanged	1	1 x X'40'
P-WRIT-PSWD	Write password	4	4 x X'00'
P-TIND-EXEC	Protection type input field for execute N: no special protection Y: standard protection G: protection by guard Blank: the current setting remains unchanged	1	1 x X'40'
P-EXEC-OWN	Exec input field for owner N: no exec privilege Y: exec privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-EXEC-GRP	Exec input field for group N: no exec privilege Y: exec privilege Blank: the current setting remains unchanged	1	1 x X'40'

Field	Meaning	Length in bytes	Contents after initialization
P-EXEC-OTH	Exec input field for others N: no exec privilege	1	1 x X'40'
	Y: exec privilege Blank: the current setting remains unchanged		
P-EXEC-PIND	Exec password input field N: no password allocated Y: password allocated Blank: the current setting remains unchanged	1	1 x X'40'
P-EXEC-PSWD	Exec password	4	4 x X'00'
P-GUARD-READ	Read guard input field	18	18 x X'40'
P-GUARD-WRIT	Write guard input field	18	18 x X'40'
P-GUARD-EXEC	Exec guard input field	18	18 x X'40'
P-TIND-HOLD	Borrowing privilege input field N: no special protection Y: special protection is set G: borrowing privilege governed by GUARD Blank: the current setting remains unchanged	1	1 x X'40'
P-HOLD-OWN	Borrowing privilege input field for owner N: no borrowing privilege Y: borrowing privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-HOLD-GRP	Borrowing privilege input field for group N: no borrowing privilege Y: borrowing privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-HOLD-OTH	Borrowing privilege for others N: no borrowing privilege Y: borrowing privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-HOLD-PIND	Borrowing privilege password input field N: no password allocated	1	1 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
	Y: password allocated Blank: the current setting remains unchanged		
P-HOLD-PSWD	Borrowing privilege password	4	4 x X'00'
P-GUARD-HOLD	Borrowing guard input field	18	18 x X'40'
	reserved	84	84 x X'40'

The descriptor PA is required in the following functions: MODEP, MODLA, MODTA.

RD (Record Description)

The descriptor RD defines the retrieval address, the record type and the record length.

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
REC-ACC-ID	Access path identification	4	4 x X'FF'
BUFFER-LEN	Buffer length of ER	4	4 x X'00'
RECORD-LEN	Record length	4	4 x X'00'
	reserved	3	3 x X'00'
RECORD-TYP	Record type	1	1 x X'01'
RECORD-NR	Record number	4	4 x X'00'
	reserved	8	8 x X'00'

The descriptor RD is required in the following functions:

CLOSE, GET, OPENGET, OPENPUT, OPENUPD, PUT.

TA (Type Attributes)

The descriptor TA specifies the member type and the valid convention for that type. In the case of a user type the supertype can be set instead. TA can be used to assign an administration privilege to a type and to set the allowable storage mode for members of the defined type. A type-wide borrowing mechanism can be initialized (see page 35).

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
CONVENTION	Type convention N: no convention S: convention STD-SEQUENCE M: convention MULTI-SEQUENCE T: convention STD-TREE	1	1 x X'40'
	reserved	3	3 x X'40'
V-EXAMPLE	Version example for the convention STD-SEQUENCE and MULTI-SEQUENCE	24	24 x X'40'
P-TIND-ADMI	Protection type input field for administration N: no special protection Y: set standard protection G: set protection by guard Blank: the current setting remains unchanged	1	1 x X'40'
P-ADMI-OWN	Administration input field for owner N: no administration indicator Y: administration indicator Blank: the current setting remains unchanged	1	1 x X'40'
P-ADMI-GRP	Administration input field for group N: no administration privilege Y: administration privilege Blank: the current setting remains unchanged	1	1 x X'40'
P-ADMI-OTH	Administration input field for others N: no administration privilege Y: administration privilege Blank: the current setting remains unchanged	1	1 x X'40'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
P-ADMI-PIND	Administration password input field N: no password allocated Y: password allocated Blank: the current setting remains unchanged	1	1 x X'40'
P-GUARD-ADMI	Administration guard	18	18 x X'40'
STORE-FORM	Storage mode for type ´N´: NONE: setting as in library ´ ´: UNCHANGED ´S´: STD: (FULL or DELTA) ´V´: FULL: member ´D´: DELTA: member	1	1 x X'40'
WRITE-CTRL	Write control for type ´N´: NONE: write control as in library ´ ´: UNCHANGED ´A´: activate write control for type ´D´: no write control (deactivate)	1	1 x X'40'
SUPER-TYPE	Supertype input field <alphanum-name 1..8> ´*NONE´: there is no supertype Blank: the current setting remains unchanged	8	8 x X'40'
	reserved	47	47 x X'40'

The descriptor TA is required for the function MODTA.

TD (Type Description)

The descriptor TD specifies the member type.

Field	Meaning	Length in bytes	Contents after initialization
TYP	Member type	8	8 x X'40'
	reserved	8	8 x X'40'

The descriptor TD is required in the following functions: MODTA, SHOWTA.

TI (Type Information)

The descriptor TI supplies the member type, supertype, base type, valid convention, administration privilege and presettings for the protection attributes. For the exact allocation of the protection attributes, see page 41. Passwords are not displayed. However, the user is informed if a password is allocated. The permissible storage mode for the type and an indicator for the borrowing mechanism are displayed

Field	Meaning	Length in bytes	Contents after initialization
TYP	Member type	8	8 x X'40'
	reserved	8	8 x X'40'
CONVENTION	Type convention N: no convention S: convention STD-SEQUENCE M: Konvention MULTI-SEQUENCE T: convention STD-TREE	1	1 x X'40'
	reserved	3	8 x X'40'
V-EXAMPLE	Version example for the convention STD-SEQUENCE and MULTI-SEQUENCE	24	24 x X'40'
P-TIND-ADMI	Protection type indicator for administration	1	1 x X'40'
P-ADMI-OWN	Administration indicator for owner	1	1 x X'40'
P-ADMI-GRP	Administration indicator for group	1	1 x X'40'
P-ADMI-OTH	Administration indicator for others	1	1 x X'40'
P-ADMI-PIND	Administration password indicator	1	1 x X'40'
	reserved	4	4 x X'40'
P-GUARD-ADMI	Administration guard indicator	18	18 x X'40'

Field	Meaning	Length in bytes	Contents after initialization
STORE-FORM	Storage mode for type ´N´: NONE: setting as in library ´S´: STD: (FULL or DELTA) ´V´: FULL: member ´D´: DELTA: member	1	1 x X'40'
WRITE-CTRL	Write control for type ´N´: NONE: write control as in library ´A´: write control for type activated ´D´: no write control (deactivated)		1 x X'40'
SUPER-TYPE	Supertype indicator <alphanum-name 1..8> ´*NONE´: there is no supertype	8	8 x X'40'
BASIS-TYPE	Base type indicator <alphanum-name 1..8> ´*NONE´: there is no base type	8	8 x X'40'
	reserved	39	39 x X'40'
	****** Protection attributes ****		
P-TIND-READ	Protection type indicator for read	1	1 x X'40'
P-READ-OWN	Read indicator for owner	1	1 x X'40'
P-READ-GRP	Read indicator for group	1	1 x X'40'
P-READ-OTH	Read indicator for others	1	1 x X'40'
P-READ-PIND	Read password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-WRIT	Protection type indicator for write	1	1 x X'40'
P-WRIT-OWN	Write indicator for owner	1	1 x X'40'
P-WRIT-GRP	Write indicator for group	1	1 x X'40'
P-WRIT-OTH	Write indicator for others	1	1 x X'40'
P-WRIT-PIND	Write password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-TIND-EXEC	Protection type indicator for execute	1	1 x X'40'
P-EXEC-OWN	Exec indicator for owner	1	1 x X'40'
P-EXEC-GRP	Exec indicator for group	1	1 x X'40'
P-EXEC-OTH	Exec indicator for others	1	1 x X'40'

Field	Meaning	Length in bytes	Contents after initialization
P-EXEC-PIND	Exec password indicator	1	1 x X'40'
	reserved	4	4 x X'00'
P-GUARD-READ	Read guard indicator	18	18 x X'40'
P-GUARD-WRIT	Write guard indicator	18	18 x X'40'
P-GUARD-EXEC	Exec guard indicator	18	18 x X'40'
P-TIND-HOLD	Borrowing privilege indicator	1	1 x X'40'
P-HOLD-OWN	Borrowing privilege indicator for owner	1	1 x X'40'
P-HOLD-GRP	Borrowing privilege indicator for group	1	1 x X'40'
P-HOLD-OTH	Borrowing privilege indicator for others	1	1 x X'40'
P-HOLD-PIND	Borrowing password indicator	1	1 x X'40'
	reserved	4	4 x X'40'
P-GUARD-HOLD	Borrowing guard indicator	18	18 x X'40'
	reserved	52	52 x X'40'

The descriptor TI is required for the function SHOWTA.

TID (TOC Identification)

The parameter TID designates the desired TOC function via the TOC identification. This enables several TOCs to be searched concurrently. The values 1-10 are permitted for TID. The value must be entered in the field right-justified and in binary format.

Field	Meaning	Length in bytes	Contents after initialization
TID	TOC identification	4	current TOC-ID

The TOC parameter TID is required in the following functions:

TOC, TOCPRIM, TOCSEC.

Interrelationship between function and parameter structure

The following table shows which function requires which parameter structures. The numbers specify the order of the parameters for the respective functions. Optional parameters are given in parentheses.

Function (F code)	**Parameter structure**															
	CB	EA	ED	EI	EM	ER	FD	LA	LD	LI	PA	RD	TA	TD	TI	TID
ADD (X'08')	1		4(5)				2		3							
CLOSE (X'13')	1											2				
CLOSLIB(X'1C')	1								2							
COPY (X'0A')	1		3,5(6)						2,4							
COPYLIB (X'1B')	1								2,3							
COPYSTR (X'0B')	1		3,5						2,4							
DEL (X'07')	1		3						2							
END (X'02')	1															
GET (X'11')	1					3						2				
GSYSELEM (X'1F')	1		3						2							
INIT (X'01')	1															
LOCK (X'0C')	1		3						2							
LST (X'14')	1		3						2							
MODEA (X'1A')	1	4	3						2							
MODEP (X'15')	1		3						2		4					
MODLA (X'16')	1							3	2		4					
MODTA (X'18')	1								2		5		4	3		
OPENGET (X'0E')	1		4						3			2				
OPENPUT (X'0F')	1		4(5)						3			2				
OPENUPD (X'10')	1		4						3			2				
PROVIDE (X'1D')	1		3,5(6)						2,4							
PUT (X'12')	1					3						2				
REN (X'06')	1		3,4						2							
REORGLIB (X‘20‘)	1								2							
RETURN (X'1E')	1		3,5,6						2,4							
SEL (X'09')	1		3				4		2							

Function (F code)	**Parameter structure**															
	CB	EA	ED	EI	EM	ER	FD	LA	LD	LI	PA	RD	TA	TD	TI	TID
SHOWLA (X'17')	1								2	3						
SHOWTA (X'19')	1								2					3	4	
TOC (X'05')	1			3												2
TOCPRIM (X'03')	1			3	5				4							2
TOCSEC (X'04')	1			3	5				4							2
UNLOCK (X'0D')	1		3						2							

3.3 Overview of the subcodes

The subcodes can only be specified for specific subroutine functions. They are specified in the SUBCODE field of the control block CB. If no subcode is specified, the SUBCODE field in control block CB must be set to blank (X’40’ / UNUSE).

The subcodes determine:

- the output format of members
- how records and members are read or written
- control of version automation.

The following functions have a subcode:

TOCPRIM, TOCSEC, GET, CLOSE, LST, ADD, COPY, OPENGET, OPENPUT, PROVIDE, REN, RETURN and COPYLIB.

The table below shows all subcodes, their meanings and how they are assigned to the respective subroutine functions.

Subcode	**Meaning**	**Function**
SHORT	Output format:	
	Outputs member name and storage mode.	TOCPRIM
	Outputs member name, secondary name, secondary attribute and storage mode.	TOCSEC
LONG	Extended output format:	
	Outputs user, creation and modification dates and times, CCS name, protection attributes, borrowing privileges, member state, holder and access stamp in addition to member name and storage mode.	TOCPRIM
	Outputs user, creation, and modification dates and times, secondary name and secondary attribute, CCS name, protection attributes, borrowing privileges, member state, holder and access stamp in addition to member name and storage mode.	TOCSEC
DIR	Direct reading of a record: Outputs the record identified by RECORD-TYPE and RECORD-NR in control block RD.	GET
SEQ	Sequential reading of a record: Outputs next the record that has the same access path identification as the last GET call.	

Subcode	**Meaning**	**Function**
WRITE	Write output member back: Writes the last member state back into the library.	CLOSE
RESET	Discard output member: Discards the member state last written.	
SYM	Display mode of a record: Represents the records depending on the member type.	LST
HEX	Display mode of a record: Outputs the record character by character and in hexadecimal form (superimposed).	
INCP	Version automation: Increments with prefix	ADD COPY OPENPUT PROVIDE REN RETURN
INCB	Version automation: Increments with base	
HIGP	Version automation: Overwrites the member with the highest version in the prefix	
UNUSE	All versions are specified explicitly	
EXTRA	Opens a member for reading irrespective of the existence of format B records	OPENGET
UNUSE	For future extensions	All as yet unnamed functions

4 Subroutine functions

The following chapter begins with an overview of all subroutine functions and their meanings and continues with a description of the functions in alphabetical order.

4.1 Overview of the functions

Function	**(F code)**	**Meaning**
ADD	(X'08')	Incorporate a file as a member
CLOSE	(X'13')	Close a member
CLOSLIB	(X'1C')	Close a library
COPY	(X'0A')	Copy a member
COPYLIB	(X'1B')	Copy a library
COPYSTR	(X'0B')	Copy a delta tree
DEL	(X'07')	Delete a member
END	(X'02')	Terminate subroutine access
GET	(X'11')	Read a record
GSYSELEM	(X'1F')	Read a member specification from a system variable
INIT	(X'01')	Initiate subroutine access
LOCK	(X'0C')	Lock a member
LST	(X'14')	Output a member to SYSLST
MODEA	(X'1A')	Modify member attributes
MODEP	(X'15')	Modify member protection
MODLA	(X'16')	Modify library attributes
MODTA	(X'18')	Modify type attributes
OPENGET	(X'0E')	Open a member for reading
OPENPUT	(X'0F')	Open a member for writing
OPENUPD	(X'10')	Open a member for reading and writing
PROVIDE	(X'1D')	“Borrow” a member from a source library

Function	**(F code)**	**Meaning**
PUT	(X'12')	Write a record
REN	(X'06')	Rename a member
REORGLIB	(X‘20‘)	Reorganize a library
RETURN	(X'1E')	“Return” a member of a source library
SEL	(X'09')	Output a member to a file
SHOWLA	(X'17')	Show library attributes
SHOWTA	(X'19')	Show type attributes
TOC	(X'05')	Continue TOCPRIM or TOCSEC
TOCPRIM	(X'03')	Search for a member in the primary directory
TOCSEC	(X'04')	Search for a member in a secondary directory
UNLOCK	(X'0D')	Release a member that was locked

4.2 Description of the functions

The description of the LMS subroutine functions is structured as follows:

- short description of the functionality
- special features (e.g. subcodes)
- required parameter structures: call and return parameters

Note

The order in which the parameter structures are specified in the manual must be strictly adhered to.

ADD: incorporate a file as a member

ADD adds a file as a member to a library. FD specifies the file, ED1 the member and LD the library.

There are four possible subcode specifications: UNUSE, INCP, HIGP and INCB
Subcode specifications other than UNUSE result in special treatment of the version specification (see page 24).

Moreover, the user can enter a date or time of day in the fields USER-DATE and USER-TIME of ED1 respectively. These fields must contain blanks if LMS is to enter the current date and time of day.
The catalog attribute CCS is entered as a member attribute for the target member.

The parameters OVERWRITE, RKEY, DESTROY and PROT-IND are interpreted.

If OVERWRITE=EXTEND is specified, the following must be true:

– No ISAM keys may be present in the member.
– If the member contains file attributes (record type 164), these must match the attributes of the file.
– If the member contains no file attributes, the entry RECORD-FORMAT=FIXED is illegal for the file.
– The CCS name of the file must match that of the member.

OVERWRITE=NAME is not permitted.

The ED1.STORE-FORM field defines the storage mode for the member being added. The storage mode must not conflict with the type or library attribute settings, and all the members of one type and name must have the same storage mode. Delta members may be overwritten only if they are part of a delta tree.

STORE-FORM=FULL

The member is created in full storage mode (error if not allowed).

STORE-FORM=STD

The member is created in the storage mode appropriate to its scope. Conflicting requirements result in an error. If there are no special requirements, full storage mode is selected.
If the required storage mode is DELTA and the subcode is UNUSE, the base is defined as the standard base. With all other subcodes the base is defined by the ED1.VERSION specification.

STORE-FORM=DELTA

The member is created in delta storage mode (error if not allowed). This is a valid specification for members of types S, P, D, J, M and X and types derived from them. If the subcode is UNUSE, the ED2 descriptor defining the base member must also be specified. If version automation is used, ED2 is ignored and the base version must then be specified in ED1.

Handling of delta members without version automation

– If the member is to be stored as a delta member, the following must be true:

```
ED1.STORE-FORM=DELTA, ED1.TYP=ED2.TYP, ED1.NAME=ED2.NAME
```

– If the member is to be added as the first member of a delta tree, no member of the same type and name may exist and in addition the following must apply:

```
ED1.VERSION=ED2.VERSION
```

– If the member is to be added as a subsequent member of a delta tree, ED2 must be used to specify the existing base member and the following must apply:

```
ED1.VERSION ≠ ED2.VERSION
```

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’08’
	SUBCODE	Version automation
		(UNUSE or INCP or INCB or HIGP)
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	RKEY	ISAM key and file attributes
	OVERWRITE	Logical overwriting
	PROT-IND	Member protection
	LD-RETURN	Full DMS file name in LD

Parameter structure	**Field**	**Meaning**
FD		File descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	NAME	File name
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED1		Member descriptor (target)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	STORE-FORM	Storage mode
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user
ED2		Member descriptor (base) if ED1.STORE-FORM = DELTA and subcode = blank
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
FD		File descriptor
	NAME	File name (if file identified by link name)
LD		Library descriptor (target library)
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED1		Member descriptor (target member)
	VERSION	Member version (with call using *HIGH or with version automation)

CLOSE: close a member

CLOSE explicitly closes a member opened via OPENGET, OPENPUT or OPENUPD. The corresponding resources are released at the same time. Field REC-ACC-ID of RD is set to 4 x X’FF’.

Two subcode entries, which are only interpreted after OPENPUT or OPENUPD, are possible for CLOSE:

SUBCODE=WRITE

The member state written last is declared valid. The member which existed in the library under this name is overwritten and can no longer be accessed.

SUBCODE=RESET

The member state written last is canceled. The member which existed in the library under this name is retained.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'13'
	SUBCODE	Subcode (WRITE or RESET)
	ACC	Subroutine access identification
RD		Record descriptor
	REC-ACC-ID	Access path identification

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
RD		Record descriptor
	REC-ACC-ID	Access path identification

CLOSLIB: close a library

CLOSLIB explicitly closes a library. The library cannot be closed if any of its members are still open. If there are still open members, or if any member of the library is locked against modification (LOCK), CLOSLIB is rejected with a return code of X´18´ (illegal call sequence).

Libraries are opened by any of the following functions:

ADD, COPY, COPYSTR, DEL, LOCK, LST, MODEA, MODEP, MODLA, MODTA, OPENGET, OPENPUT, OPENUPD, PROVIDE, REN, RETURN, SEL, SHOWLA, SHOWTA, TOCPRIM, TOCSEC.

For performance reasons, once libraries have been opened they remain opened. They are generally closed implicitly by the END function or when there is a resource bottleneck. A subroutine user generally does not know whether a given library is still open. Therefore CLOSLIB can also be applied to closed libraries, producing a return code of X´00´ (OK).

When library lists are closed, there is no hit library, so no DMS file name can be determined. When library lists are used, the input in the NAME field of LD remains unchanged.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'1C'
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)

COPY: copy a member

COPY copies a member.

ED1 specifies the source member, LD1 the source library, ED2 the target member and LD2 the target library.

LD1 and LD2 may designate the same library. The complete type, name and version of the target member must be specified.

There are four possible subcode specifications: UNUSE, INCP, HIGP and INCB
Subcode specifications other than UNUSE result in special treatment of the version specification (see page 24).

Moreover, the user can enter a date or time of day in fields USER-DATE and USER-TIME of ED2 respectively. If the date and time of the input member are to be taken over, these fields must contain blanks.

Parameters OVERWRITE, DESTROY and PROT-IND are interpreted.

If OVERWRITE=EXTEND is specified, the following must be true:

– No ISAM keys may be present in the members.
– If the input and output members contain file attributes (record type 164), these must match.
– If only one of the members contains file attributes, RECORD-FORMAT=FIXED is illegal.
– The CCS name of the source member must match that of the target member.

Overwriting the target name space (OVERWRITE=NAME)

OVERWRITE=NAME can be used to make the copied member the only member of the target library with the same type and name. Before making the copy, LMS deletes from the target library any members which have the same type and name as the target member. Hence all user specifications for the target member (such as INCP) are applied to a now empty target name space. INCP, for example, always causes the default version to be generated.

Restrictions

– The input and output libraries must be different.
– If an error occurs while the target name space is being cleared (typically if there is a write-protected member), the COPY function is aborted.

The ED2.STORE-FORM field defines the storage mode for the member being added. The storage mode must not conflict with the type or library attribute settings, and all the members of one type and name must have the same storage mode. Delta members may be overwritten only if they are part of a delta tree.

STORE-FORM=FULL

The member is created in full storage mode (error if not allowed).

STORE-FORM=STD

The member is created in the storage mode appropriate to its scope. Conflicting requirements result in an error. If there are no special requirements, full storage mode is selected.
If the required storage mode is DELTA and the subcode is UNUSE, the base is defined as the standard base. With all other subcodes the base is defined by the ED2.VERSION specification.

STORE-FORM=DELTA

The member is created in delta storage mode (error if not allowed). This is a valid specification for members of types S, P, D, J, M and X and types derived from them. If the subcode is UNUSE, the ED3 descriptor defining the base member must also be specified. If version automation is used, ED3 is ignored and the base version must then be specified in ED2.

Handling of delta members without version automation

– If the member is to be included as a delta member, the following must apply:

```
ED2.STORE-FORM=DELTA, ED2.TYP=ED3.TYP, ED2.NAME=ED3.NAME
```

– If the member is to be included as the first member of a delta tree, no member of the same type and name may exist and in addition the following must apply:

```
ED2.VERSION=ED3.VERSION
```

– If the member is to be included as a subsequent member of a delta tree, ED3 must be used to specify the existing base member and the following must apply:

```
ED2.VERSION ≠ ED3.VERSION
```

Delta members can only be overwritten if they are part of a delta tree.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'0A'
	SUBCODE	Version automation (UNUSE or INCP or INCB or HIGP)
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	OVERWRITE	Logical overwriting
	PROT-IND	Member protection (STD or SAME)
	LD-RETURN	Full DMS file name in LD
LD1		Library descriptor (source library)
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of the library name
	NAME	Library name
ED1		Member descriptor (source member)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
LD2		Library descriptor (target library)
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name

Parameter structure	Field	Meaning
ED2		Member descriptor (target member)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	STORE-FORM	Storage mode
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user
ED3		Member descriptor (base member) if ED2.STORE-FORM = DELTA and subcode=blank
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD1		Source library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED1		Member descriptor (source member)
	VERSION	Member version (if call used *HIGH)
LD2		Target library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED2		Member descriptor (target member)
	VERSION	Member version (with call using *HIGH or with version automation)

COPYLIB: copy a library

COPYLIB permits logical copying of a complete library with all its library, type and member attributes.

LD1 defines the source library, LD2 the target library. The target library must not yet exist or it must have FCBTYPE=NONE.

With COPYLIB, the ATTR-IND field of the CB parameter is evaluated. If its value is “S” (SAME), the file protection attributes of the source library are applied to the target library.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’1B’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	ATTR-IND	Copy library attributes (STD/SAME)
	LD-RETURN	Full DMS file name in LD
LD1		Source library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
LD2		Target library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD1		Source library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
LD2		Target library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)

COPYSTR: copy a delta tree

COPYSTR copies a delta tree with its structure being retained.

ED1 specifies the delta tree to be copied, LD1 the source library, ED2 the type and name of the copied delta tree and LD2 the target library.

LD1 and LD2 may designate the same library.

The complete type and name of the input and output delta tree must be specified. No member of the type and name specified with ED2 may exist. Version information and user-specified date and time of day are not interpreted but transferred unchanged to the output delta tree. All versions are copied.

Parameters DESTROY and PROT-IND are interpreted so that a delta member can be physically deleted if necessary.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’0B’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	PROT-IND	Member protection (STD or SAME)
	LD-RETURN	Full DMS file name in LD
LD1		Library descriptor (source library)
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED1		Member descriptor (source member)
	TYP	Member type
	NAME	Member name

Parameter structure	Field	Meaning
LD2		Library descriptor (target library)
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED2		Member descriptor (target member)
	TYP	Member type
	NAME	Member name

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD1		Source library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
LD2		Target library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)

DEL: delete a member

DEL deletes a member. ED specifies the member to be deleted and LD the library containing this member. The complete type, name and version of the member must be specified.

The DESTROY parameter is interpreted.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’07’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	VERSION	Member version (if call used *HIGH)

END: terminate subroutine access

END is the last function call to LMS and terminates subroutine access. All resources (e.g. memory space) requested for this access are released. Field ACC of CB is set to 4 x X’FF’. Open members are implicitly closed via SUBCODE = RESET. Open libraries are closed as well.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’02’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	ACC	Subroutine access identification
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code

GET: read a record

GET reads a record.

The user is offered two subcodes for record selection.

- SUBCODE=SEQ
  The next record, relative to the last GET call with the same access path identification, is output to the ER field. During the call, only fields REC-ACC-ID (see OPENGET) and BUFFER-LEN need to be supplied. The record buffer must have been set to length BUFFER-LEN. After reading, the record type and number are stored in fields RECORD-TYPE and RECORD-NR of RD respectively. When the end of the member is reached, X’08’ is written to the RETURNCODE field of CB. A change of the record type does not cause a message but can be recognized by analyzing the RECORD-TYPE field.
- SUBCODE=DIR
  The RECORD-TYPE and RECORD-NR fields must also be defined. The record identified by RECORD-TYPE and RECORD-NR of RD (retrieval address) is output. If the record is not available, X’0C’ is written to the RETURNCODE field of CB. RD and ER remain unchanged.

SEQ and DIR may be freely combined.

When a record is read, the record header in ER is supplied by LMS. The record length value is set either to the record length of the record read or to the buffer length specified by the user in RD, whichever is the smaller. This is the maximum length in which the record is transferred. If the record is too long, X’04’ is written to the RETURNCODE field of CB. The record length/type specifications in ER are transferred to the corresponding fields of RD.

After reading, fields RECORD-TYPE and RECORD-LEN of RD contain the record type and length respectively and RECORD-NR contains the record number relative to the record type. RD contains the true record length which can be used in the event of an error to calculate the buffer length.

GET can be used, for instance, to read records of various record types alternately. If a member has been opened with OPENGET, it cannot be overwritten.

Record types 1-159, 163 and 164 can be read (see page 140).

Members containing format B records cannot be read with GET unless the EXTRA subcode is set in the OPENGET function.

When a format B record is read, sets the following values:

1. ER is the format B record without its 4-byte header. The ER buffer must be 256K long.
2. RECORD-LEN is the original length of the format B record
3. RECORD-TYPE is record type 160
4. RECORD-NR is the record number relative to the record type

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’11’
	ACC	Subroutine access identification
	SUBCODE	Subcode (DIR or SEQ)
RD		Record descriptor
	REC-ACC-ID	Access path identification
	BUFFER-LEN	Buffer length of ER
	RECORD-TYPE	Record type (optional for SEQ)
	RECORD-NR	Record number (optional for SEQ)
ER		Member record area
	-	Does not have to be supplied

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
RD		Record descriptor
	RECORD-LEN	Record length
	RECORD-TYPE	Record type
	RECORD-NR	Record number
ER		Member record area
		Record contents (including 4-byte record header)

GSYSELEM: read a member specification from a system variable

GSYSELEM reads the value of a system variable and interprets it as a member specification (library, member name, version and type).

The library (library or link name), member type and version may be predefined (non-blank value). If any of these components is left unspecified in the system variable (or is assigned the default value *BY-PROGRAM), it defaults to its predefined value when the function returns. It is an error for there to be no explicit value and no default value. Default values must be defined in the standard format for the subroutine (for example, for the highest available version: X'FF' and at least one blank).

System variable names must be specified in the NAME field of the ED structure. The subroutine looks first for a procedure-local system variable by that name, then for a task-global system variable by that name.

If the return code is anything other than LMSOK (X´00´), the fields in ED and LD are left unchanged.

Syntax of contents of system variables (lowercase is **not** converted to uppercase and is treated as syntactically invalid):

```
*LIBRARY-ELEMENT(...)
  | LIBRARY = *BY-PROGRAM / <full-filename 1..54 without-vers> / *LINK(...)
  |
  |    *LINK(...)
  |
  |      | LINK-NAME = <structured-name 1..8>
  |
  | ,ELEMENT = <composed-name 1..64 with-under>(...)
  |
  |    <composed-name 1..64 with-under>(...)
  |
  |      | VERSION = *BY-PROGRAM / *HIGHEST-EXISTING / *UPPER-LIMIT /
  |      |                 <composed-name 1..24 with-under>
  |
  | ,TYPE = *BY-PROGRAM / <alphanum-name 1..8>
```

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'1F'
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
LD		Library descriptor
	LINK	Link name (poss. predefined)
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name (poss. predefined)
ED		Member descriptor
	TYP	Member type (poss. predefined)
	NAME	Name of S variable
	VERSION	Member version (poss. predefined)

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
LD		Library descriptor
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

INIT: initialize subroutine access

INIT opens a subroutine access and is the first function call to LMS. The subroutine access identification is returned in field ACC of the CB. This identification is required for all further function calls. Field LMSVERSION of the CB contains the current LMS version number. The processing operands are set to default values.

Several INIT calls may be issued in succession if parallel subroutine accesses are to be implemented. A separate CB must be created for each subroutine access, since every INIT call causes a new subroutine access identification to be stored in field ACC of the CB.

If an error occurred during the INIT call, field ACC in the CB contains 4 x X’FF’.

Subroutine access is terminated with END and the appropriate subroutine access identification.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’01’
	SUBCODE	Subcode currently not used: UNUSE

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
	ACC	Subroutine access identification
	LMSVERSION	LMS version
	DESTROY	Physical overwriting
	FCB	File attributes (for Select)
	RKEY	ISAM key and file attributes
	OVERWRITE	Logical overwriting
	COLUMN-P	Columns per line
	LINE-P	Lines per page
	PROT-IND	Member protection
	ATTR-IND	Copy library attributes
	INFO	Member subarea to be processed
	LD-RETURN	Use NAME field of LD as return parameter for full DMS file name of library

LOCK: lock a member

LOCK explicitly locks a member against modification. ED specifies the member to be locked and LD specifies the library containing this member. The complete type, name and version of the member must be specified.

The member remains locked until UNLOCK or END is entered or the program is terminated. The lockout applies in memory only.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'0C'
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	VERSION	Member version (if call used *HIGH)

LST: output a member to SYSLST

LST outputs a member to SYSLST. ED specifies the member to be output and LD the library containing this member. The complete type, name and version of the member must be specified.

The parameters COLUM, LINE and INFO are taken into account during output to SYSLST. The value of INFO must be “S” (STD, process standard area). LST is allowed for all member types, with user types being treated in the same way as their base types.

LST offers the user two subcodes for determining the display type of a record:

- SUBCODE=SYM

  Records are displayed dependent on their member type:

  - Members are displayed character by character except for types P, R, L and C and types derived from them.
  - Type P members are also displayed character by character, with the first character of every record being interpreted as feed control character.
  - In type R members, ESD, ISD, RLD, TXT, TXTP, REP and END information is output in edited form. Other information, for example LSD or DSDD, is output in unedited form. This means that the record length field and, if applicable, the record number are also output. Continuous text information is not divided up.
  - Type L members (LLMs) are displayed in edited form.
  - Type C members (load modules) are displayed character by character and in hexadecimal form side by side.

- SUBCODE=HEX

  Records are displayed character by character and in hexadecimal form one over the other. This means that for every member record, two lines are output. In the first line, the record is displayed character by character and in the second line, in hexadecimal form.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’14’
	ACC	Subroutine access identification
	SUBCODE	Output format (SYM or HEX)
	COLUMN-P	Number of columns
	LINE-P	Number of lines
	INFO	Member subarea to be processed
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	VERSION	Member version (if call used *HIGH)

MODEA: modify member attributes

MODEA selects the member attributes for a specific member. ED determines the member and LD specifies the library containing this member. The complete type, name and version of the member must be specified.

The date and time can only be changed together. This means:

– If the USER-DATE in the EA descriptor is a blank, the USER-TIME in EA is ignored and the old member attributes are retained.
– If the USER-DATE in the EA descriptor is not a blank, the USER-TIME from EA is taken over unchanged.

MOD-DATE-IND controls whether the current time of day and the current date are recorded in the member as the time of last modification.

A member may be marked as borrowed or returned (see page 21).

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’1A’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Parameter structure	Field	Meaning
EA		Member attributes
	USER-DATE	- User date
	USER-TIME	- Time of day
	CCS-NAME	- Coded Character Set name
	HOLD-STATE	- Hold flag: state assigned to the member
	MOD-DATE-IND	MODIFICATION DATE INDICATOR controls updating of the modification date

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	VERSION	Member version (if call used *HIGH)

MODEP: modify member protection

MODEP selects member protection for a specific member. ED determines the member and LD specifies the library containing this member. The complete type, name and version of the member must be specified.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	SUBCODE	Subcode currently not used: UNUSE
	FUNCTION	Function code X’15’
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version
PA		Protection attributes
	P-TIND-READ	All PA fields (page 41) can be preset.
	...	

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	VERSION	Member version (if call used *HIGH)

MODLA: modify library attributes

MODLA determines the attributes for a library. LD specifies the library and LA defines its attributes. A library administration privilege and presettings for member protection for new members can be allocated. The storage mode allowed for members of the library can be defined, access date recording can be activated, and a borrowing mechanism can be applied.

PA defines presettings for member protection and the group of users with borrowing privileges for the library.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’16’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
LA		Library attributes
	P-TIND-ADMI	Protection type indicator for administration
	P-ADMI-OWN	Administration indicator for owner
	P-ADMI-GRP	Administration indicator for group
	P-ADMI-OTH	Administration indicator for others
	P-ADMI-PIND	Administration password indicator
	P-ADMI-PSWD	Administration password
	STORE-FORM	Storage mode for library
	WRITE-CTRL	Write control for library
	ACCESS-DATE	Record date of last access for members

Parameter structure	Field	Meaning
PA		Protection attributes
	P-TIND-READ	All PA fields (page 41) can be preset.
	...	

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)

MODTA: modify type attributes

MODTA defines type attributes.

LD specifies the library, TD the member type. TA sets the applicable (version) convention and the supertype. The storage mode allowed for members of the type can be defined and the borrowing mechanism can be activated. The type administration privilege determines the group of users with administration privileges for the type. Only users with this privilege are allowed to create, delete and rename members. These settings have priority over values set by MODLA.

PA defines presettings for member protection and the group of users with borrowing privileges for the type.

Notes

– The definition of the SUPER-TYPE must be non-recursive (tree structure).
– No supertypes can be declared for standard types (those one character long or beginning with $ or SYS).

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'18'
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
TD		Type descriptor
	TYP	Member type

Parameter structure	Field	Meaning
TA		Type attributes
	CONVENTION	Valid convention for the member type
	V-EXAMPLE	Version example for the convention STD-SEQUENCE and MULTI-SEQUENCE
	P-TIND-ADMI	Protection type input field for administration
	P-ADMI-OWN	Administration input field for owner
	P-ADMI-GRP	Administration input field for group
	P-ADMI-OTH	Administration input field for others
	P-ADMI-PIND	Administration password input field
	P-ADMI-PSWD	Administration password
	STORE-FORM	Storage mode for type
	WRITE-CTRL	Write control for type
	SUPER-TYPE	Supertype input field
PA		Protection attributes
	P-TIND-READ	All PA fields (page 41) can be preset.
	...	

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)

OPENGET: open a member for reading

OPENGET opens a member for reading. ED specifies the member to be read and LD the library containing this member. The complete type, name and version of the existing member must be specified.

During opening, an access path identification is stored in field REC-ACC-ID of RD. If an error occurred during opening, this identification has the value 4 x X'FF'.

A member can be multiply opened for reading if several OPENGETs with the same member name and library name are given. A separate RD should be created for each OPEN, as every OPENGET causes a new access path identification to be stored in field REC-ACC-ID of RD. This identification is required for all further accesses and for closing the member.

A member cannot be simultaneously opened with OPENGET, OPENPUT and OPENUPD.

After opening, the read pointer (for sequential reading) is located at the beginning of the member.

USER-DATE, USER-TIME and STORE-FORM information from ED is not interpreted.

Members containing format B records cannot be read with GET unless the EXTRA subcode is set in the OPENGET function.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’0E’
	SUBCODE	EXTRA: Format B records allowed in member UNUSE: otherwise
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
RD		Record descriptor
	-	Does not have to be supplied
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
RD		Record descriptor
	REC-ACC-ID	Access path identification
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	VERSION	Member version (if call used *HIGH)

OPENPUT: open a member for writing

OPENPUT opens a member for writing. ED1 specifies the member to be written and LD the library containing this member.
The complete type, name and version of the member must be specified.

There are four possible subcode specifications: UNUSE, INCP, HIGP and INCB
Subcode specifications other than UNUSE result in special treatment of the version specification (see page 24).

If the library does not already exist, it is created. Moreover, the user may enter a date or time of day in fields USERDATE and USER-TIME of ED1, respectively. If the current date and time are to be taken over, these fields must contain blanks.

Parameters OVERWRITE and DESTROY are interpreted. OVERWRITE=NAME is not allowed. Delta members can only be overwritten if they are part of a delta tree.

During opening, an access path identification is stored in field REC-ACC-ID of RD. If an error occurred during opening, this identification has the value 4 x X'FF'.

Several successive OPENPUT calls for different members, but not for the same member, are possible. A separate RD should be created for each call, as every OPENPUT causes a new access path identification to be written to field REC-ACC-ID of RD.

A member cannot be opened simultaneously with OPENGET, OPENPUT and OPENUPD.

Member types C and L are not allowed, as members of these types contain records which cannot be processed.

The ED1.STORE-FORM field defines the storage mode for the member being added. The storage mode must not conflict with the type or library attribute settings, and all the members of one type and name must have the same storage mode. Delta members may be overwritten only if they are part of a delta tree.

STORE-FORM=FULL

The member is created in full storage mode (error if not allowed).

STORE-FORM=STD

The member is created in the storage mode appropriate to its scope. Conflicting requirements result in an error. If there are no special requirements, full storage mode is selected.
If the required storage mode is DELTA and the subcode is UNUSE, the base is defined as the standard base. With all other subcodes the base is defined by the ED1.VERSION specification.

STORE-FORM=DELTA

The member is created in delta storage mode (error if not allowed). This is a valid specification for members of types S, P, D, J, M and X and types derived from them. If the subcode is UNUSE, the ED2 descriptor defining the base member must also be specified. If version automation is used, ED2 is ignored and the base version must then be specified in ED1.

Handling of delta members without version automation

– If the member is to be included as a delta member, the following must apply:

```
ED1.STORE-FORM=DELTA, ED1.TYP=ED2.TYP, ED1.NAME=ED2.NAME
```

– If the member is to be included as the first member of a delta tree, no member of the same type and name may exist and the following must apply in addition:

```
ED1.VERSION=ED2.VERSION
```

– If the member is to be included as a subsequent member of a delta tree, ED2 must be used to specify the existing base member and the following must apply:

```
ED1.VERSION ?  ED2.VERSION
```

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’0F’
	SUBCODE	Version automation
		(UNUSE or INCP or INCB or HIGP)
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	OVERWRITE	Logical overwriting
	LD-RETURN	Full DMS file name in LD
RD		Record descriptor
	-	Does not have to be supplied

Parameter structure	**Field**	**Meaning**
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED1		Member descriptor (target)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	STORE-FORM	Storage mode
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user
ED2		Member descriptor (base) if ED1.STORE-FORM = DELTA
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
RD		Record descriptor
	REC-ACC-ID	Access path identification
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED1		Target member descriptor
	VERSION	Member version (with version automation and *HIGH)

OPENUPD: open a member for reading and writing

OPENUPD opens a member for reading and writing. ED specifies the member to be read and written and LD the library containing this member. The complete type, name and version of the existing member must be specified. Moreover, the user may enter a date or time of day in fields USER-DATE and USER-TIME of ED, respectively. If the current date and time are to be taken over, these fields must contain blanks.

The DESTROY parameter is interpreted so that a member can be physically deleted if necessary.

During opening, an access path identification is stored in field REC-ACC-ID of RD. If an error occurred during opening, this identification has the value 4 x X’FF’.

Several successive OPENUPD calls for different members, but not for the same member, are possible. A separate RD should be created for each call, as every OPENUPD causes a new access path identification to be written to field REC-ACC-ID of RD.

GET and PUT accesses are coordinated. The old variant can still be read while a new variant is being written. GET and PUT calls issued under the same access path identification do not affect each other.

A member cannot be opened simultaneously with OPENGET, OPENPUT and OPENUPD.

Field STORE-FORM of ED is not interpreted; the old value cannot be changed.

Member types C and L are not allowed, as members of these types contain records which cannot be processed.

OPENUPD for delta members is only possible if they are part of a delta tree.

Even if only one record is to be changed, the whole member must be read and written.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’10’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	LD-RETURN	Full DMS file name in LD
RD		Record descriptor
	-	Does not have to be supplied
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor (source, target)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
RD		Record descriptor
	REC-ACC-ID	Access path identification
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Source and target member descriptor
	VERSION	Member version (with version automation and *HIGH)

PROVIDE: reserve and copy a member

PROVIDE reserves a member of a source library and makes a copy available in an output library. If write control is active, a reserved member is protected against modification by other users. If the appropriate convention is also set, the entire version space for the specified version is reserved for the holder.

ED1 specifies the source member, LD1 the source library, ED2 the target member and LD2 the target library. LD1 and LD2 may designate the same library. The complete type, name and version of the member must be specified.

There are four possible subcode specifications: UNUSE, INCP, HIGP and INCB
Subcode specifications other than UNUSE result in special treatment of the version specification (see page 24).

Moreover, the user can enter a date or time of day in the fields USER-DATE and USER-TIME of ED1. These fields must contain blanks if the date and time are to be copied from the input member.

The parameters OVERWRITE and DESTROY are interpreted.

OVERWRITE=EXTEND and OVERWRITE=NAME are not allowed.

The ED2.STORE-FORM field defines the storage mode for the member being added. The storage mode must not conflict with the type or library attribute settings, and all the members of one type and name must have the same storage mode. Delta members may be overwritten only if they are part of a delta tree.

STORE-FORM=FULL

The member is created in full storage mode (error if not allowed).

STORE-FORM=STD

The member is created in the storage mode appropriate to its scope. Conflicting requirements result in an error. If there are no special requirements, full storage mode is selected.
If the required storage mode is DELTA and the subcode is UNUSE, the base is defined as the standard base. With all other subcodes the base is defined by the ED2.VERSION specification.

STORE-FORM=DELTA

The member is created in delta storage mode (error if not allowed). This is a valid specification for members of types S, P, D, J, M and X and types derived from them. If the subcode is UNUSE, the ED3 descriptor defining the base member must also be specified. If version automation is used, ED3 is ignored and the base version must then be specified in ED2.

Handling of delta members without version automation

- If the member is to be stored as a delta member, the following must be true:

  ```
  ED2.STORE-FORM=DELTA, ED2.TYP=ED3.TYP, ED2.NAME=ED3.NAME
  ```

- If the member is to be added as the first member of a delta tree, no member of the same type and name may exist and in addition the following must apply:

  ```
  ED2.VERSION=ED3.VERSION
  ```

- If the member is to be added as a subsequent member of a delta tree, ED3 must be used to specify the existing base member and the following must apply:

  ```
  ED2.VERSION ?  ED3.VERSION
  ```

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'1D'
	SUBCODE	Version automation
		(UNUSE or INCP or INCB or HIGP)
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	OVERWRITE	Logical overwriting
	LD-RETURN	Full DMS file name in LD
LD1		Source library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED1		Source member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version
LD2		Target library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name

Parameter structure	Field	Meaning
ED2		Target member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	STORE-FORM	Storage mode
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user
ED3		Base member descriptor if ED2.STORE-FORM = DELTA and subcode=blank
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD1		Source library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED1		Source member descriptor
	VERSION	Member version (if call used *HIGH)
LD2		Target library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED2		Target member descriptor
	VERSION	Member version (with call using *HIGH or with version automation)

PUT: write a record

PUT writes a record. The record types may be written in any order. Within a record type, writing is performed sequentially. Record length (RECORD-LEN) and type (RECORD-TYPE) are taken from RD. The first 4 characters in the buffer must be reserved for the record header.

Frequent changes of the record type are not recommended because they lead to time and space problems.

Record types 1-159, 163 and 164 can be written (see page 140).

Record types must not be mixed for delta members.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’12’
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
RD		Record descriptor
	REC-ACC-ID	Access path identification
	RECORD-LEN	Record length
	RECORD-TYPE	Record type
ER		Record contents (including 4-byte record header)

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code

REN: rename a member

REN renames a member. ED1 specifies the member to be renamed, LD the library containing this member and ED2 the new designation of the member. The complete type, name and version must be specified both for the input member and for the output member. In addition, a valid user-defined date and time of day must be specified for the output member. If the date and time of the input member are to be taken over, the corresponding fields must contain blanks.

The date and time can only be changed together. This means:

– If the USER-DATE in the ED2 descriptor is a blank, the USER-TIME in ED2 is ignored and the old member attributes are retained.

– If the USER-DATE in the ED2 descriptor is not a blank, the USER-DATE and USER-TIME from ED2 are taken over unchanged.

The parameters DESTROY and OVERWRITE are interpreted.

OVERWRITE=EXTEND and OVERWRITE=NAME have the same effect as OVERWRITE=NO.

Delta members cannot be renamed. The output member must not yet exist as a delta member.

There are four possible subcode specifications: UNUSE, INCP, HIGP and INCB Subcode specifications other than UNUSE result in special treatment of the version specification (see page 24).

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’06’
	SUBCODE	Version automation (UNUSE or INCP or INCB or HIGP)
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	OVERWRITE	Logical overwriting
	LD-RETURN	Full DMS file name in LD

Parameter structure	**Field**	**Meaning**
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED1		Member descriptor (old)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
ED2		Member descriptor (new)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED1		Member descriptor (old)
	VERSION	Member version (if call used *HIGH)

REORGLIB: reorganize a library

REORGLIB reorganizes a library. In the process, as much unused storage space as possible is released at the end of the library file. This can often substantially reduce the amount of storage space needed by a library. If this space is not reduced, e.g. because following a system error blocks at the back may already be reserved but are not yet being used, or if the absolute minimum size of the library is to be reached, a buffer should be used for copying (COPY-LIBRARY).

The library is determined by LD.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'06'
	SUBCODE	Subcode currently not used: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)

RETURN: return a member

RETURN copies a member of a source library to an output library if the base specified for the target version has been reserved by the user in the output library. It deletes the member from the source library and cancels the reservations in the output library.

There is no base for the first version of a member. In this case administration authorization is required to call RETURN.

ED1 specifies the source member, LD1 the source library, ED2 the target member and LD2 the target library. LD1 and LD2 may designate the same library. The complete type, name and version of the member must be specified.

ED3 designates the base for member ED2. The following must apply:

```
ED2.TYP=ED3.TYP, ED2.NAME=ED3.NAME
```

If there is no version specified in ED3.VERSION (=blank), the version borrowed by the user is taken to be the base version (an error is reported in the event of ambiguity).

There are four possible subcode specifications: UNUSE, INCP, HIGP and INCB Subcode specifications other than UNUSE result in special treatment of the version specification (see page 24). The version specification in ED3 is ignored.

Moreover, the user can enter a date or time of day in the fields USER-DATE and USER-TIME of ED2. These fields must contain blanks if the date and time are to be copied from the input member.

The parameters OVERWRITE and DESTROY are interpreted.

OVERWRITE=EXTEND and OVERWRITE=NAME are not allowed.

The storage mode of the target member (ED2) is governed by the base (ED3).

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X'1E'
	SUBCODE	Version automation (UNUSE or INCP or INCB or HIGP)
	ACC	Subroutine access identification
	DESTROY	Physical overwriting
	OVERWRITE	Logical overwriting
	LD-RETURN	Full DMS file name in LD
LD1		Source library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED1		Source member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version
LD2		Target library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED2		Target member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	STORE-FORM	Storage mode
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user

Parameter structure	**Field**	**Meaning**
ED3		Base member descriptor
	TYP	Member type = ED2.TYP
	NAME	Member name = ED2.NAME
	VERSION	Member version (base version)

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD1		Source library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED1		Source member descriptor
	VERSION	Member version (if call used *HIGH)
LD2		Target library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED2		Target member descriptor
	VERSION	Member version (with version automation and *HIGH)

SEL: output a member to a file

SEL outputs a member to a file. ED specifies the member to be output, LD the library containing the member and FD the file to which it is to be output.The complete type, name and version of the member must be specified.

The parameters FCBTYPE, OVERWRITE, INFO and PROT-IND are interpreted.

The catalog attribute CCS is taken over from the member. In the case of OVERWRITE=EXTEND the CCS name of the member must match that of the file.

INFO=STD has the same effect as INFO=TXT. The text proper, i.e. record type 1, is output. With INFO=COM the separately stored comment, i.e. record type 2, is output.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’09’
	SUBCODE	Subcode currently not supported: UNUSE
	ACC	Subroutine access identification
	FCB	File attributes
	OVERWRITE	Logical overwriting
	INFO	Member subarea to be processed
	PROT-IND	File protection
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Parameter structure	Field	Meaning
FD		File descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	NAME	File name

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	VERSION	Member version (with call using *HIGH or with version automation)

SHOWLA: show library attributes

SHOWLA outputs library attributes. LD defines the library and LI provides the desired information.

The current administration privilege settings and the initial values for member protection are output. Passwords are not output. However, the user is informed that a password is allocated.

The library size, the free 2-K pages, the library format and a UPAM protection indicator are output as well.

The function shows the allowable storage mode for library members and the indicators for the borrowing mechanism and for member access date recording.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’17’
	SUBCODE	Subcode currently not supported: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
LI		Library information
	-	Does not have to be supplied

Return parameters

Parameter structure	**Field**	**Meaning**
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
LI	P-TIND-ADMI ...	LI fields (see page 38)

SHOWTA: show type attributes

SHOWTA outputs the current setting for the type attributes and the administration privilege and the initial values for member protection. Passwords are not displayed. However, the user is informed if a password is allocated. It also outputs the applicable (version) convention, the supertype, the base type, the permissible storage mode for members of the type, and the attribute defining the borrowing mechanism.

LD defines the library, TD the member type. TI returns the result.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’19’
	SUBCODE	Subcode currently not supported: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
TD		Type descriptor
	TYP	Member type
TI		Type information
	-	Type information

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
TI		Type information
	TYP	TI fields (see page 48)
	...	

TOC: continue TOCPRIM or TOCSEC

TOC continues either the TOCPRIM or the TOCSEC function, depending on the TOC identification previously defined in a TOCPRIM or TOCSEC function. The same TOC identification can be used any number of times. The last assignment always applies for a TOC identification.

The member information is written to the fields of EI as long as member entries satisfying the criteria specified with EM in the TOCPRIM or TOCSEC function are available. The output format depends on the member information selected by the subcode in TOCPRIM or TOCSEC (see the descriptions of the TOCPRIM and TOCSEC functions).

If no further member fulfills the defined criteria, the code EOF is entered in the RETURNCODE field of CB. The fields of EI remain unchanged.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’05’
	SUBCODE	Subcode currently not supported: UNUSE
	ACC	Subroutine access identification
TID		TOC identification
	-	Does not have to be supplied
EI		Member information
	-	Does not have to be supplied

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
EI		Member information
	TYP	EI fields (see page 26)
	...	filled as appropriate to subcode in TOCPRIM or TOCSEC

TOCPRIM: search for a member in the primary directory

TOCPRIM provides information on the member entries in the library specified with LD. An alphabetic search for the member entries (`TYP NAME VERSION`) is performed in the primary directory.

A valid TOC identification must be specified in each TOCPRIM call so that a subsequent TOC can refer to this identification. 1,...,10 are permissible values for a TOC identification. In the case of TOCPRIM functions with identical values for the TOC identification the last assignment applies.

Member information for the first member satisfying the criteria specified in EM is written to the fields of EI.

If no member fulfills the defined criteria, EOF is entered in the RETURNCODE field of CB. The fields of EI remain unchanged.

A member corresponds to the criteria if it matches the mask specified in EM, i.e. each field of the member must match the corresponding field or, in the case of the range specification for the member size, fields of the mask.

Evaluating the string fields of the EM mask:
Each string field is interpreted up to the first blank. A string field starting with a blank is assumed to be empty. A blank at the beginning of the mask is equivalent to the entry ’*’; the member field always matches the mask field.

Evaluating the E-SIZE-MIN and E-SIZE-MAX fields:
All members match whose size (in PAM pages) complies with the following equation: E-SIZE-MIN $\leq$ size $\leq$ E-SIZE-MAX. If X‘00000000‘ is entered for E-SIZE-MIN and X‘FFFFFFFF‘ for E-SIZE-MAX, any value can be used for selection.

In addition, two keywords may be entered in the VERSION field of EM:

– *HIGH or *HIGH|prefix

  Only the highest existing version or the highest existing version with a specified prefix is sought.

– *LOW or *LOW|prefix

  Only the lowest existing version or the lowest existing version with a specified prefix is sought.

These entries must start on the left in the VERSION field of EM and end with a blank. Any additional character entered in this field cancels the keywords. If a version ending in HIGH is to be sought, only ’**HIGH’ needs to be specified.

Two output formats are available:

– SHORT
  The member information returned comprises only the member designation (TYP, NAME, VERSION) and the storage mode (STORE-FORM). The remaining fields contain blanks.
  This output option is recommended when loops are programmed in the calling program and when only conditions for the member designation were specified with EM.

– LONG
  The member information is output with extensions 1 and 3. No secondary names are output. Fields SEC-NAME and SEC-ATTRIBUTE of EM are not interpreted.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	**Field**	**Meaning**
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’03’
	SUBCODE	Subcode (SHORT or LONG)
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
TID		TOC identification
	-	Does not have to be supplied
EI		Member information
	-	Does not have to be supplied
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name

Parameter structure	**Field**	**Meaning**
EM		Member mask (search pattern)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	STORE-FORM	Storage mode
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user
	CREATION-DATE	Date of member generation
	CREATION-TIME	Time of member generation
	MODIFI-DATE	Date of last update
	MODIFI-TIME	Time of last update
	P-TIND-READ	Protection type indicator for read
	P-READ-OWN	Read indicator for owner
	P-READ-GRP	Read indicator for group
	P-READ-OTH	Read indicator for others
	P-READ-PIND	Read password indicator
	P-TIND-WRIT	Protection type indicator for write
	P-WRIT-OWN	Write indicator for owner
	P-WRIT-GRP	Write indicator for group
	P-WRIT-OTH	Write indicator for others
	P-WRIT-PIND	Write password indicator
	P-TIND-EXEC	Protection type indicator for execute
	P-EXEC-OWN	Exec indicator for owner
	P-EXEC-GRP	Exec indicator for group
	P-EXEC-OTH	Exec indicator for others
	P-EXEC-PIND	Exec password indicator
	P-GUARD-READ	Read guard
	P-GUARD-WRIT	Write guard
	P-GUARD-EXEC	Exec guard
	CCS-NAME	Coded Character Set name
	P-TIND-HOLD	Borrowing privilege indicator
	P-HOLD-OWN	Borrowing privilege indicator for owner

Parameter structure	Field	Meaning
	P-HOLD-GRP	Borrowing privilege indicator for group
	P-HOLD-OTH	Borrowing privilege indicator for others
	P-HOLD-PIND	Borrowing password indicator
	P-GUARD-HOLD	Borrowing guard
	HOLD-STATE	Hold flag: member state
	ACCESS-DATE	Date of last access to member
	ACCESS-TIME	Time of last access to member
	E-SIZE-MIN	Lower limit for member size selection (PAM pages, 2-K unit)
	E-SIZE-MAX	Upper limit for member size selection (PAM pages, 2-K unit)

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
EI		Member information
	TYP	EI fields (see page 26)
	...	filled as appropriate to subcode in TOCPRIM or TOCSEC

TOCSEC: search for a member in the secondary directory

TOCSEC provides information on the member entries in the library specified with LD. An alphabetic search for the member entries (`TYP SEC-NAME SEC-ATTRIBUTE NAME VERSION`) is performed in the secondary directory.

A valid TOC identification must be specified in each TOCSEC call so that a subsequent TOC can refer to this identification. 1,...,10 are permissible values for a TOC identification. In the case of TOCSEC functions with identical values for the TOC identification the last TOCSEC assignment applies.

Member information for the first member satisfying the criteria specified in EM is written to the fields of EI.

If no member fulfills the defined criteria, EOF is entered in the RETURNCODE field of CB. The fields of EI remain unchanged.

A member corresponds to the criteria if it matches the mask specified in EM, i.e. each field of the member must match the corresponding field or, in the case of the range specification for the member size, fields of the mask.

Evaluating the string fields of the EM mask:
Each string field is interpreted up to the first blank. A string field starting with a blank is assumed to be empty. A blank at the beginning of the mask is equivalent to the entry ’*’; the member field always matches the mask field.

Evaluating the E-SIZE-MIN and E-SIZE-MAX fields:
All members match whose size (in PAM pages) complies with the following equation: E-SIZE-MIN $\leq$ size $\leq$ E-SIZE-MAX. If X‘00000000‘ is entered for E-SIZE-MIN and X‘FFFFFFFF‘ for E-SIZE-MAX, any value can be used for selection.

In addition to the mask fields of the TOCPRIM function, fields SEC-NAME and SEC-ATTRIBUTE of EM are used as search criteria. CSECT names which exceed 32 characters are truncated to this length before the wildcard comparison.

Two keywords may be entered in the VERSION field of EM:

– *HIGH

  Only the highest existing version is sought.

– *LOW

  Only the lowest existing version is sought.

These entries must start on the left in the VERSION field of EM and end with a blank. Any additional character entered in this field invalidates the keywords. If a version ending in HIGH is to be sought, only “**HIGH” needs to be specified.

Two output formats are available:

– SHORT
  The member information returned comprises only the member designation (TYP, NAME, VERSION), the secondary name (SEC-NAME), the secondary attribute (SEC-ATTRIBUTE) and the storage mode (STORE-FORM). The remaining fields contain blanks.
  This output option is recommended when loops are programmed in the calling program and when only conditions for the member designation, the secondary name and the secondary attribute were specified with EM.

– LONG
  The member information is output with extensions 1, 2 and 3. CSECT names which exceed 32 characters are truncated to this length for output in the SEC-NAME field. If such names are required in full, the secondary records of the member must be read with GET.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’04’
	SUBCODE	Subcode (SHORT or LONG)
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
TID		TOC identification
	-	Does not have to be supplied
EI		Member information
	-	Does not have to be supplied
LD		Library descriptor
	PASSWORD	Password as per PASSWORD command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name

Parameter structure	**Field**	**Meaning**
EM		Member mask (search pattern)
	TYP	Member type
	NAME	Member name
	VERSION	Member version
	STORE-FORM	Storage mode
	USER-DATE	Date specified by user
	USER-TIME	Time of day specified by user
	CREATION-DATE	Date of member generation
	CREATION-TIME	Time of member generation
	MODIFI-DATE	Date of last update
	MODIFI-TIME	Time of last update
	SEC-NAME	Reference name
	SEC-ATTRIBUTE	Reference attribute
	P-TIND-READ	Protection type indicator for read
	P-READ-OWN	Read indicator for owner
	P-READ-GRP	Read indicator for group
	P-READ-OTH	Read indicator for others
	P-READ-PIND	Read password indicator
	P-TIND-WRIT	Protection type indicator for write
	P-WRIT-OWN	Write indicator for owner
	P-WRIT-GRP	Write indicator for group
	P-WRIT-OTH	Write indicator for others
	P-WRIT-PIND	Write password indicator
	P-TIND-EXEC	Protection type indicator for execute
	P-EXEC-OWN	Exec indicator for owner
	P-EXEC-GRP	Exec indicator for group
	P-EXEC-OTH	Exec indicator for others
	P-EXEC-PIND	Exec password indicator
	P-GUARD-READ	Read guard
	P-GUARD-WRIT	Write guard
	P-GUARD-EXEC	Exec guard
	CCS-NAME	Coded Character Set name

Parameter structure	Field	Meaning
	P-TIND-HOLD	Borrowing privilege indicator
	P-HOLD-OWN	Borrowing privilege indicator for owner
	P-HOLD-GRP	Borrowing privilege indicator for group
	P-HOLD-OTH	Borrowing privilege indicator for others
	P-HOLD-PIND	Borrowing password indicator
	P-GUARD-HOLD	Borrowing guard
	HOLD-STATE	Hold flag: member state
	HOLDER	User ID of holder
	ACCESS-DATE	Date of last access to member
	ACCESS-TIME	Time of last access to member
	E-SIZE-MIN	Lower limit of member size selection (PAM pages, 2-K unit)
	E-SIZE-MAX	Upper limit of member size selection (PAM pages, 2-K unit)

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
EI		Member information
	TYP	EI fields (see page 26)
	...	filled as appropriate to subcode in TOCPRIM or TOCSEC

UNLOCK: release a member

UNLOCK releases a lockout which was set for a member with the aid of LOCK. ED specifies the member to be released and LD the library containing this member. The complete type, name and version of the member must be specified. The lockout can only be released within the subroutine access in which it was set.

Call parameters

The parameter structures must be given in the following sequence in the subroutine call.

Parameter structure	Field	Meaning
CB		Function control block
	SCBVERSION	Interface version
	FUNCTION	Function code X’0D’
	SUBCODE	Subcode currently not supported: UNUSE
	ACC	Subroutine access identification
	LD-RETURN	Full DMS file name in LD
LD		Library descriptor
	PASSWORD	Password as per password command
	LINK	Link name
	MAX-NAME-LEN	Maximum length of library name
	NAME	Library name
ED		Member descriptor
	TYP	Member type
	NAME	Member name
	VERSION	Member version

Return parameters

Parameter structure	Field	Meaning
CB		Function control block
	RETURNCODE	Return code
	LMS-MSG	LMS message code
	DMS-MSG	DMS message code
	PLAM-MSG	PLAM message code
LD		Library descriptor
	NAME	Full DMS file name of library (dependent on LD-RETURN field of CB)
ED		Member descriptor
	NAME	Member version (if call used *HIGH)

4.3 Programming aids

This section lists recommended symbolic names and describes records of the types 163 and 164.

4.3.1 Symbolic names

A number of symbolic names (equates) are available for the processing operands, function codes, subcodes, return codes and for the storage mode of members.

LMS processing parameter values

Symbol	**Value**	**Meaning**
		--- for general use ---
YES	'Y'	YES
NO	'N'	NO
NONE	'N'	NONE
ANY	' '	ANY
UNCH	' '	UNCHANGED
SAME	'M'	SAME
STD	'S'	STD
		--- for CB field FCB ---
ISAM	'I'	ISAM
SAM	'Q'	SAM
CAT	'C'	CAT
		--- for CB field OV ---
EXT	'E'	EXTEND
ONLY	'O'	ONLY
NAME	'A'	NAME
		--- for CB field INFO ---
TXT	X'01'	TEXT ONLY
COM	X'02'	COMMENT / DOCUMENTATION

Function codes

Symbol	Value	Meaning	Call parameters
INIT	X'01'	INIT	CB
END	X'02'	END	CB
TOCP	X'03'	TOCPRIM	CB, TID,EI, LD, EM
TOCS	X'04'	TOCSEC	CB, TID,EI, LD, EM
TOC	X'05'	TOC	CB, TID,EI
REN	X'06'	REN	CB, LD,ED1,ED2
DEL	X'07'	DEL	CB, LD, ED
ADD	X'08'	ADD	CB, FD, LD, ED1 [,ED2]
SEL	X'09'	SEL	CB, LD, ED, FD
COPY	X'0A'	COPY	CB, LD1,ED1,LD2,ED2 [,ED3]
COPST	X'0B'	COPYSTR	CB, LD1,ED1,LD2,ED2
LOCK	X'0C'	LOCK	CB, LD, ED
UNLK	X'0D'	UNLOCK	CB, LD, ED
OPENG	X'0E'	OPEN GET	CB, RD, LD, ED
OPENP	X'0F'	OPEN PUT	CB, RD, LD, ED1 [,ED2]
OPENU	X'10'	OPEN UPD	CB, RD, LD, ED
GET	X'11'	GET	CB, RD, ER
PUT	X'12'	PUT	CB, RD, ER
CLOSE	X'13'	CLOSE	CB, RD
LST	X'14'	LIST ELEMENT	CB, LD, ED
MEP	X'15'	MODIFY ELEMENT PROTECTION	CB, LD, ED, PA
MLA	X'16'	MODFIY LIBRARY ATTRIBUTES	CB, LD, LA, PA
SLA	X'17'	SHOW LIBRARY ATTRIBUTES	CB, LD, LI
MTA	X'18'	MODFIY TYPE ATTRIBUTES	CB, LD, TD, TA, PA
STA	X'19'	SHOW TYPE ATTRIBUTES	CB, LD, TD, TI
MEA	X'1A'	MODIFY ELEMENT ATTRIBUTES	CB, LD, ED, EA
COPLB	X'1B'	COPY LIBRARY	CB, LD1,LD2
CLOLB	X'1C'	CLOSE LIBRARY	CB, LD
PROVI	X'1D'	PROVIDE ELEMENT	CB, LD1,ED1,LD2,ED2 [,ED3]
RETUR	X'1E'	RETURN ELEMENT	CB, LD1,ED1,LD2,ED2,ED3
GSYSE	X'1F'	GET SYSELEM	CB, LD, ED
REOLB	X‘20‘	REORGANIZE LIB	CB, LD

Subcodes

Symbol	**Value**	**Meaning**
UNUSE	' '	DEFAULT
SHORT	'S'	TOC SHORT
LONG	'L'	TOC LONG
DIR	'D'	READ DIRECT
SEQ	'S'	READ SEQUENTIAL
WRITE	'W'	CLOSE OUTPUT ELEMENT FOR WRITE
RESET	'R'	FORGET OUTPUT ELEMENT
SYM	'S'	SHOW-ELEMENT SYMBOLIC
HEX	'H'	SHOW-ELEMENT ALPHA+HEX
INCP	'P'	INCREMENT WITH PREFIX
INCB	'B'	INCREMENT WITH BASE
HIGP	'H'	HIGHEST EXISTING WITH PREFIX
EXTRA	'X'	FORMAT-B RECORDS ALLOWED

Return codes

Symbol	**Value**	**Meaning**
OK	X'00'	OK
TRUNC	X'04'	RECORD TRUNCATED
EOF	X'08'	END OF GET/TOC
LMSER	X'0C'	LMS ERROR
PARER	X'14'	PARAMETER ERROR
SEQER	X'18'	SEQUENCE ERROR
INTER	X'1C'	LMS INTERNAL ERROR

Symbols for various fields

Symbol	Value	Meaning
		--- STORAGE MODE ---
FULL	'V'	Non-delta member
DELTA	'D'	Delta member
		--- CONVENTIONS ---
CNONE	'N'	NONE
CSEQ	'S'	STD-SEQUENCE
CTREE	'T'	STD-TREE
		--- PROTECTION INDICATORS ---
PNONE	'N'	NONE
PSTD	'Y'	STD-PROTECTION
PGD	'G'	PROTECTION BY GUARD
		--- SOURCE CODE CONTROL ---
FREE	'-'	FREE
INHLD	'H'	IN HOLD
		--- WRITE CONTROL ---
ACTIV	'A'	ACTIVATE ACTIVE
DEACT	'D'	DEACTIVATE DEACTIVATED
		--- ACCESS DATE ---
STD	'S'	Do not record date of access
KEEP	'K'	Record date of access
		--- MODIFICATION DATE INDICATOR ---
OLD	'O'	BY-SOURCE
SDAT	'S'	NEW (SYSTEM DATE)

4.3.2 Format of the secondary record (record type 163)

Secondary records are written to make entries in the secondary directory of a PLAM library. The secondary records must have the following format:

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
PM163LL	Record length	2	0
	reserved	1	0
PM163RT	Record type	1	163
PMSECNA	Secondary name	32	32 x X'40'
PMSECAT	Attribute for secondary name '0 ' - CSECT '1 ' - ENTRY	8	8 x X'40'
PMSFIND	Format indicator	1	0
PMSNAML	LONG SEC NAME: - 32K-45	varies	1 x X'40'

4.3.3 Format of the attribute record (record type 164)

The attribute record (if any) always has record type 164. The attribute record stores the attributes of the file which originally contained the data.

Field	**Meaning**	**Length in bytes**	**Contents after initialization**
PMRECLEN	Record length	2	540
	reserved	1	0
PMRECID	IDENTIFICATION OF PLAM RECORD	1	164
PMVERS	VERSION OF SPECIFIED PLAM RECORD	1	2
PMRECNUM	RECORD DESCRIBES FORMAT OF PLAM RECORD OF RECORD TYPE WITH SPECIFIED NUMBER	1	1
PMFNAME	FILE NAME TAKEN FROM FCB	54	54 x X'40'

PMFTYPE	FCBTYPE (SET/RESET)
	PMFTYPES=X'C0' : R SAM
	PMFTYPEI=X'40' : R ISAM
	PMFTYPEP=X'C0' : S PAM
	PMVMIN =X'01' * R VALPROP MIN. F.
	PMVMAX =X'01' * S VALPROP MAX. F.
		1	1 x X'00'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**

PMSHARE	SHARE (SET/RESET) CAT
	PMSHAREY=X'04' : S YES
	PMACCESS=X'08' : S ACCESS=READ
	PMSHCCNO=X'C0': R NO CONTROL CHAR
	PMSHCCM =X'40' : S MACHINE CODE CONTROL CHAR
	PMSHCCA =X'C0' : S ASA CONTROL CHAR
		1	1x X'00‘

PMSIZE	FILE SIZE	3	3 x X'00'
PMSALL	SECONDARY ALLOCATION	2	2 x X'00'

PMRECF	RECFORM (SET RESET)
	PMRECFF =X'04' : S FIXED
	PMRECFV =X'02' : S VARIABLE
	PMRECFU =X'06' : S UNDEFINED
		1	1 x X'00'

PMBLKSIZ	BLKSIZE	2	0
PMRECSIZ	RECSIZE	2	0
PMKEYPOS	KEYPOS	2	0
PMKEYLEN	KEYLEN	1	1 x X'00'
PMPAD	PAD	1	1 x X'00'
PMLOGLN	LOGLEN	1	1 x X'00'
PMVALLN	VALLEN	1	1 x X'00'

PMKEY	DOES KEY EXIST IN MEMBER?
	PMKEYY =C'Y' : YES
	PMKEYN =C'N' : NO
		1	1 x X'40'

PMCFID	CFID (not used)	4	4 x X'00'

PMCTRLI	BLKCTRL-INDICATOR
	PMCTRLN =X'80' : S BLKCTRL=NO
	PMCTRLP =X'40' : S BLKCTRL=PAMKEY
	PMCTRLD =X'20' : S BLKCTRL=DATA
	PMCTRL0 =X'10' : S BLKCTRL=NULL
	PMCTRLR =X'F0' : R BLKCTRL NOT SPECIF.
	PMBCF4K =X'08' : S BLOCK CTRL 4K
	PMBCF2K =X'04' : S BLOCK CTRL 2K
	PMCTRLU =X'03' : R BLKCTRL=RESERVED,0!
		1	1 x X'00'

PMPERF	IOPERF-INDICATOR
	PMPFVH =X'03' : S IOPERF=VERY-HIGH
	PMPFHI =X'02' : S IOPERF=HIGH
	PMPFST =X'01' : S IOPERF=STD
	PMPFNS =X'00' : S IOPERF NOT SPECIF.
		1	1 x X'00'

Field	**Meaning**	**Length in bytes**	**Contents after initialization**	

PMUSAG	IOUSAGE-INDICATOR
	PMUSRW =X'03' : S IOUSAGE=RDWRT
	PMUSWR =X'02' : S IOUSAGE=WRITE
	PMUSRD =X'01' : S IOUSAGE=READ
	PMUSNS =X'00' : S IOUSAGE NOT SPECIF.
		1	1 x X'00'	

PMEDMS3	CATALOG-INDICATOR (IDCEX) PMESPEC =X'08' : S PLAM FILE INDICATOR	1	1 x X'00'	
	reserved (must be 0)	1	1 x X'00'	
PMAIXCNT	ALTERNATE INDEX COUNT PMAIXMAX=30: MAX. NR. AIX ENTRIES	2	0	
PMFSIZ	FILE SIZE >= 32 GB PMSIZE must be X'FFFFFF'	4	4 x X'00'	
	reserved (must be 0)		86 x X'00'	
PMAIXNAM	KEYNAME	8	8 x X'00' *	*
PMAIXKPO	KEYPOS	2	0 *	
PMAIXKLE	KEYLEN	1	0 *	
PMAIXIND	INDICATOR PMAIXIDK=X'80' : S DUPKEY=YES : R DUPKEY=NO PMAIX# =12 : LENGTH OF AIX ENTRY		1 x X'00' *	

* 30 times (AIXMAX)

5 COBOL interface

5.1 Linkage module LMSUP1

LMS functions are called from COBOL programs via the linkage module LMSUP1, which is invoked as follows:

```
CALL "LMSUP1" USING parameter-list.
```

For entries in the parameter list see the description of the respective functions (page 17ff). Module LMSUP1 is fetched from library SYSLNK.LMS.034 and linked to the main program.

5.2 Generation of parameter structures for COBOL

In order for the COBOL parameter structures to be generated, library SYSLIB.LMS.033 (which contains the COPY members) must first be assigned. Assignment is possible using the following command:

```
/ADD-FILE-LINK FILE-NAME = $.SYSLIB.LMS.034, LINK-NAME = COBLIB
```

The installation location of this library is freely selectable via IMON.

The installation location of SYSLIB.LMS.034 can be determined and stored into an S variable using the builtin function INSTALLATION-PATH:

```
/SET-VARIABLE LIBRARY-NAME =INSTALLATION-PATH             -
                            (LOGICAL-ID = ´SYSLIB´        -
                            ,INSTALLATION-UNIT = ´LMS´    -
                            ,VERSION = ´3.4´              -
                            ,DEFAULT-PATH-NAME = ´$.SYSLIB.LMS.034´)
```

Then the library can be assigned as follows:

```
/ADD-FILE-LINK FILE-NAME=&(LIBRARY-NAME),LINK-NAME=COBLIB
```

No COPY members are available for parameter structures TID and ER.

LMSCOBCB

LMSCOBCB generates the function control block.

```
COPY LMSCOBCB.
```

Expansion of LMSCOBCB

```
    01  LMSUP-SCB.
        02  SCBVERSION PIC X(2)       VALUE "04".
        02  FUNC       PIC X(1)       VALUE LMSUP-UPINIT.
        02  SUBCODE    PIC X(1)       VALUE SPACE.
        02  ACC        PIC S9(9) COMP VALUE -1.
        02  RETURNCODE PIC X(1)       VALUE LOW-VALUE.
           88 LMSRET-OK               VALUE LMSUP-OK.
           88 LMSRET-TRUNC            VALUE LMSUP-TRUNC.
           88 LMSRET-EOF              VALUE LMSUP-EOF.
           88 LMSRET-LMSERR           VALUE LMSUP-LMSERR.
           88 LMSRET-PARERR           VALUE LMSUP-PARERR.
           88 LMSRET-SEQERR           VALUE LMSUP-SEQERR.
           88 LMSRET-INTERR           VALUE LMSUP-INTERR.
        02  FILLER     PIC X(1)       VALUE LOW-VALUE.
        02  LMS-MSG    PIC 9(4) COMP  VALUE 0.
        02  DMS-MSG    PIC 9(4) COMP  VALUE 0.
        02  PLAM-MSG   PIC 9(4) COMP  VALUE 0.
        02  LMSVERSION PIC X(12)      VALUE SPACES.
        02  FILLER     PIC 9(5) COMP  VALUE 0.
   ***                                                              ****
   **************  LMS   PARAMETERS  **********************************
   ***                                                              ****
        02  DESTROY    PIC X(1)       VALUE SPACE.
        02  FCB        PIC X(1)       VALUE SPACE.
        02  RKEY       PIC X(1)       VALUE SPACE.
        02  OVERWRITE  PIC X(1)       VALUE SPACE.
        02  COLUMN-P   PIC 9(4) COMP  VALUE 0.
        02  LINE-P     PIC 9(4) COMP  VALUE 0.
        02  PROT-IND   PIC X(1)       VALUE SPACE.
        02  ATTR-IND   PIC X(1)       VALUE SPACE.
        02  INFO       PIC X(1)       VALUE SPACE.
        02  FILLER     PIC X(5)       VALUE SPACES.
   **************  END OF COPY ELEMENT LMSCOBCB   **********************
```

LMSCOBEA

LMSCOBEA generates the member attributes.

```
COPY LMSCOBEA.
```

Expansion of LMSCOBEA

```
01  LMSUP-EA.
      02  USER-DATE     PIC X(14)      VALUE SPACES.
      02  USER-TIME     PIC X(8)       VALUE SPACES.
      02  CCS-NAME      PIC X(8)       VALUE SPACES.
      02  HOLD-STATE    PIC X(1)       VALUE SPACES.
 *        HOLD FLAG:                '-' : FREE
 *                                  'H' : INHOLD
 *                                  ' ' : UNCHANGE
      02  FILLER        PIC X(8)       VALUE SPACES.
      02  MOD-DATE-IND  PIC X(1)       VALUE "O".
 *    MODIFICATION DATE INDICATOR:  'O' : OLD
 *                                  'S' : SYSTEM DATE
      02  FILLER        PIC X(56)      VALUE SPACES.


 **************  END OF COPY ELEMENT   LMSCOBEA   *********************
```

LMSCOBED

LMSCOBED generates the member description.

```
COPY LMSCOBED.
```

Expansion of LMSCOBED

```
01  LMSUP-ED.
      02 LMSUP-ED-ELEM.
       03  TYP          PIC X(8)       VALUE SPACES.
       03  NAME         PIC X(64)      VALUE SPACES.
       03  VERSION      PIC X(24)      VALUE SPACES.
 *
      02  STORE-FORM    PIC X          VALUE "V".
      02  USER-DATE     PIC X(14)      VALUE SPACES.
      02  USER-TIME     PIC X(8)       VALUE SPACES.
 **************  END OF COPY ELEMENT LMSCOBED   *************************
```

LMSCOBEI

LMSCOBEI generates the member information.

```
COPY LMSCOBEI.
```

Expansion of LMSCOBEI

```
01  LMSUP-EI.
      02  LMSUP-EI-ED.
       03  LMSUP-EI-ED-ELEM.
        04  TYP         PIC X(8)       VALUE SPACES.
        04  NAME        PIC X(64)      VALUE SPACES.
        04  VERSION     PIC X(24)      VALUE SPACES.
 *
       03  STORE-FORM   PIC X(1)       VALUE SPACES.
       03  USER-DATE    PIC X(14)      VALUE SPACES.
       03  USER-TIME    PIC X(8)       VALUE SPACES.
 *
      02  CREATION-DATE PIC X(14)      VALUE SPACES.
      02  CREATION-TIME PIC X(8)       VALUE SPACES.
      02  MODIFI-DATE   PIC X(14)      VALUE SPACES.
      02  MODIFI-TIME   PIC X(8)       VALUE SPACES.
      02  SEC-NAME      PIC X(32)      VALUE SPACES.
      02  SEC-ATTRIBUTE PIC X(8)       VALUE SPACES.
      02  FILLER        PIC X(5)       VALUE SPACES.

 ***********    PROTECTION ATTRIBUTES   **************************
      02  P-TIND-READ   PIC X(1)       VALUE SPACES.
      02  P-READ-OWN    PIC X(1)       VALUE SPACES.
      02  P-READ-GRP    PIC X(1)       VALUE SPACES.
      02  P-READ-OTH    PIC X(1)       VALUE SPACES.
      02  P-READ-PIND   PIC X(1)       VALUE SPACES.
      02  FILLER        PIC S9(9) COMP VALUE 0.

      02  P-TIND-WRIT   PIC X(1)       VALUE SPACES.
      02  P-WRIT-OWN    PIC X(1)       VALUE SPACES.
      02  P-WRIT-GRP    PIC X(1)       VALUE SPACES.
      02  P-WRIT-OTH    PIC X(1)       VALUE SPACES.
      02  P-WRIT-PIND   PIC X(1)       VALUE SPACES.
      02  FILLER        PIC S9(9) COMP VALUE 0.

      02  P-TIND-EXEC   PIC X(1)       VALUE SPACES.
      02  P-EXEC-OWN    PIC X(1)       VALUE SPACES.
      02  P-EXEC-GRP    PIC X(1)       VALUE SPACES.
      02  P-EXEC-OTH    PIC X(1)       VALUE SPACES.
      02  P-EXEC-PIND   PIC X(1)       VALUE SPACES.
      02  FILLER        PIC S9(9) COMP VALUE 0.

      02  P-GUARD-READ  PIC X(18)       VALUE SPACES.
```

```
       02  P-GUARD-WRIT  PIC X(18)       VALUE SPACES.
       02  P-GUARD-EXEC  PIC X(18)       VALUE SPACES.
       02  CCS-NAME      PIC X(8)        VALUE SPACES.

       02  P-TIND-HOLD   PIC X(1)       VALUE SPACES.
       02  P-HOLD-OWN    PIC X(1)       VALUE SPACES.
       02  P-HOLD-GRP    PIC X(1)       VALUE SPACES.
       02  P-HOLD-OTH    PIC X(1)       VALUE SPACES.
       02  P-HOLD-PIND   PIC X(1)       VALUE SPACES.
       02  FILLER        PIC S9(9) COMP VALUE 0.
       02  P-GUARD-HOLD  PIC X(18)      VALUE SPACES.

       02  HOLD-STATE    PIC X(1)       VALUE SPACES.
*          HOLD FLAG:                '-' : FREE
*                                    'H' : INHOLD
       02  HOLDER        PIC X(8)       VALUE SPACES.
       02  ACCESS-DATE   PIC X(14)      VALUE SPACES.
       02  ACCESS-TIME   PIC X(8)       VALUE SPACES.
       02  FILLER        PIC X(1)       VALUE SPACES.
       02  ELEMENT-SIZE  PIC S9(9) COMP VALUE 0.
       02  FILLER        PIC X(40)      VALUE SPACES.

**************  END OF COPY ELEMENT   LMSCOBEI   **********************
```

LMSCOBEM

LMSCOBEM generates the member mask.

```
COPY LMSCOBEM.
```

Expansion of LMSCOBEM

```
    01  LMSUP-EM.
          02  TYP           PIC X(20)      VALUE SPACES.
          02  NAME          PIC X(132)     VALUE SPACES.
          02  VERSION       PIC X(52)      VALUE SPACES.
          02  STORE-FORM    PIC X(6)       VALUE SPACES.
          02  USER-DATE     PIC X(32)      VALUE SPACES.
          02  USER-TIME     PIC X(20)      VALUE SPACES.
          02  CREATION-DATE PIC X(32)      VALUE SPACES.
          02  CREATION-TIME PIC X(20)      VALUE SPACES.
          02  MODIFI-DATE   PIC X(32)      VALUE SPACES.
          02  MODIFI-TIME   PIC X(20)      VALUE SPACES.
          02  SEC-NAME      PIC X(68)      VALUE SPACES.
          02  SEC-ATTRIBUTE PIC X(20)      VALUE SPACES.
          02  FILLER        PIC X(14)      VALUE SPACES.

     ***********    PROTECTION ATTRIBUTES   **************************
          02  P-TIND-READ   PIC X(1)       VALUE SPACES.
          02  P-READ-OWN    PIC X(1)       VALUE SPACES.
          02  P-READ-GRP    PIC X(1)       VALUE SPACES.
          02  P-READ-OTH    PIC X(1)       VALUE SPACES.
          02  P-READ-PIND   PIC X(1)       VALUE SPACES.
          02  FILLER        PIC S9(9) COMP VALUE 0.

          02  P-TIND-WRIT   PIC X(1)       VALUE SPACES.
          02  P-WRIT-OWN    PIC X(1)       VALUE SPACES.
          02  P-WRIT-GRP    PIC X(1)       VALUE SPACES.
          02  P-WRIT-OTH    PIC X(1)       VALUE SPACES.
          02  P-WRIT-PIND   PIC X(1)       VALUE SPACES.
          02  FILLER        PIC S9(9) COMP VALUE 0.

          02  P-TIND-EXEC   PIC X(1)       VALUE SPACES.
          02  P-EXEC-OWN    PIC X(1)       VALUE SPACES.
          02  P-EXEC-GRP    PIC X(1)       VALUE SPACES.
          02  P-EXEC-OTH    PIC X(1)       VALUE SPACES.
          02  P-EXEC-PIND   PIC X(1)       VALUE SPACES.
          02  FILLER        PIC S9(9) COMP VALUE 0.

          02  P-GUARD-READ  PIC X(40)       VALUE SPACES.
          02  P-GUARD-WRIT  PIC X(40)       VALUE SPACES.
          02  P-GUARD-EXEC  PIC X(40)       VALUE SPACES.
          02  CCS-NAME      PIC X(20)       VALUE SPACES.
```

```
           02  P-TIND-HOLD   PIC X(1)       VALUE SPACES.
           02  P-HOLD-OWN    PIC X(1)       VALUE SPACES.
           02  P-HOLD-GRP    PIC X(1)       VALUE SPACES.
           02  P-HOLD-OTH    PIC X(1)       VALUE SPACES.
           02  P-HOLD-PIND   PIC X(1)       VALUE SPACES.
           02  FILLER        PIC S9(9) COMP VALUE 0.
           02  P-GUARD-HOLD  PIC X(40)      VALUE SPACES.

           02  HOLD-STATE    PIC X(1)       VALUE SPACES.
      *        HOLD FLAG:                '-' : FREE
      *                                  'H' : INHOLD
      *                                  ' ' : ANY
           02  HOLDER        PIC X(20)      VALUE SPACES.
           02  ACCESS-DATE   PIC X(32)      VALUE SPACES.
           02  ACCESS-TIME   PIC X(20)      VALUE SPACES.
           02  FILLER        PIC X(3)       VALUE SPACES.
           02  E-SIZE-MIN    PIC S9(9) COMP VALUE 0.
           02  E-SIZE-MAX    PIC S9(9) COMP VALUE -1.
           02  FILLER        PIC X(64)      VALUE SPACES.

      **************  END OF COPY ELEMENT   LMSCOBEM   **********************
```

LMSCOBFD

LMSCOBFD generates the file description.

```
COPY LMSCOBFD.
```

Expansion of LMSCOBFD

```
 01  LMSUP-FD.
       02  PASSWORD      PIC S9(9) COMP VALUE 0.
          02 PASSWORD-X    REDEFINES PASSWORD   PIC X(4).
       02  LINK          PIC X(8)       VALUE SPACES.
       02  NAME          PIC X(54)      VALUE SPACES.
  **************  END OF COPY ELEMENT   LMSCOBFD   *******************
```

LMSCOBLA

LMSCOBLA generates the library attributes.

```
COPY LMSCOBLA.
```

Expansion of LMSCOBLA

```
 01  LMSUP-LA.
       02  P-TIND-ADMI   PIC X(1)       VALUE SPACES.
       02  P-ADMI-OWN    PIC X(1)       VALUE SPACES.
       02  P-ADMI-GRP    PIC X(1)       VALUE SPACES.
       02  P-ADMI-OTH    PIC X(1)       VALUE SPACES.
       02  P-ADMI-PIND   PIC X(1)       VALUE SPACES.
       02  P-ADMI-PSWD   PIC S9(9) COMP VALUE 0.
          02 P-ADMI-PSWD-X REDEFINES P-ADMI-PSWD PIC X(4).
       02  P-GUARD-ADMI  PIC X(18)      VALUE SPACES.
       02  STORE-FORM    PIC X(1)       VALUE SPACES.
  *        STORAGE FORM FOR LIBRARY  'S' : STD ( FULL OR DELTA )
  *                                  'V' : FULL  ELEMENT
  *                                  'D' : DELTA ELEMENT
       02  WRITE-CTRL    PIC X(1)       VALUE SPACES.
  *        WRITE-CONTROL FOR LIBRARY 'A' : ACTIVATE
  *                                  'D' : DEAVTIVATE
       02  ACCESS-DATE   PIC X(1)       VALUE SPACES.
       02  FILLER        PIC X(34)      VALUE SPACES.

  **************  END OF COPY ELEMENT   LMSCOBLA   *********************
```

LMSCOBLD

LMSCOBLD generates the library description.

```
COPY LMSCOBLD.
```

Expansion of LMSCOBLD

```
001  LMSUP-LD.
      02  PASSWORD      PIC S9(9) COMP VALUE 0.
         02 PASSWORD-X    REDEFINES PASSWORD   PIC X(4).
      02  LINK          PIC X(8)       VALUE SPACES.
      02  FILLER        PIC S9(9) COMP VALUE 0.
      02  FILLER        PIC S9(9) COMP VALUE 0.
      02  MAX-NAME-LEN  PIC 9(4) COMP  VALUE 54.
      02  NAME          PIC X(54)      VALUE SPACES.
 **************  END OF COPY ELEMENT   LMSCOBLD   ********************
```

Note:

When allocating passwords, it is important to ensure that the password is not shorter than 4 bytes. The reason for this is that COBOL transfers data from left to right to the destination field during a MOVE and fills the remaining positions (right) with blanks. However, in BS2000, blanks represent a valid password combination.

LMSCOBLI

LMSCOBLI generates the library information.

```
COPY LMSCOBLI.
```

Expansion of LMSCOBLI

```
 1  LMSUP-LI.
       02  P-TIND-ADMI   PIC X(1)       VALUE SPACES.
       02  P-ADMI-OWN    PIC X(1)       VALUE SPACES.
       02  P-ADMI-GRP    PIC X(1)       VALUE SPACES.
       02  P-ADMI-OTH    PIC X(1)       VALUE SPACES.
       02  P-ADMI-PIND   PIC X(1)       VALUE SPACES.
       02  FILLER        PIC S9(9) COMP VALUE 0.
       02  P-GUARD-ADMI  PIC X(18)      VALUE SPACES.

       02  STORE-FORM    PIC X(1)       VALUE SPACES.
  *        STORAGE FORM FOR LIBRARY  'S' : STD ( FULL OR DELTA )
  *                                  'V' : FULL  ELEMENT
  *                                  'D' : DELTA ELEMENT
       02  WRITE-CTRL    PIC X(1)       VALUE SPACES.
  *        WRITE-CONTROL FOR LIBRARY 'A' : ACTIVE
  *                                  'D' : DEAVTIVATED
       02  ACCESS-DATE   PIC X(1)       VALUE SPACES.
       02  FILLER        PIC X(24)      VALUE SPACES.
       02  LIB-FORM      PIC X(1)       VALUE SPACES.
  *        LIBRARY FORMAT (NK2/NK4)   '2' : NK2 LIBRARY FORMAT
  *                                   '4' : NK4 LIBRARY FORMAT
       02  UPAM-PROT     PIC X(1)       VALUE SPACES.
  *        UPAM PROTECTED (YES/NO)    'Y' : LIB IS UPAM PROTECTED
  *                                   'N' : LIB IS NOT UPAM PROT.
       02  FILE-SIZE     PIC S9(9) COMP VALUE 0.
       02  FREE-SIZE     PIC S9(9) COMP VALUE 0.

       02  P-TIND-READ   PIC X(1)       VALUE SPACES.
       02  P-READ-OWN    PIC X(1)       VALUE SPACES.
       02  P-READ-GRP    PIC X(1)       VALUE SPACES.
       02  P-READ-OTH    PIC X(1)       VALUE SPACES.
       02  P-READ-PIND   PIC X(1)       VALUE SPACES.
       02  FILLER        PIC S9(9) COMP VALUE 0.

       02  P-TIND-WRIT   PIC X(1)       VALUE SPACES.
       02  P-WRIT-OWN    PIC X(1)       VALUE SPACES.
       02  P-WRIT-GRP    PIC X(1)       VALUE SPACES.
       02  P-WRIT-OTH    PIC X(1)       VALUE SPACES.
       02  P-WRIT-PIND   PIC X(1)       VALUE SPACES.
       02  FILLER        PIC S9(9) COMP VALUE 0.
```

```
           02  P-TIND-EXEC   PIC X(1)       VALUE SPACES.
           02  P-EXEC-OWN    PIC X(1)       VALUE SPACES.
           02  P-EXEC-GRP    PIC X(1)       VALUE SPACES.
           02  P-EXEC-OTH    PIC X(1)       VALUE SPACES.
           02  P-EXEC-PIND   PIC X(1)       VALUE SPACES.
           02  FILLER        PIC S9(9) COMP VALUE 0.

           02  P-GUARD-READ  PIC X(18)      VALUE SPACES.
           02  P-GUARD-WRIT  PIC X(18)      VALUE SPACES.
           02  P-GUARD-EXEC  PIC X(18)      VALUE SPACES.

           02  P-TIND-HOLD   PIC X(1)       VALUE SPACES.
           02  P-HOLD-OWN    PIC X(1)       VALUE SPACES.
           02  P-HOLD-GRP    PIC X(1)       VALUE SPACES.
           02  P-HOLD-OTH    PIC X(1)       VALUE SPACES.
           02  P-HOLD-PIND   PIC X(1)       VALUE SPACES.
           02  FILLER        PIC S9(9) COMP VALUE 0.
           02  P-GUARD-HOLD  PIC X(18)      VALUE SPACES.
           02  FILLER        PIC X(68)      VALUE SPACES.

      **************  END OF COPY ELEMENT   LMSCOBLI   *********************
```

LMSCOBPA

LMSCOBPA generates the protection attributes.

```
COPY LMSCOBPA.
```

Expansion of LMSCOBPA

```
01  LMSUP-PA.
      02  P-TIND-READ   PIC X(1)       VALUE SPACES.
      02  P-READ-OWN    PIC X(1)       VALUE SPACES.
      02  P-READ-GRP    PIC X(1)       VALUE SPACES.
      02  P-READ-OTH    PIC X(1)       VALUE SPACES.
      02  P-READ-PIND   PIC X(1)       VALUE SPACES.
      02  P-READ-PSWD   PIC S9(9) COMP VALUE 0.
         02 P-READ-PSWD-X REDEFINES P-READ-PSWD PIC X(4).

      02  P-TIND-WRIT   PIC X(1)       VALUE SPACES.
      02  P-WRIT-OWN    PIC X(1)       VALUE SPACES.
      02  P-WRIT-GRP    PIC X(1)       VALUE SPACES.
      02  P-WRIT-OTH    PIC X(1)       VALUE SPACES.
      02  P-WRIT-PIND   PIC X(1)       VALUE SPACES.
      02  P-WRIT-PSWD   PIC S9(9) COMP VALUE 0.
         02 P-WRIT-PSWD-X REDEFINES P-WRIT-PSWD PIC X(4).

      02  P-TIND-EXEC   PIC X(1)       VALUE SPACES.
      02  P-EXEC-OWN    PIC X(1)       VALUE SPACES.
      02  P-EXEC-GRP    PIC X(1)       VALUE SPACES.
      02  P-EXEC-OTH    PIC X(1)       VALUE SPACES.
      02  P-EXEC-PIND   PIC X(1)       VALUE SPACES.
      02  P-EXEC-PSWD   PIC S9(9) COMP VALUE 0.
         02 P-EXEC-PSWD-X REDEFINES P-EXEC-PSWD PIC X(4).

      02  P-GUARD-READ  PIC X(18)      VALUE SPACES.
      02  P-GUARD-WRIT  PIC X(18)      VALUE SPACES.
      02  P-GUARD-EXEC  PIC X(18)      VALUE SPACES.

      02  P-TIND-HOLD   PIC X(1)       VALUE SPACES.
      02  P-HOLD-OWN    PIC X(1)       VALUE SPACES.
      02  P-HOLD-GRP    PIC X(1)       VALUE SPACES.
      02  P-HOLD-OTH    PIC X(1)       VALUE SPACES.
      02  P-HOLD-PIND   PIC X(1)       VALUE SPACES.
      02  P-HOLD-PSWD   PIC S9(9) COMP VALUE 0.
         02 P-HOLD-PSWD-X REDEFINES P-HOLD-PSWD PIC X(4).
      02  P-GUARD-HOLD  PIC X(18)      VALUE SPACES.
      02  FILLER        PIC X(84)      VALUE SPACES.

 **************  END OF COPY ELEMENT   LMSCOBPA   ********************
```

LMSCOBRD

LMSCOBRD generates the record description.

```
COPY LMSCOBRD.
```

Expansion of LMSCOBRD

```
    01  LMSUP-RD.
          02  REC-ACC-ID    PIC S9(9) COMP VALUE -1.
          02  BUFFER-LEN    PIC 9(9) COMP  VALUE 0.
          02  RECORD-LEN    PIC 9(9) COMP  VALUE 0.
          02  FILLER        PIC X(3)       VALUE LOW-VALUE.
          02  RECORD-TYPE   PIC X          VALUE LMSUP-ONE.
          02  RECORD-NR     PIC 9(9) COMP  VALUE 0.
          02  FILLER        PIC S9(9) COMP VALUE 0.
          02  FILLER        PIC S9(9) COMP VALUE 0.
     **************  END OF COPY ELEMENT   LMSCOBRD   ********************
```

LMSCOBTA

LMSCOBTA generates the type attributes.

```
COPY LMSCOBTA.
```

Expansion of LMSCOBTA

```
 01  LMSUP-TA.
       02  CONVENTION    PIC X(1)       VALUE SPACES.
       02  FILLER        PIC X(3)       VALUE SPACES.
       02  V-EXAMPLE     PIC X(24)      VALUE SPACES.

       02  P-TIND-ADMI   PIC X(1)       VALUE SPACES.
       02  P-ADMI-OWN    PIC X(1)       VALUE SPACES.
       02  P-ADMI-GRP    PIC X(1)       VALUE SPACES.
       02  P-ADMI-OTH    PIC X(1)       VALUE SPACES.
       02  P-ADMI-PIND   PIC X(1)       VALUE SPACES.
       02  P-ADMI-PSWD   PIC S9(9) COMP VALUE 0.
          02 P-ADMI-PSWD-X REDEFINES P-ADMI-PSWD PIC X(4).
       02  P-GUARD-ADMI  PIC X(18)      VALUE SPACES.

       02  STORE-FORM    PIC X(1)       VALUE SPACES.
  *        STORAGE FORM FOR TYPE     'N' : NONE
  *                                  'S' : STD ( FULL OR DELTA )
  *                                  'V' : FULL  ELEMENT
  *                                  'D' : DELTA ELEMENT
       02  WRITE-CTRL    PIC X(1)       VALUE SPACES.
  *        WRITE-CONTROL FOR TYPE    'A' : ACTIVATE
  *                                  'D' : DEAVTIVATE
       02  SUPER-TYPE    PIC X(8)       VALUE SPACES.
       02  FILLER        PIC X(47)      VALUE SPACES.


  **************  END OF COPY ELEMENT   LMSCOBTA   *********************
```

LMSCOBTD

LMSCOBTD generates the type description.

```
COPY LMSCOBTD.
```

Expansion of LMSCOBTD

```
     01  LMSUP-TD.
           02  TYP            PIC X(8)       VALUE SPACES.
           02  FILLER         PIC X(8)       VALUE SPACES.

      **************  END OF COPY ELEMENT   LMSCOBTD   *********************
```

LMSCOBTI

LMSCOBTI generates the type information.

```
COPY LMSCOBTI.
```

Expansion of LMSCOBTI

```
     01  LMSUP-TI.
           02  TYP            PIC X(8)       VALUE SPACES.
           02  FILLER         PIC X(8)       VALUE SPACES.
           02  CONVENTION     PIC X(1)       VALUE SPACES.
           02  FILLER         PIC X(3)       VALUE SPACES.
           02  V-EXAMPLE      PIC X(24)      VALUE SPACES.

           02  P-TIND-ADMI    PIC X(1)       VALUE SPACES.
           02  P-ADMI-OWN     PIC X(1)       VALUE SPACES.
           02  P-ADMI-GRP     PIC X(1)       VALUE SPACES.
           02  P-ADMI-OTH     PIC X(1)       VALUE SPACES.
           02  P-ADMI-PIND    PIC X(1)       VALUE SPACES.
           02  FILLER         PIC S9(9) COMP VALUE 0.
           02  P-GUARD-ADMI   PIC X(18)      VALUE SPACES.

           02  STORE-FORM     PIC X(1)       VALUE SPACES.
      *        STORAGE FORM FOR TYPE     'N' : NONE
      *                                  'S' : STD ( FULL OR DELTA )
      *                                  'V' : FULL  ELEMENT
      *                                  'D' : DELTA ELEMENT
           02  WRITE-CTRL     PIC X(1)       VALUE SPACES.
      *        WRITE-CONTROL FOR TYPE    'A' : ACTIVE
      *                                  'D' : DEAVTIVATED
           02  SUPER-TYPE     PIC X(8)       VALUE SPACES.
           02  BASIS-TYPE     PIC X(8)       VALUE SPACES.
           02  FILLER        PIC X(39)      VALUE SPACES.

      ****************   PROTECTION ATTRIBUTES   ******************
           02  P-TIND-READ   PIC X(1)       VALUE SPACES.
           02  P-READ-OWN    PIC X(1)       VALUE SPACES.
           02  P-READ-GRP    PIC X(1)       VALUE SPACES.
           02  P-READ-OTH    PIC X(1)       VALUE SPACES.
           02  P-READ-PIND   PIC X(1)       VALUE SPACES.
           02  FILLER        PIC S9(9) COMP VALUE 0.

           02  P-TIND-WRIT   PIC X(1)       VALUE SPACES.
           02  P-WRIT-OWN    PIC X(1)       VALUE SPACES.
           02  P-WRIT-GRP    PIC X(1)       VALUE SPACES.
           02  P-WRIT-OTH    PIC X(1)       VALUE SPACES.
           02  P-WRIT-PIND   PIC X(1)       VALUE SPACES.
           02  FILLER        PIC S9(9) COMP VALUE 0.

           02  P-TIND-EXEC   PIC X(1)       VALUE SPACES.
           02  P-EXEC-OWN    PIC X(1)       VALUE SPACES.
           02  P-EXEC-GRP    PIC X(1)       VALUE SPACES.
           02  P-EXEC-OTH    PIC X(1)       VALUE SPACES.
           02  P-EXEC-PIND   PIC X(1)       VALUE SPACES.
           02  FILLER        PIC S9(9) COMP VALUE 0.

           02  P-GUARD-READ  PIC X(18)       VALUE SPACES.
           02  P-GUARD-WRIT  PIC X(18)       VALUE SPACES.
           02  P-GUARD-EXEC  PIC X(18)       VALUE SPACES.

           02  P-TIND-HOLD   PIC X(1)       VALUE SPACES.
           02  P-HOLD-OWN    PIC X(1)       VALUE SPACES.
           02  P-HOLD-GRP    PIC X(1)       VALUE SPACES.
           02  P-HOLD-OTH    PIC X(1)       VALUE SPACES.
           02  P-HOLD-PIND   PIC X(1)       VALUE SPACES.
           02  FILLER        PIC S9(9) COMP VALUE 0.
           02  P-GUARD-HOLD  PIC X(18)      VALUE SPACES.
           02  FILLER        PIC X(52)       VALUE SPACES.

      **************  END OF COPY ELEMENT   LMSCOBTI   *********************
```

5.3 Programming aids

LMSCOBEQ symbolic names

LMSCOBEQ must be entered in the SYMBOLIC CHARACTERS clause of the SPECIAL-NAMES paragraph. It defines symbolic names which serve as a programming aid for processing operand values, function codes, subcodes, return codes and storage mode of members.

When using LMSCOBEQ it is important to bear in mind that the SPECIAL-NAMES paragraph is an input record. If LMSCOBEQ is the only entry in the SPECIAL-NAMES paragraph, the concluding period must be explicitly set in the program.

```
COPY LMSCOBEQ.
```

Expansion of LMSCOBEQ

```
***************  LMSUP PARAMETER VALUES      *************************
           LMSUP-YES         IS  233
           LMSUP-NO          IS  214
           LMSUP-NONE        IS  214
           LMSUP-ANY         IS   65
           LMSUP-UNCHANGE    IS   65
           LMSUP-SAME        IS  213
           LMSUP-STD         IS  227

  **************  FOR: CBFCB                  *************************
           LMSUP-ISAM        IS  202
           LMSUP-SAM         IS  217
           LMSUP-CATA        IS  196

  **************  FOR: CBOV                   *************************
           LMSUP-EXT         IS  198
           LMSUP-ONLY        IS  215
           LMSUP-NAME        IS  194

  **************  FOR: INFO                   *************************
           LMSUP-TXT         IS  2
           LMSUP-COM         IS  3

  **************  LMSUP FUNCTION VALUES       *************************
           LMSUP-UPINIT      IS  02
           LMSUP-UPEND       IS  03
           LMSUP-TOCPRIM     IS  04
           LMSUP-TOCSEC      IS  05
           LMSUP-TOC         IS  06
           LMSUP-REN         IS  07
```

```
                  LMSUP-DEL         IS  08
                  LMSUP-ADD         IS  09
                  LMSUP-SEL         IS  10
                  LMSUP-COPY        IS  11
                  LMSUP-COPSTRUC    IS  12
                  LMSUP-LOCK        IS  13
                  LMSUP-UNLOCK      IS  14
                  LMSUP-OPEN-GET    IS  15
                  LMSUP-OPEN-PUT    IS  16
                  LMSUP-OPEN-UPD    IS  17
                  LMSUP-GET         IS  18
                  LMSUP-PUT         IS  19
                  LMSUP-CLOSE       IS  20
                  LMSUP-LIST        IS  21
                  LMSUP-MOD-EL-PROT IS  22
                  LMSUP-MOD-LIB-A   IS  23
                  LMSUP-SHOW-LIB-A  IS  24
                  LMSUP-MOD-TYP-A   IS  25
                  LMSUP-SHOW-TYP-A  IS  26
                  LMSUP-MOD-EL-A    IS  27
                  LMSUP-COPY-LIB    IS  28
                  LMSUP-CLOSE-LIB   IS  29
                  LMSUP-PROVIDE     IS  30
                  LMSUP-RETURN      IS  31
                  LMSUP-GET-SYS-EL  IS  32
                  LMSUP-REORG-LIB   IS  33

*************** LMSUP SUBCODE VALUES        *************************
                  LMSUP-UNUSED      IS   65
                  LMSUP-SHORT       IS  227
                  LMSUP-LONG        IS  212
                  LMSUP-DIR         IS  197
                  LMSUP-SEQ         IS  227
                  LMSUP-WRITE       IS  231
                  LMSUP-RESET       IS  218
                  LMSUP-SYM         IS  227
                  LMSUP-HEX         IS  201
                  LMSUP-INC-PRE     IS  216
                  LMSUP-INC-BASE    IS  195
                  LMSUP-HIGH-PRE    IS  201
                  LMSUP-EXTRA       IS  232

*************** LMSUP RETURN CODE VALUES  ****************************
                  LMSUP-OK          IS   1
                  LMSUP-TRUNC       IS   5
                  LMSUP-EOF         IS   9
                  LMSUP-LMSERR      IS  13
                  LMSUP-PARERR      IS  21
                  LMSUP-SEQERR      IS  25
                  LMSUP-INTERR      IS  29
```

```
**************  STORAGE FORM VALUES         ************************
         LMSUP-FULL    IS  230
         LMSUP-DELTA   IS  197

**************  CONVENTIONS                 ************************
         LMSUP-CNONE   IS  214
         LMSUP-CSEQ    IS  227
         LMSUP-CMSEQ   IS  213
         LMSUP-CTREE   IS  228

**************  PROTECTION INDICATORS       ************************
         LMSUP-PNONE   IS  214
         LMSUP-PSTD    IS  233
         LMSUP-PGUARD  IS  200

**************  SOURCE CODE CONTROL         ************************
         LMSUP-FREE    IS  97
         LMSUP-INHOLD  IS  201

**************  WRITE CONTROL               ************************
         LMSUP-ACTIV   IS  194
         LMSUP-DEACTIV IS  197

**************  ACCESS DATE                 ************************
         LMSUP-KEEP    IS  211

**************  MODIFICATION DATE INDICATOR ************************
         LMSUP-OLD      IS  215
         LMSUP-SYS-DATE IS  227

**************  OTHER VALUES                ************************
         LMSUP-ONE     IS  2
         LMSUP-TWO     IS  3

**************  END OF COPY ELEMENT   LMSCOBEQ   *******************
```

Format of a record of type 163

The record LMSCO163 is described below as an example of a type 163 record (see section “Format of the secondary record (record type 163)” on page 140.

```
COPY LMSCO163.
```

Expansion of LMSCO163

```
01 LMS-REC163.
     02 LEN             PIC 9(4) COMP  VALUE 0.
     02 FILLER          PIC X          VALUE LOW-VALUE.
     02 REC-TYPE        PIC X          VALUE SA-163.
     02  SEC-NAME       PIC X(32)      VALUE SPACES.
     02  SEC-ATTRIBUTE PIC X(8)        VALUE SPACES.
 *          '0        ' :  -  CSECT
 *          '1        ' :  -  ENTRY
     02  SEC-ATTR-BIT    REDEFINES SEC-ATTRIBUTE  PIC X.
     02  SEC-FORMAT-IND PIC X          VALUE LOW-VALUE.
     02  SEC-NAME-LONG.
     03  SEC-NAME-BYTE1 PIC X          VALUE SPACE.
 ***                                                              ****
 **************  END OF COPY ELEMENT   LMSCO163    *******************
```

Format of a record of type 164

The LMSCO164 record is described below as an example of a record of type 164 (see section “Format of the attribute record (record type 164)” on page 140).

```
COPY LMSCO164.
```

Expansion of LMSCO164

```
     01   LMS-REC164.
          02  LEN            PIC 9(4) COMP  VALUE 540.
          02  FILLER         PIC X          VALUE LOW-VALUE.
          02  REC-TYPE       PIC X          VALUE SA-164.
          02  VERSION        PIC X          VALUE LMSUP-TWO.
          02  REC-NUMBER     PIC X          VALUE LMSUP-ONE.
          02  FILENAME       PIC X(54)      VALUE SPACES.
          02  FCBTYPE-FIELD PIC X           VALUE LOW-VALUE.
 *              FCBTYPE (SET/RESET)
 *                X'C0'               R SAM
 *                X'40'               S ISAM
 *                X'C0'               S PAM
 *                X'01'               R VALPROP MIN. FUNCT.
 *                X'01'               S VALPROP MAX. FUNCT.
          02  SHARE-FIELD    PIC X          VALUE LOW-VALUE.
 *              SHARE (SET,RESET)
 *                X'04'               S YES
 *                X'08'               S ACCESS=READ
 *                X'C0'               R NO CONTROL CHAR
 *                X'40'               S MACHINE CODE CONTROL CHAR
 *                X'C0'               S ASA CONTROL CHAR
          02  FILESIZE       PIC X(3)       VALUE LOW-VALUE.
 *            FILESIZE < 32GB
          02  SEC-ALLOC      PIC 9(4) COMP  VALUE 0.
          02  RECFORM-FIELD PIC X           VALUE LOW-VALUE.
 *              RECFORM (SET,RESET)
 *                X'04'               S FIXED
 *                X'02'               S VARIABLE
 *                X'06'               S UNDEFINED
          02  BLKSIZE        PIC 9(4) COMP  VALUE 0.
          02  RECSIZE        PIC 9(4) COMP  VALUE 0.
          02  KEYPOS         PIC 9(4) COMP  VALUE 0.
          02  KEYLEN         PIC X          VALUE LOW-VALUE.
          02  PAD            PIC X          VALUE LOW-VALUE.
          02  LOGLEN         PIC X          VALUE LOW-VALUE.
          02  VALLEN         PIC X          VALUE LOW-VALUE.
          02  KEY-EX         PIC X          VALUE SPACE.
 *              KEY EXISTS IN MEMBER ? (YES/NO)
              88  KEY-YES                   VALUE LMSUP-YES.
              88  KEY-NO                    VALUE LMSUP-NO.
          02  CFID           PIC S9(9) COMP VALUE 0.
```

```
    02  BLKCTRL-FIELD PIC X          VALUE LOW-VALUE.
*         BLKCTRL-INDICATOR
*           X'80'               S BLKCTRL=NO
*           X'40'               S BLKCTRL=PAMKEY
*           X'20'               S BLKCTRL=DATA
*           X'10'               S BLKCTRL=NULL
*           X'F0'               R BLKCTRL=NOT SPECIFIED
*           X'08'               S BLOCK CONTROL FIELD 4K
*           X'04'               S BLOCK CONTROL FIELD 2K
*           X'03'               R -- RESERVED, MUST BE 0 --
    02  PMPERF        PIC X          VALUE LOW-VALUE.
*                             IOPERF-INDICATOR
*           X'03'               S IOPERF=VERY-HIGH
*           X'02'               S IOPERF=HIGH
*           X'01'               S IOPERF=STD
*           X'00'               S IOPREF NOT SPECIFIED
    02  PMUSAG        PIC X          VALUE LOW-VALUE.
*                             IOUSAGE-INDICATOR
*           X'03'               S IOUSAGE=RDWRT
*           X'02'               S IOUSAGE=WRITE
*           X'01'               S IOUSAGE=READ
*           X'00'               S IOUSAGE NOT SPECIFIED
    02  PMEDMS3       PIC X          VALUE LOW-VALUE.
*                             CATALOG-INDIC   (IDCEX)
*           X'08'               S PLAM FILE
*
    02  FILLER        PIC X          VALUE LOW-VALUE.
    02  A-IND-COUNT   PIC 9(4) COMP  VALUE 0.
    02  FILESIZE-32   PIC X(4)       VALUE LOW-VALUE.
        >= 32GB, FILESIZE MUST BE X‘FFFFFF‘
    02  FILLER        PIC X(86)      VALUE LOW-VALUE.
*
    02    AIX         OCCURS  30 TIMES.
      03  KEYNAME     PIC X(8)       VALUE LOW-VALUE.
      03  A-KEYPOS    PIC 9(4) COMP  VALUE 0.
      03  A-KEYLEN    PIC X          VALUE LOW-VALUE.
      03  INDICATOR   PIC X          VALUE LOW-VALUE.
*           X'80'               S  DUPKEY=YES
*                               R  DUPKEY=NO
***                                                            ****
**************  END OF COPY ELEMENT   LMSCO164    *******************
```

5.4 Example

The following COBOL program contains the functions listed below:

1. Open a subroutine access (INIT)
2. Incorporate a file as a member (ADD)
3. Search the directory for a member (TOCPRIM)
4. Open a member (OPENGET)
5. Read a member record by record (GET)
6. Close a member (CLOSE)
7. Terminate the subroutine access (END)

To make the example easier to understand, comments have been included.

```
IDENTIFICATION DIVISION.
*****************************************************************
 PROGRAM-ID.       LMSUPCOB.
*****************************************************************

 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SPECIAL-NAMES.
     TERMINAL IS MONITOR
     SYMBOLIC CHARACTERS
              COPY LMSCOBEQ.
     .

*    THE ABOVE PERIOD IS MANDATORY IN THAT IT COMPLETES THE     *
*    SPECIAL-NAMES PARAGRAPH                                    *
/
 DATA DIVISION.
 WORKING-STORAGE SECTION.
*****************************************************************
*****************************************************************
**                                                             **
** THE CONTROL BLOCKS FOR USING LMS AS A SUBROUTINE            **
** ARE STORED AS COPY MEMBERS IN THE LIBRARY                   **
** SYSLIB.LMS.<VERS>.                                          **
*****************************************************************
*****************************************************************

 COPY LMSCOBCB.

 COPY LMSCOBED.

 COPY LMSCOBEI.

 COPY LMSCOBEM.
 COPY LMSCOBFD.

 COPY LMSCOBLD.

 COPY LMSCOBRD.

*****************************************************************
*   TOC IDENTIFICATION                                          *
*****************************************************************
 01 LMSUP-TID                        PIC  9(08) BINARY VALUE 1.

*****************************************************************
*   MEMBER RECORD (FOR TRANSFERRING MEMBERS VIA PUT/GET)        *
*****************************************************************
 01 LMSUP-ER.
    05 SATZKOPF.
       10 SATZLAENGE                 PIC S9(04) BINARY.
       10 FILLER                     PIC  X(02).
    05 SATZPUFFER                    PIC  X(256).

*****************************************************************
*   AUXILIARY FIELDS, CONSTANTS DEFINED FOR THE PROGRAM         *
*****************************************************************
 01 H01-HILFSFELDER.
    05 H01-DATEI-LINKNAME            PIC  X(08) VALUE "FILELINK".
    05 H01-BIBLIOTHEK-LINKNAME       PIC  X(08) VALUE "LIBLINK".
    05 H01-ELEMENT-NAME              PIC  X(54) VALUE "PROBEELEM".
    05 H01-ELEMENT-TYP               PIC  X     VALUE "S".
    05 H01-ELEMENT-VERSION           PIC  X(24) VALUE "1".
    05 H01-PUFFER-LAENGE             PIC  9(09) BINARY VALUE 260.

*****************************************************************
*   OUTPUT AREAS FOR MEMBER RECORD                              *
*****************************************************************
 01 A01-AUSGABE-FELDER.
    05 A01-SATZLAENGE                PIC 9(04) BINARY.
    05 A01-AUSGABE-SATZ.
       10 FILLER                     PIC X     OCCURS 1 TO 256
                                               DEPENDING ON
                                               A01-SATZLAENGE.
/

***************************************************************
 PROCEDURE DIVISION
***************************************************************


 STEUER SECTION.
 ST-ANFANG.
     PERFORM LMS-INITIALISIEREN.
     IF LMSRET-OK
     THEN
            PERFORM LMS-AUFNEHMEN
            PERFORM LMS-INHALT
            PERFORM LMS-ELEM-BEARBEITEN
            PERFORM LMS-BEENDEN
     END-IF.
 ST-ENDE.
     STOP RUN.

/
 LMS-INITIALISIEREN SECTION.
 LMS-INIT-ANFANG.

*****************************************************************
* PREPARE CONTROL BLOCK FOR INITIALIZATION                      *
*****************************************************************
     MOVE LMSUP-UPINIT TO  FUNC    IN LMSUP-SCB.
     MOVE LMSUP-UNUSED TO  SUBCODE IN LMSUP-SCB.
     CALL "LMSUP1"     USING  LMSUP-SCB.

*****************************************************************
* EVALUATE RETURN CODE                                          *
*****************************************************************
     IF LMSRET-OK
     THEN
          DISPLAY  "INITIALIZATION COMPLETED"
                   UPON MONITOR
     ELSE
          DISPLAY  "ERROR OCCURRED DURING INITIALIZATION"
                   UPON MONITOR
     END-IF.
 LMS-INIT-ENDE.
     EXIT.

/
 LMS-AUFNEHMEN SECTION.
 LMS-AUF-ANFANG.
```

```
*****************************************************************
* THE FILE WITH "DATEI-LINKNAME" IS ENTERED UNDER "ELEMENT-NAME"*
* IN THE LIBRARY WITH "BIBLIOTHEK-LINKNAME".                    *
*                                                               *
* CONTROL BLOCK, FILE DESCRIPTION, LIBRARY DESCRIPTION AND      *
* ELEMENT DESCRIPTION ARE TO BE PREPARED FOR ADDITION OF        *
* A MEMBER.                                                     *
*                                                               *
* (ALL OTHER FIELDS SAME AS FOR INIT)                           *
*****************************************************************

     MOVE LMSUP-ADD               TO FUNC       IN LMSUP-SCB.
     MOVE LMSUP-UNUSED            TO SUBCODE    IN LMSUP-SCB.
     MOVE LMSUP-YES               TO OVERWRITE  IN LMSUP-SCB.
     MOVE H01-DATEI-LINKNAME      TO LINK       IN LMSUP-FD.
     MOVE H01-BIBLIOTHEK-LINKNAME TO LINK       IN LMSUP-LD.
     MOVE H01-ELEMENT-TYP         TO TYP        IN LMSUP-ED.
     MOVE H01-ELEMENT-NAME        TO NAME       IN LMSUP-ED.
     MOVE H01-ELEMENT-VERSION     TO VERSION    IN LMSUP-ED.
     CALL "LMSUP1"  USING         LMSUP-SCB,
                                  LMSUP-FD,
                                  LMSUP-LD,
                                  LMSUP-ED.

*****************************************************************
* EVALUATE RETURN CODE                                          *
*****************************************************************

     IF LMSRET-OK
     THEN
         DISPLAY "MEMBER " H01-ELEMENT-NAME
                 " ADDED"
                 UPON MONITOR
     ELSE
         DISPLAY "ERROR DURING ADDITION OF A MEMBER"
                 UPON MONITOR
     END-IF.
 LMS-AUF-ENDE.
     EXIT.

/
 LMS-INHALT SECTION.
 LMS-INHALT-ANFANG.
```

```
*****************************************************************
* SEARCH FOR A PARTICULAR MEMBER WITH "ELEMENT-NAME"            *
* AND "ELEMENT-TYP".                                            *
*                                                               *
* CONTROL BLOCK AND ELEMENT MASK ARE TO BE PREPARED FOR         *
* A SEARCH FOR A MEMBER (TOCPRIM).                              *
* (LIBRARY DEFINITION AS PREDEFINED)                            *
*****************************************************************

     MOVE LMSUP-TOCPRIM        TO FUNC       IN LMSUP-SCB.
     MOVE LMSUP-LONG           TO SUBCODE    IN LMSUP-SCB.
     MOVE H01-ELEMENT-TYP      TO TYP        IN LMSUP-EM.
     MOVE H01-ELEMENT-NAME     TO NAME       IN LMSUP-EM.
     CALL "LMSUP1"  USING      LMSUP-SCB,
                               LMSUP-TID,
                               LMSUP-EI,
                               LMSUP-LD,
                               LMSUP-EM.

*****************************************************************
* EVALUATE RETURN CODE,                                         *
* OUTPUT ELEMENT INFORMATION IN EDITED FORM.                    *
*****************************************************************

     IF LMSRET-OK
     THEN
          DISPLAY "SEARCH FOR MEMBER PERFORMED: "
                  UPON MONITOR
          DISPLAY "TYP       ",  TYP           IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "NAME      ",  NAME          IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "VERSION   ",  VERSION       IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "FORMAT    ",  STORE-FORM    IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "USER-DATE ",  USER-DATE     IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "USER-TIME ",  USER-TIME     IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "CR-DATE   ",  CREATION-DATE IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "CR-TIME   ",  CREATION-TIME IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "MOD-DATE  ",  MODIFI-DATE   IN LMSUP-EI
                  UPON MONITOR
          DISPLAY "MOD-TIME  ",  MODIFI-TIME   IN LMSUP-EI
                  UPON MONITOR
```

```
                DISPLAY "SEC-NAME  ",  SEC-NAME      IN LMSUP-EI
                        UPON MONITOR
                DISPLAY "SEC-ATTR  ",  SEC-ATTRIBUTE IN LMSUP-EI
                        UPON MONITOR
           ELSE
                DISPLAY "ERROR DURING SEARCH FOR A MEMBER"
                        UPON MONITOR
           END-IF.
       LMS-INHALT-ENDE.
           EXIT.


      /
       LMS-ELEM-BEARBEITEN SECTION.
       LMS-ELEM-BEA-ANFANG.

      *****************************************************************
      * A MEMBER IS OPENED FOR PROCESSING, READ RECORD-BY-RECORD      *
      * AND THEN CLOSED AGAIN                                         *
      *****************************************************************
      *****************************************************************
      * CONTROL BLOCK AND ELEMENT DESCRIPTION ARE TO BE PREPARED FOR  *
      * OPENING A MEMBER.                                             *
      * (LIBRARY DESCRIPTION SAME AS BEFORE, BUT MUST BE SUPPLIED.    *
      *  THE MEMBER MUST BE UNIQUELY DESCRIBED BY THE THREE           *
      *  ATTRIBUTES: TYPE, NAME AND VERSION)                          *
      *****************************************************************
           MOVE LMSUP-OPEN-GET       TO FUNC      IN LMSUP-SCB.
           MOVE LMSUP-UNUSED         TO SUBCODE   IN LMSUP-SCB.
           MOVE H01-ELEMENT-NAME     TO NAME      IN LMSUP-ED.
           MOVE H01-ELEMENT-TYP      TO TYP       IN LMSUP-ED.
           MOVE H01-ELEMENT-VERSION  TO VERSION   IN LMSUP-ED.
           CALL "LMSUP1" USING       LMSUP-SCB,
                                     LMSUP-RD,
                                     LMSUP-LD,
                                     LMSUP-ED.
```

```
*****************************************************************
* EVALUATE RETURN CODE;                                         *
* IF NO ERROR HAS OCCURRED, READING STARTS                      *
*****************************************************************
     IF LMSRET-OK
     THEN

*****************************************************************
* CONTROL BLOCK AND RECORD DESCRIPTION ARE TO BE PREPARED FOR   *
* RECORD-BY-RECORD READING OF A MEMBER.                         *
* (PUFFER-LAENGE SPECIFIES THE MAXIMUM EXPECTED RECORD          *
*  LENGTH)                                                      *
*****************************************************************
        MOVE LMSUP-GET         TO FUNC        IN LMSUP-SCB
        MOVE LMSUP-SEQ         TO SUBCODE     IN LMSUP-SCB
        MOVE H01-PUFFER-LAENGE TO BUFFER-LEN  IN LMSUP-RD
*****************************************************************
* READING LOOP UNTIL END OF MEMBER IS REACHED                   *
*****************************************************************
        PERFORM UNTIL NOT LMSRET-OK
           CALL "LMSUP1"  USING  LMSUP-SCB,
                                 LMSUP-RD,
                                 LMSUP-ER

*****************************************************************
* EVALUATE RETURN CODE                                          *
*****************************************************************
           EVALUATE TRUE
           WHEN LMSRET-OK
                SUBTRACT  4  FROM  SATZLAENGE  IN  LMSUP-ER
                          GIVING A01-SATZLAENGE
                MOVE      SATZPUFFER    TO    A01-AUSGABE-SATZ
                DISPLAY   A01-AUSGABE-SATZ    UPON MONITOR
           WHEN LMSRET-EOF
                DISPLAY  "END OF MEMBER REACHED"
                         UPON MONITOR
           WHEN LMSRET-TRUNC
                DISPLAY  "RECORD BUFFER NOT SUFFICIENT"
                         UPON MONITOR
           WHEN OTHER
                DISPLAY  "ERROR DURING READING OF A RECORD"
                         UPON MONITOR
           END-EVALUATE
        END-PERFORM
```

```
*****************************************************************
* CONTROL BLOCK AND RECORD DESCRIPTION ARE TO BE PREPARED       *
* FOR CLOSING A MEMBER                                          *
*****************************************************************
        MOVE LMSUP-CLOSE   TO FUNC    IN LMSUP-SCB
        MOVE LMSUP-UNUSED  TO SUBCODE IN LMSUP-SCB
        CALL "LMSUP1"      USING  LMSUP-SCB,
                                  LMSUP-RD

*****************************************************************
* EVALUATE RETURN CODE                                          *
*****************************************************************
        IF LMSRET-OK
        THEN
                DISPLAY "MEMBER " H01-ELEMENT-NAME
                        " CLOSED"
                        UPON MONITOR
        ELSE
                DISPLAY "ERROR DURING CLOSING OF A MEMBER"
                        UPON MONITOR
        END-IF
     ELSE
            DISPLAY "ERROR DURING OPENING OF A MEMBER"
                    UPON MONITOR
     END-IF.
 LMS-ELEM-BEA-ENDE.
     EXIT.

/
 LMS-BEENDEN SECTION.
 LMS-BEENDEN-ANFANG.

*****************************************************************
* PREPARE CONTROL BLOCK FOR LMS TERMINATION                     *
*****************************************************************
     MOVE LMSUP-UPEND  TO FUNC    IN LMSUP-SCB.
     MOVE LMSUP-UNUSED TO SUBCODE IN LMSUP-SCB.
     CALL "LMSUP1" USING LMSUP-SCB.
```

```
***************************************************************
* EVALUATE RETURN CODE                                        *
***************************************************************
     IF LMSRET-OK
     THEN
          DISPLAY "LMS TERMINATED"
                  UPON MONITOR
     ELSE
          DISPLAY "ERROR DURING LMS TERMINATION"
                  UPON MONITOR
     END-IF.
 LMS-BEENDEN-ENDE.
     EXIT.
```

6 C interface

6.1 Linkage module LMSUP1

LMS functions are called from C programs via the linkage module LMSUP1, which is invoked as follows:

```
LMSUP1 (parameter-list)
```

For the entries in the parameter list see the description of the various functions (page 17ff). Module LMSUP1 is fetched from library SYSLNK.LMS.034 and linked to the main program.

6.2 Include elements for the C/C++ compiler

In order for the C parameter structures to be generated, library SYSLIB.LMS.034 (containing the INCLUDE member LMS.H) must first be assigned. Assignment is possible using the following C/C++ compiler statement:

```
//MODIFY-INCLUDE-LIBRARIES  -
//            STD-INCLUDE-LIBRARY = (…, $.SYSLIB.LMS.034)
```

The installation location of this library is freely selectable via IMON.

The installation location of SYSLNK.LMS.034 can be determined and stored into an S variable using the builtin function INSTALLATION-PATH:

```
/SET-VARIABLE LIBRARY-NAME =INSTALLATION-PATH             -
                            (LOGICAL-ID = ´SYSLIB´        -
                            ,INSTALLATION-UNIT = ´LMS´    -
                            ,VERSION = ´3.4´              -
                            ,DEFAULT-PATH-NAME = ´$.SYSLIB.LMS.034´)
```

Then the library can be assigned as follows:

```
/START-CPLUS-COMPILER
//MODIFY-INCLUDE-LIBRARIES  -
//               STD-INCLUDE-LIBRARY = (…, &(LIBRARY-NAME))
```

Include element LMS.H

```
/* LMS.H       @>V3.3000 1997-11-06<@ */

#ifdef WAS_LMS
#else

#ifdef __cplusplus
extern "C" {
#endif

/*********************************************************************/
/*                          L M S U P                           ****/
/*********************************************************************/



#define WAS_LMS
#define BLANK4  ' ',' ',' ',' '
#define BLANK8  ' ',' ',' ',' ',' ',' ',' ',' '
#define BLANK18 BLANK8,BLANK8,' ',' '
#define BLANK24 BLANK8,BLANK8,BLANK8
#define BLANK64 BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8
#define ZERO4   {'\0','\0','\0','\0'}
#define X0_8    '\0','\0','\0','\0','\0','\0','\0','\0'
#define X0_32   X0_8,X0_8,X0_8,X0_8

/*********************************************************************/
/*                  Function Control Block                      ****/
/*********************************************************************/
struct lmsc_cb
{
       char  scbvers[2];        /* interface-version              in */
       char  function;          /* function-code                  in */
       char  subcode;           /* function-subcode               in */
       int   acc;               /* up access-identification    inout */
       char  retcode;           /* returncode                    out */
       char  filler1;           /* reserved                          */
       short lms_msg;           /* lms message-code              out */
       short dms_msg;           /* dms-message-code              out */
       short plam_msg;          /* plam-message-code             out */
       char  lmsvers[12];       /* version of lms                out */
       int   filler2;          /* reserved                           */
       /********  lms  parameters   ********************************/
       char  destroy;          /* element-destroy-option      inout */
       char  fcb;              /* fcbtype for file            inout */
       char  rkey;             /* key-handling                inout */
       char  overwrite;        /* overwrite-option            inout */
       short column;           /* lms parameter: column       inout */
       short line;             /* lms parameter: line         inout */
       char  prot_ind;         /* lms par: protection ind.    inout */
       char  attr;             /* copylib with attributes     inout */
       char  info;             /* record types for ....       inout */
       char  ld_return;        /* DMS-filename in ld          inout */
       char  filler3[4];       /* reserved                           */
};
#define LMSC_CB_PROTO \
{\
       {'\xF0','\xF4'},                         /* scbvers           */\
       '\x01',                                  /* function          */\
       ' ',                                     /* subcode           */\
       -1,                                      /* acc               */\
       '\0',                                    /* retcode           */\
       '\0',                                    /* filler1           */\
       0,                                       /* lms_msg           */\
       0,                                       /* dms_msg           */\
       0,                                       /* plam_msg          */\
       {BLANK8,BLANK4},                         /* lmsvers           */\
       0,                                       /* filler2           */\
       ' ',                                     /* destroy           */\
       ' ',                                     /* fcbtype           */\
       ' ',                                     /* rkey              */\
       ' ',                                     /* overwrite         */\
       0,                                       /* column            */\
       0,                                       /* line              */\
       ' ',                                     /* prot_ind          */\
       ' ',                                     /* attr              */\
       ' ',                                     /* rec_range         */\
       ' ',                                     /* ld_return         */\
       {BLANK4}                                 /* filler3           */\
}
static struct lmsc_cb lmsc_cb_proto = LMSC_CB_PROTO;


/*******************************************************************/
/*                     Element Attributes                      ****/
/*******************************************************************/
struct lmsc_ea
{
       char  user_date[14];    /* date from user                 in */
       char  user_time[8];     /* time from user hh:mm:ss        in */
       char  ccs_name[8];      /* ccs  from user                 in */
                               /*   V03.00  */
       char  hold_state;       /* hold flag: '-': free           in */
                               /*            'H': in hold           */
                               /*            ' ': unchange          */
       char  filler1[8];       /* reserved (holder userid)       in */
       char  mod_date_ind;     /* modification date indicator    in */
                               /*            'O': old               */
                               /*            'S': system date       */
       char  filler2[56];      /* reserved                       in */
};
#define LMSC_EA_PROTO \
{\
       {BLANK8,BLANK4,' ',' '},                /* user-date         */\
       {BLANK8},                               /* user-time         */\
       {BLANK8},                               /* ccs-name          */\
       ' ',                                    /* hold-state        */\
       {BLANK8},                               /* reserved (holder) */\
       'O',                                    /* mod-dat-ind       */\
       {BLANK24,BLANK24,BLANK8}                /* reserved          */\
}
static struct lmsc_ea lmsc_ea_proto = LMSC_EA_PROTO;



/*******************************************************************/
/*                    Element Description                      ****/
/*******************************************************************/
struct lmsc_ed
{
       char  typ[8];           /* typ of element                 in */
       char  name[64];         /* name of element                in */
       char  version[24];      /* version fo element             in */
       char  store_form;       /* storage form                   in */
       char  user_date[14];    /* date from user                 in */
       char  user_time[8];     /* time from user hh:mm:ss        in */
};
#define LMSC_ED_PROTO \
{\
       {BLANK8},                               /* typ               */\
       {BLANK64},                              /* name              */\
       {BLANK8,BLANK8,BLANK8},                 /* version           */\
       'V',                                    /* stroage-form      */\
       {BLANK8,BLANK4,' ',' '},                /* user-date         */\
       {BLANK8}                                /* user-time         */\
}
```

```
static struct lmsc_ed lmsc_ed_proto = LMSC_ED_PROTO;



/*******************************************************************/
/*                    Element Information                     ****/
/*******************************************************************/
struct lmsc_ei
{
       char  typ[8];           /* type of element                out */
       char  name[64];         /* name of element                out */
       char  version[24];      /* version fo element             out */
       char  store_form;       /* storgae form                   out */
       char  user_date[14];    /* date from user                 out */
       char  user_time[8];     /* time from user                 out */
       char  crea_date[14];    /* creation date of element       out */
       char  crea_time[8];     /* creation time                  out */
       char  modif_date[14];   /* date of last modfication       out */
       char  modif_time[8];    /* time of last modifcation       out */
       char  sec_name[32];     /* secondary name                 out */
       char  sec_attr[8];      /* attribute of secondary name    out */
       char  filler1[5];       /* reserved                           */
       /*********    protection attributes   ***********************/
       char  p_tind_read;      /*  prot. type indi. for read     out */
       char  p_read_own;       /*  read indicator for owner      out */
       char  p_read_grp;       /*  read indicator for group      out */
       char  p_read_oth;       /*  read indicator for others     out */
       char  p_read_pind;      /*  read password indicator       out */
       char  filler2[4];       /*  reserved                          */
       char  p_tind_writ;      /*  prot. type indi. for write    out */
       char  p_writ_own;       /*  write indicator for owner     out */
       char  p_writ_grp;       /*  write indicator for group     out */
       char  p_writ_oth;       /*  write indicator for others    out */
       char  p_writ_pind;      /*  write password indicator      out */
       char  filler3[4];       /*  reserved                          */
       char  p_tind_exec;      /*  prot. type indi. for exec     out */
       char  p_exec_own;       /*  exec indicator for owner      out */
       char  p_exec_grp;       /*  exec indicator for group      out */
       char  p_exec_oth;       /*  exec indicator for others     out */
       char  p_exec_pind;      /*  exec password indicator       out */
       char  filler4[4];       /*  reserved                          */
       char  p_read_guard[18]; /*  read guard                    out */
       char  p_writ_guard[18]; /*  write guard                   out */
       char  p_exec_guard[18]; /*  exec guard                    out */
       char  ccs_name[8];      /*  ccs-name                      out */
                               /*   V03.00:  source code control     */
       char  p_tind_hold;      /*  prot. type indi. for holder   out */
       char  p_hold_own;       /*  holder indicator for owner    out */
```

```
        char  p_hold_grp;       /*  holder indicator for group    out */
        char  p_hold_oth;       /*  holder indicator for others   out */
        char  p_hold_pind;      /*  holder password indicator     out */
        char  filler5[4];       /*  reserved                          */
        char  p_hold_guard[18]; /*  holder guard                  out */
                                /*                                    */
        char  hold_state;       /* hold flag: '-': free           out */
                                /*            'H': in hold            */
        char  holder[8];        /* holder userid                  out */
        char  access_date[14];  /* access date                    out */
        char  access_time[ 8];  /* access time                    out */
        char  filler8[ 1];      /*  reserved                          */
        unsigned int element_size;/*element size                  out */
        char  filler9[40];      /*  reserved                          */
};
#define LMSC_EI_PROTO \
{\
        {BLANK8},                                /* typ               */\
        {BLANK64},                               /* name              */\
        {BLANK8,BLANK8,BLANK8},                  /* version           */\
        ' ',                                     /* storage form      */\
        {BLANK8,BLANK4,' ',' '},                 /* user date         */\
        {BLANK8},                                /* user time         */\
        {BLANK8,BLANK4,' ',' '},                 /* creation date     */\
        {BLANK8},                                /* creation time     */\
        {BLANK8,BLANK4,' ',' '},                 /* modification date */\
        {BLANK8},                                /* modification time */\
        {BLANK8,BLANK8,BLANK8,BLANK8},           /* secondary name    */\
        {BLANK8},                                /* secondary attr    */\
        {BLANK4,' '},                            /* reserved          */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for read     */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for write    */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for exec     */\
        {BLANK18},                               /* read guard        */\
        {BLANK18},                               /* write guard       */\
        {BLANK18},                               /* exec guard        */\
        {BLANK8},                                /* ccs-name          */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for holder   */\
        {BLANK18},                               /* holder guard      */\
        ' ',                                     /* hold-state        */\
        {BLANK8},                                /* holder            */\
        {BLANK8,BLANK4,' ',' '},                 /* access date       */\
        {BLANK8},                                /* access time       */\
        ' ',                                     /* reserved          */\
        0,                                       /* element size      */\
        {BLANK18,BLANK18,BLANK4}                 /* reserved          */\
}
static struct lmsc_ei lmsc_ei_proto = LMSC_EI_PROTO;
```

```
/*********************************************************************/
/*                    Element Mask                               ****/
/*********************************************************************/
struct lmsc_em
{
       char  typ[20];          /* type of element                   in */
       char  name[132];        /* name of element                   in */
       char  version[52];      /* version of element                in */
       char  store_form[6];    /* storage form                      in */
       char  user_date[32];    /* date from user                    in */
       char  user_time[20];    /* time from user hh:mm:ss           in */
       char  crea_date[32];    /* creation date of element          in */
       char  crea_time[20];    /* creation time of element          in */
       char  modif_date[32];   /* date of last modification         in */
       char  modif_time[20];   /* time of last modification         in */
       char  sec_name[68];     /* secondary name                    in */
       char  sec_attr[20];     /* attribute of secondary name       in */
       char  filler1[14];      /* reserved                             */
       /*********    protection attributes   ************************/
       char  p_tind_read;      /*  prot. type indi. for read        in */
       char  p_read_own;       /*  read indicator for owner         in */
       char  p_read_grp;       /*  read indicator for group         in */
       char  p_read_oth;       /*  read indicator for others        in */
       char  p_read_pind;      /*  read password indicator          in */
       char  filler2[4];       /*  reserved                            */
       char  p_tind_writ;      /*  prot. type indi. for write       in */
       char  p_writ_own;       /*  write indicator for owner        in */
       char  p_writ_grp;       /*  write indicator for group        in */
       char  p_writ_oth;       /*  write indicator for others       in */
       char  p_writ_pind;      /*  write password indicator         in */
       char  filler3[4];       /*  reserved                            */
       char  p_tind_exec;      /*  prot. type indi. for exec        in */
       char  p_exec_own;       /*  exec indicator for owner         in */
       char  p_exec_grp;       /*  exec indicator for group         in */
       char  p_exec_oth;       /*  exec indicator for others        in */
       char  p_exec_pind;      /*  exec password indicator          in */
       char  filler4[4];       /*  reserved                            */
       char  p_read_guard[40]; /*  read guard                       in */
       char  p_writ_guard[40]; /*  write guard                      in */
       char  p_exec_guard[40]; /*  exec guard                       in */
       char  ccs_name[20];     /*  ccs-name                         in */
                               /*   V03.00:  source code control       */
       char  p_tind_hold;      /*  prot. type indi. for holder      in */
       char  p_hold_own;       /*  holder indicator for owner       in */
       char  p_hold_grp;       /*  holder indicator for group       in */
```

```
        char  p_hold_oth;       /*  holder indicator for others    in */
        char  p_hold_pind;      /*  holder password indicator      in */
        char  filler5[4];       /*  reserved                          */
        char  p_hold_guard[40]; /*  holder guard                   in */
                                /*                                    */
        char  hold_state;       /* hold flag: '-': free            in */
                                /*            'H': inhold             */
                                /*            ' ': any                */
        char  holder[20];       /* holder userid                   in */
        char  access_date[32];  /* access date                     in */
        char  access_time[20];  /* access time                     in */
        char  filler8[ 3];      /* reserved                           */
        unsigned int e_size_min;/* min element size                in */
        unsigned int e_size_max;/* max element size                in */
        char  filler9[64];      /* reserved                           */
};
#define LMSC_EM_PROTO \
{\
        {BLANK8,BLANK8,BLANK4},                   /* type              */\
        {BLANK64,BLANK64,BLANK4},                 /* name              */\
        {BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK4}, /* vers   */\
        {BLANK4,' ',' '},                         /* storage form      */\
        {BLANK8,BLANK8,BLANK8,BLANK8},            /* user date         */\
        {BLANK8,BLANK8,BLANK4},                   /* user time         */\
        {BLANK8,BLANK8,BLANK8,BLANK8},            /* creation date     */\
        {BLANK8,BLANK8,BLANK4},                   /* creation time     */\
        {BLANK8,BLANK8,BLANK8,BLANK8},            /* modification date */\
        {BLANK8,BLANK8,BLANK4},                   /* modification time */\
        {BLANK64,BLANK4},                         /* secondary name    */\
        {BLANK8,BLANK8,BLANK4},                   /* secondary attr    */\
        {BLANK8,BLANK4,' ',' '},                  /* reserved          */\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for read     */\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for write    */\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for exec     */\
        {BLANK18,BLANK18,BLANK4},                 /* read guard        */\
        {BLANK18,BLANK18,BLANK4},                 /* write guard       */\
        {BLANK18,BLANK18,BLANK4},                 /* exec guard        */\
        {BLANK8,BLANK8,BLANK4},                   /* ccs-name          */\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for holder   */\
        {BLANK18,BLANK18,BLANK4},                 /* holder guard      */\
        ' ',                                      /* hold-state        */\
        {BLANK18,' ',' '},                        /* holder            */\
        {BLANK8,BLANK8,BLANK8,BLANK8},            /* access date       */\
        {BLANK8,BLANK8,BLANK4},                   /* access time       */\
        {' ',' ',' '},                            /* reserved          */\
        0,                                        /* min element size  */\
        0xFFFFFFFF,                               /* max element size  */\
        {BLANK64}                                 /* reserved          */\
}
static struct lmsc_em lmsc_em_proto = LMSC_EM_PROTO;


/*******************************************************************/
/*                      File Description                       ****/
/*******************************************************************/
struct lmsc_fd
{
      char  password[4];       /* password                      in */
      char  link[8];           /* linkname for file             in */
      char  name[54];          /* filename                   inout */
};
#define LMSC_FD_PROTO \
{\
      ZERO4,                                  /* password          */\
      {BLANK8},                               /* linkname          */\
      {BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK4,' ',' '}  \
}
static struct lmsc_fd lmsc_fd_proto = LMSC_FD_PROTO;



/*******************************************************************/
/*                    Library Attributes                       ****/
/*******************************************************************/
struct lmsc_la
{
      char  p_tind_admin;     /*  prot. type indi. for admin     in */
      char  p_admin_own;      /*  admin indicator for owner      in */
      char  p_admin_grp;      /*  admin indicator for group      in */
      char  p_admin_oth;      /*  admin indicator for others     in */
      char  p_admin_pind;     /*  admin password indicator       in */
      char  p_admin_pswd[4];  /*  admin password                 in */
      char  p_admin_guard[18]; /*  admin guard                   in */
                              /*   V03.00  */
      char  store_form;       /*  ind. for holder-authorization  in */
                              /*  'S': STD   fully or delta stored  */
                              /*  'V': FULL  all el. fully stored   */
                              /*  'D': DELTA all el. fully stored   */
      char  write_ctrl;       /*  write-control for library      in */
                              /*  'A': activated                    */
                              /*  'D': deactivated                  */
      char  access_date;      /*  access date                    in */
                              /*  'N': NONE (no KEEP)               */
                              /*  'K': KEEP                         */
      char  filler2[34];      /*  reserved                          */
};
```

```
#define LMSC_LA_PROTO \
{\
       ' ',' ',' ',' ',' ',ZERO4,                /* prot for Adm   in */\
       {BLANK18},                                /* admin guard       */\
       ' ',' ',' ',                              /* store-form/wr-ctrl*/\
       {BLANK24,BLANK8,' ',' '}                  /* reserved          */\
}
static struct lmsc_la lmsc_la_proto = LMSC_LA_PROTO;



/*********************************************************************/
/*                     Library Description                       ****/
/*********************************************************************/
struct lmsc_ld
{
       char  password[4];       /* password for library           in */
       char  link[8];           /* linkname                       in */
       char  reserv1[8];        /*                                   */
       short max_name_len;      /* max.length of libary_name      in */
       char  name[54];          /* name of libary              inout */
};
#define LMSC_LD_PROTO \
{\
       ZERO4,                                    /* password          */\
       {BLANK8},                                 /* linkname          */\
       {'\0','\0','\0','\0','\0','\0','\0','\0'},/* reserv1           */\
       (short)54,                                /* max length name   */\
       {BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK4,' ',' '}  \
}
static struct lmsc_ld lmsc_ld_proto = LMSC_LD_PROTO;



/*********************************************************************/
/*                     Library Information                       ****/
/*********************************************************************/
struct lmsc_li
{
       char  p_tind_admin;     /*  prot. type indi. for admin    out */
       char  p_admin_own;      /*  admin indicator for owner     out */
       char  p_admin_grp;      /*  admin indicator for group     out */
       char  p_admin_oth;      /*  admin indicator for others    out */
       char  p_admin_pind;     /*  admin password indicator      out */
       char  filler1[4];       /*  reserved                          */
       char  p_admin_guard[18]; /*  admin guard                      */
       char  store_form;       /*  ind. for holder-authorization out */
                               /*  'S': STD   fully or delta stored  */
```

```
                          /*  'V': FULL  all el. fully stored   */
                          /*  'D': DELTA all el. fully stored   */
char  write_ctrl;         /*  write-control for library     out */
                          /*  'A': activated                    */
                          /*  'D': deactivated                  */
char  access_date;        /*  access date                   out */
                          /*  'N': NONE (no KEEP)               */
                          /*  'K': KEEP                         */
char  filler2[24];        /* reserved                           */
char  lib_form;           /*  library format (NK2/NK4)      out */
                          /*    '2' : NK2 library format        */
                          /*    '4' : NK4 library format        */
char  upam_prot;          /*  UPAM protected (yes/no)       out */
                          /*    'Y' : lib is UPAM protected     */
                          /*    'N' : lib is not UPAM protected */
int   file_size;          /*  file size                     out */
int   free_size;          /*  free size                     out */
/*********    protection attributes   ************************/
char  p_tind_read;        /*  prot. type indi. for read     out */
char  p_read_own;         /*  read indicator for owner      out */
char  p_read_grp;         /*  read indicator for group      out */
char  p_read_oth;         /*  read indicator for others     out */
char  p_read_pind;        /*  read password indicator       out */
char  filler4[4];         /*  reserved                          */
char  p_tind_writ;        /*  prot. type indi. for write    out */
char  p_writ_own;         /*  write indicator for owner     out */
char  p_writ_grp;         /*  write indicator for group     out */
char  p_writ_oth;         /*  write indicator for others    out */
char  p_writ_pind;        /*  write password indicator      out */
char  filler5[4];         /*  reserved                          */
char  p_tind_exec;        /*  prot. type indi. for exec     out */
char  p_exec_own;         /*  exec indicator for owner      out */
char  p_exec_grp;         /*  exec indicator for group      out */
char  p_exec_oth;         /*  exec indicator for others     out */
char  p_exec_pind;        /*  exec password indicator       out */
char  filler6[4];         /*  reserved                          */
char  p_read_guard[18];   /*  read guard                    out */
char  p_writ_guard[18];   /*  write guard                   out */
char  p_exec_guard[18];   /*  exec guard                    out */
                          /*   V03.00  */
char  p_tind_hold;        /*  prot. type indi. for holder   out */
char  p_hold_own;         /*  holder indicator for owner    out */
char  p_hold_grp;         /*  holder indicator for group    out */
char  p_hold_oth;         /*  holder indicator for others   out */
char  p_hold_pind;        /*  holder password indicator     out */
char  filler7[4];         /*  reserved                          */
char  p_hold_guard[18];   /*  holder guard                  out */
char  filler8[68];        /*  reserved                          */
```

```
};
#define LMSC_LI_PROTO \
{\
        ' ',' ',' ',' ',' ',ZERO4,                  /* prot for admin    */\
        {BLANK18},                                  /* admin guard       */\
        ' ',' ',' ',                                /* store-form/wr-ctrl*/\
        {BLANK24},                                  /* reserved          */\
        ' ',' ', 0,  0,                             /* fomat + size      */\
        ' ',' ',' ',' ',' ',ZERO4,                  /* prot for read     */\
        ' ',' ',' ',' ',' ',ZERO4,                  /* prot for write    */\
        ' ',' ',' ',' ',' ',ZERO4,                  /* prot for exec     */\
        {BLANK18},                                  /* read guard        */\
        {BLANK18},                                  /* write guard       */\
        {BLANK18},                                  /* exec guard        */\
        ' ',' ',' ',' ',' ',ZERO4,                  /* holder            */\
        {BLANK18},                                  /* holder guard      */\
        {BLANK64,BLANK4}                            /* reserved          */\
}
static struct lmsc_li lmsc_li_proto = LMSC_LI_PROTO;



/******************************************************************/
/*                    Protection Attributes                   ****/
/******************************************************************/
struct lmsc_pa
{
        char  p_tind_read;      /*  prot. type indi. for read      in */
        char  p_read_own;       /*  read indicator for owner       in */
        char  p_read_grp;       /*  read indicator for group       in */
        char  p_read_oth;       /*  read indicator for others      in */
        char  p_read_pind;      /*  read password indicator        in */
        char  p_read_pswd[4];   /*  read password                  in */
        char  p_tind_writ;      /*  prot. type indi. for write     in */
        char  p_writ_own;       /*  write indicator for owner      in */
        char  p_writ_grp;       /*  write indicator for group      in */
        char  p_writ_oth;       /*  write indicator for others     in */
        char  p_writ_pind;      /*  write password indicator       in */
        char  p_writ_pswd[4];   /*  write password                 in */
        char  p_tind_exec;      /*  prot. type indi. for exec      in */
        char  p_exec_own;       /*  exec indicator for owner       in */
        char  p_exec_grp;       /*  exec indicator for group       in */
        char  p_exec_oth;       /*  exec indicator for others      in */
        char  p_exec_pind;      /*  exec password indicator        in */
        char  p_exec_pswd[4];   /*  exec password                  in */
        char  p_read_guard[18]; /*  read guard                     in */
        char  p_writ_guard[18]; /*  write guard                    in */
        char  p_exec_guard[18]; /*  exec guard                     in */
```

```
                                /*   V03.00  */
        char  p_tind_hold;      /*  prot. type indi. for holder    in */
        char  p_hold_own;       /*  holder indicator for owner     in */
        char  p_hold_grp;       /*  holder indicator for group     in */
        char  p_hold_oth;       /*  holder indicator for others    in */
        char  p_hold_pind;      /*  holder password indicator      in */
        char  p_hold_pswd[4];   /*  holder password                in */
        char  p_hold_guard[18]; /*  holder guard                   in */
        char  filler1[84];      /*  reserved                          */
};
#define LMSC_PA_PROTO \
{\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for read      */\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for write     */\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for exec      */\
        {BLANK18},                                /* read guard         */\
        {BLANK18},                                /* write guard        */\
        {BLANK18},                                /* exec guard         */\
        ' ',' ',' ',' ',' ',ZERO4,                /* prot for holder    */\
        {BLANK18},                                /* holder guard       */\
        {BLANK64,BLANK18,' ',' '}                 /* reserved           */\
}
static struct lmsc_pa lmsc_pa_proto = LMSC_PA_PROTO;



/*******************************************************************/
/*                      Record Description                     ****/
/*******************************************************************/
struct lmsc_rd
{
        int   rec_acc_id;        /* element access id          inout */
        int   buffer_len;        /* length of recordbuffer        in */
        int   record_len;        /* length of record           inout */
        char  filler1[3];        /* reserved                         */
        char  record_type;       /* record type                inout */
        int   record_num;        /* number of record           inout */
        char  filler2[8];        /* reserved                         */
};
#define LMSC_RD_PROTO \
{\
        -1,                                       /* rec_acc_id         */\
        0,                                        /* buffer_len         */\
        0,                                        /* record_len         */\
        {'\0','\0','\0'},                         /* reserv1            */\
        '\x01',                                   /* record_type        */\
        0,                                        /* record_num         */\
        {'\0','\0','\0','\0','\0','\0','\0','\0'} /* reserv2            */\
}
static struct lmsc_rd lmsc_rd_proto = LMSC_RD_PROTO;



/*****************************************************************/
/*                     Type Attributes                      ****/
/*****************************************************************/
struct lmsc_ta
{
      char  convention;        /* type convention              in */
      char  filler1[3];        /* reserved                        */
      char  v_example[24];     /* version example (f. seq)     in */
                              /*   V03.00  */
      char  p_tind_admin;     /*  prot. type indi. for admin    in */
      char  p_admin_own;      /*  admin indicator for owner     in */
      char  p_admin_grp;      /*  admin indicator for group     in */
      char  p_admin_oth;      /*  admin indicator for others    in */
      char  p_admin_pind;     /*  admin password indicator      in */
      char  p_admin_pswd[4];  /*  admin password                in */
      char  p_admin_guard[18]; /*  admin guard                  in */
                              /*   V03.00  */
      char  store_form;       /*  ind. for holder-authorization  in */
                              /*  'S': STD   fully or delta stored  */
                              /*  'V': FULL  all el. fully stored   */
                              /*  'D': DELTA all el. fully stored   */
      char  write_ctrl;       /*  write-control for type        in */
                              /*  'A': activated                   */
                              /*  'D': deactivated                 */
      char  super_type[8];    /*  super type                    in */
      char  filler2[47];      /*  reserved                         */
};
#define LMSC_TA_PROTO \
{\
      ' ',                                    /* convention       */\
      {' ',' ',' '},                          /* reserved         */\
      {BLANK24},                              /* v_example        */\
      ' ',' ',' ',' ',' ',ZERO4,              /* prot for Adm   in */\
      {BLANK18},                              /* admin guard      */\
      ' ',' ',                                /* store-form/wr-ctrl*/\
      {BLANK8},                               /* super type       */\
      {BLANK24,BLANK18,BLANK4,' '}            /* reserved         */\
}
static struct lmsc_ta lmsc_ta_proto = LMSC_TA_PROTO;



/*****************************************************************/
```

```
/*                      Type Description                          ****/
/*********************************************************************/
struct lmsc_td
{
       char  typ[8];            /* element type                   in */
       char  filler1[8];        /* reserved                          */
};
#define LMSC_TD_PROTO \
{\
       {BLANK8},                                 /* typ              */\
       {BLANK8}                                  /* reserved         */\
}
static struct lmsc_td lmsc_td_proto = LMSC_TD_PROTO;



/*********************************************************************/
/*                      Type Information                          ****/
/*********************************************************************/
struct lmsc_ti
{
       char  typ[8];            /* element type                  out */
       char  filler1[8];        /* reserved                          */
       char  convention;        /* type convention               out */
       char  filler2[3];        /* reserved                          */
       char  v_example[24];     /* version example (f. seq)      out */
                               /*   V03.00  */
       char  p_tind_admin;     /*  prot. type indi. for admin    out */
       char  p_admin_own;      /*  admin indicator for owner     out */
       char  p_admin_grp;      /*  admin indicator for group     out */
       char  p_admin_oth;      /*  admin indicator for others    out */
       char  p_admin_pind;     /*  admin password indicator      out */
       char  filler3[4];       /*  reserved                          */
       char  p_admin_guard[18]; /*  admin guard                  out */
                               /*   V03.00  */
       char  store_form;       /*  ind. for holder-authorization out */
                               /*  'S': STD   fully or delta stored  */
                               /*  'V': FULL  all el. fully stored   */
                               /*  'D': DELTA all el. fully stored   */
       char  write_ctrl;       /*  write-control for library     out */
                               /*  'A': activated                    */
                               /*  'D': deactivated                  */
       char  super_type[8];    /*  super type                    out */
       char  basis_type[8];    /*  basis type                    out */
       char  filler5[39];      /*  reserved                          */
       /*********    protection attributes   ************************/
       char  p_tind_read;      /*  prot. type indi. for read     out */
       char  p_read_own;       /*  read indicator for owner      out */
```

```
        char  p_read_grp;        /*  read indicator for group      out */
        char  p_read_oth;        /*  read indicator for others     out */
        char  p_read_pind;       /*  read password indicator       out */
        char  filler6[4];        /*  reserved                          */
        char  p_tind_writ;       /*  prot. type indi. for write    out */
        char  p_writ_own;        /*  write indicator for owner     out */
        char  p_writ_grp;        /*  write indicator for group     out */
        char  p_writ_oth;        /*  write indicator for others    out */
        char  p_writ_pind;       /*  write password indicator      out */
        char  filler7[4];        /*  reserved                          */
        char  p_tind_exec;       /*  prot. type indi. for exec     out */
        char  p_exec_own;        /*  exec indicator for owner      out */
        char  p_exec_grp;        /*  exec indicator for group      out */
        char  p_exec_oth;        /*  exec indicator for others     out */
        char  p_exec_pind;       /*  exec password indicator       out */
        char  filler8[4];        /*  reserved                          */
        char  p_read_guard[18]; /*  read guard                    out */
        char  p_writ_guard[18]; /*  write guard                   out */
        char  p_exec_guard[18]; /*  exec guard                    out */
        char  p_tind_hold;       /*  prot. type indi. for holder   out */
        char  p_hold_own;        /*  holder indicator for owner    out */
        char  p_hold_grp;        /*  holder indicator for group    out */
        char  p_hold_oth;        /*  holder indicator for others   out */
        char  p_hold_pind;       /*  holder password indicator     out */
        char  filler4[4];        /*  reserved                          */
        char  p_hold_guard[18]; /*  holder guard                  out */
        char  filler9[52];       /*  reserved                          */
};
#define LMSC_TI_PROTO \
{\
        {BLANK8},                                /* typ              */\
        {BLANK8},                                /* reserved         */\
        ' ',                                     /* convention       */\
        {' ',' ',' '},                           /* reserved         */\
        {BLANK24},                               /* v_example        */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for Admin   */\
        {BLANK18},                               /* admin guard      */\
        ' ',' ',                                 /* store-form/wr-ctrl*/\
        {BLANK8},                                /* super type       */\
        {BLANK8},                                /* basis type       */\
        {BLANK18,BLANK18,' ',' ',' '},           /* reserved         */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for read    */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for write   */\
        ' ',' ',' ',' ',' ',ZERO4,               /* prot for exec    */\
        {BLANK18},                               /* read guard       */\
        {BLANK18},                               /* write guard      */\
        {BLANK18},                               /* exec guard       */\
        ' ',' ',' ',' ',' ',ZERO4,               /* holder           */\
       {BLANK18},                                /* holder guard      */\
       {BLANK24,BLANK24,BLANK4}                  /* reserved          */\
}
static struct lmsc_ti lmsc_ti_proto = LMSC_TI_PROTO;



/* -------------- LMS Parameter values ------------------------------*/
#define LMSUP_YES            'Y'   /* yes                             */
#define LMSUP_NO             'N'   /* no                              */
#define LMSUP_NONE           'N'   /* none                            */
#define LMSUP_ANY            ' '   /* any                             */
#define LMSUP_UNCHANGE       ' '   /* unchange                        */
#define LMSUP_SAME           'M'   /* same                            */
#define LMSUP_STD            'S'   /* std                             */
/* -------------- FOR: CBFCB  ---------------------------------------*/
#define LMSUP_ISAM           'I'   /* isam                            */
#define LMSUP_SAM            'Q'   /* sam                             */
#define LMSUP_CATALOG        'C'   /* catalog                         */
/* -------------- FOR: CBOV -----------------------------------------*/
#define LMSUP_EXTEND         'E'   /* extend                          */
#define LMSUP_ONLY           'O'   /* only                            */
#define LMSUP_NAME           'A'   /* name                            */
/* -------------- FOR: INFO      ------------------------------------*/
#define LMSUP_TXT          '\x01'  /* text only                       */
#define LMSUP_COM          '\x02'  /* coment / documentation only     */
/* -------------- functioncodes -------------------------------------*/
#define LMSUP_INIT         '\x01'  /* init                            */
#define LMSUP_END          '\x02'  /* end                             */
#define LMSUP_TOCPRIM      '\x03'  /* tocprim                         */
#define LMSUP_TOCSEC       '\x04'  /* tocsec                          */
#define LMSUP_TOC          '\x05'  /* toc                             */
#define LMSUP_REN          '\x06'  /* ren                             */
#define LMSUP_DEL          '\x07'  /* ddel                            */
#define LMSUP_ADD          '\x08'  /* add                             */
#define LMSUP_SEL          '\x09'  /* sel                             */
#define LMSUP_COPY         '\x0A'  /* copy                            */
#define LMSUP_COPSTRUC     '\x0B'  /* copystr                         */
#define LMSUP_LOCK         '\x0C'  /* lock                            */
#define LMSUP_UNLOCK       '\x0D'  /* unlock                          */
#define LMSUP_OPEN_GET     '\x0E'  /* open get                        */
#define LMSUP_OPEN_PUT     '\x0F'  /* open put                        */
#define LMSUP_OPEN_UPD     '\x10'  /* open upd                        */
#define LMSUP_GET          '\x11'  /* get                             */
#define LMSUP_PUT          '\x12'  /* put                             */
#define LMSUP_CLOSE        '\x13'  /* close                           */
#define LMSUP_LIST         '\x14'  /* list element                    */
#define LMSUP_MOD_EL_PROT  '\x15'  /* modify element protection      */
```

```
#define LMSUP_MOD_LIB_A    '\x16'  /* modify library attributes      */
#define LMSUP_SHOW_LIB_A   '\x17'  /* show   library attributes      */
#define LMSUP_MOD_TYP_A    '\x18'  /* modify type attributes         */
#define LMSUP_SHOW_TYP_A   '\x19'  /* show   type attributes         */
#define LMSUP_MOD_EA       '\x1A'  /* modify element attributes      */
#define LMSUP_COPY_LIB     '\x1B'  /* copy  library                  */
#define LMSUP_CLOSE_LIB    '\x1C'  /* close library                  */
#define LMSUP_PROVIDE      '\x1D'  /* provide element                */
#define LMSUP_RETURN       '\x1E'  /* return element                 */
#define LMSUP_GET_SYS_EL   '\x1F'  /* get variable syslmselem        */
#define LMSUP_REORG_LIB    '\x20'  /* reorganize library             */
/* -------------- subcodes ----------------------------------------- */
#define LMSUP_UNUSED         ' '   /* subcode unused (default)       */
#define LMSUP_SHORT          'S'   /* toc short                      */
#define LMSUP_LONG           'L'   /* toc long                       */
#define LMSUP_DIR            'D'   /* direct read                    */
#define LMSUP_SEQ            'S'   /* sequential read                */
#define LMSUP_WRITE          'W'   /* close output element           */
#define LMSUP_RESET          'R'   /* reset output element           */
#define LMSUP_SYM            'S'   /* show element symbolic          */
#define LMSUP_HEX            'H'   /* show element alpha + hex       */
#define LMSUP_INC_PRE        'P'   /* increment with prefix          */
#define LMSUP_INC_BASE       'B'   /* increment with baset           */
#define LMSUP_HIGH_PRE       'H'   /* highest with prefix            */
#define LMSUP_EXTRA          'X'   /* Format-B records allowed       */
/* -------------- returncodes -------------------------------------- */
#define LMSUP_OK           '\0'    /* OK                             */
#define LMSUP_TRUNC        '\x04'  /* REC TRUNC                      */
#define LMSUP_EOF          '\x08'  /* EOF (GET/TOC)                  */
#define LMSUP_LMSERR       '\x0C'  /* LMS ERROR                      */
#define LMSUP_PARERR       '\x14'  /* PARAMETER ERROR                */
#define LMSUP_SEQERR       '\x18'  /* SEQUENCE ERROR                 */
#define LMSUP_INTERR       '\x1C'  /* LMS INTERNAL ERROR             */
/* -------------- storage form ------------------------------------- */
#define LMSUP_FULL           'V'   /* fully stored                   */
#define LMSUP_DELTA          'D'   /* delta stored                   */
/* -------------- conventions  ------------------------------------- */
#define LMSUP_CNONE          'N'   /* none                           */
#define LMSUP_CSEQ           'S'   /* std-sequence                   */
#define LMSUP_CMSEQ          'M'   /* multi-sequence                 */
#define LMSUP_CTREE          'T'   /* std-tree                       */
/* -------------- protection indicators  --------------------------- */
#define LMSUP_PNONE          'N'   /* none                           */
#define LMSUP_PSTD           'Y'   /* standard protection            */
#define LMSUP_PGUARD         'G'   /* protection by guard            */
/* -------------- source code control  ----------------------------- */
#define LMSUP_FREE           '-'   /* free                           */
#define LMSUP_INHOLD         'H'   /* in hold                        */
```

```
/* -------------- write control  ----------------------------------- */
#define LMSUP_ACTIV           'A'   /* active                         */
#define LMSUP_DEACTIV         'D'   /* inactive                       */
/* -------------- access date  ------------------------------------- */
#define LMSUP_KEEP            'K'   /* keep access date               */
/* -------------- modification date indicator ---------------------- */
#define LMSUP_OLD             'O'   /* by-source                      */
#define LMSUP_SYS_DATE        'S'   /* new (system date)              */
/* ----------------------------------------------------------------- */

#ifdef C_V1
extern void lmsup1 ();
#else
extern void lmsup1( struct lmsc_cb *cb, ...);
#endif



#ifdef __cplusplus
}
#endif

#endif
```

Include element LMSREC.H

This element contains the definitions for the construction of special records (record type 163,164, see section “Format of the secondary record (record type 163)” on page 140 and section “Format of the attribute record (record type 164)” on page 140)

```
/* LMSREC.H    @>V3.1000 1995-04-26<@ */

#ifdef WAS_LMSREC
#else

/*******************************************************************/
/*                          L M S R E C                        ****/
/*******************************************************************/

#define WAS_LMSREC
#define BLANK4  ' ',' ',' ',' '
#define BLANK8  ' ',' ',' ',' ',' ',' ',' ',' '
#define BLANK18 BLANK8,BLANK8,' ',' '
#define BLANK24 BLANK8,BLANK8,BLANK8
#define BLANK64 BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8,BLANK8
```

```
#define ZERO4   {'\0','\0','\0','\0'}
#define X0_4    '\0','\0','\0','\0'
#define X0_8    '\0','\0','\0','\0','\0','\0','\0','\0'
#define X0_32   X0_8,X0_8,X0_8,X0_8


/*******************************************************************/
/*                  RECORD TYPE :  1 6 3                       ****/
/*******************************************************************/
struct lmsc_rec163
{
       short rec_len;            /*   record length                  */
       char  filler1;            /*   reserved                       */
       char  rec_type;           /*   identification of plam record  */
       char  sec_name[32];       /*   secondary name                 */
       char  sec_attr[8];        /*   secondary attribute            */
                                 /*   '0       ' :  -  CSECT         */
                                 /*   '1       ' :  -  ENTRY         */
       char  format_ind;         /*    format indicator              */
       char  sec_name_long[1];   /*   long secondary name: - 32K-45  */
};
#define LMSC_REC163_PROTO  \
{\
       0,                        /*   record length                  */\
       '\0',                     /*   reserved                       */\
       163,                      /*   identification of plam record  */\
       {BLANK24,BLANK8},         /*   secondary name                 */\
       {BLANK8},                 /*   secondary attribute            */\
       0,                        /*    format indicator              */\
       ' '                       /*   long secondary name: - 32K-45  */\
}
static struct lmsc_rec163 lmsc_rec163_proto = LMSC_REC163_PROTO;


/*******************************************************************/
/*                  RECORD TYPE :  1 6 4                       ****/
/*******************************************************************/
#define LMSREC_AIXMAX      30    /*   max nr. of aix entries         */
struct lmsc_rec164
{
       short rec_len;            /*   record length                  */
       char  filler1;            /*   reserved                       */
       char  rec_type;           /*   identification of plam record  */
       char  version;            /*   version of specified PLAM record*/
       char  rec_num;            /*   record describes format of
                                      PLAM record of record type
                                      with specified number          */
       char  filename[54];       /*   file name taken from FCB       */
       char  fcb;              /*   fcbtype (SET/RESET)            */
       char  share;            /*   share (SET,RESET)              */
       char  size[3];          /*   file size < 32GB               */
       char  sec_alloc[2];     /*   secondary allocation (byte alig)*/
       char  recform;          /*   recform (SET,RESET)            */
       short blksize;          /*   blksize                        */
       short recsize;          /*   recsize                        */
       short keypos;           /*   keypos                         */
       char  keylen;           /*   keylen                         */
       char  pad;              /*   pad                            */
       char  loglen;           /*   loglen                         */
       char  vallen;           /*   vallen                         */
       char  key_ind;          /*   does key exists in member      */
       char  cfid[4];          /*   cfid                           */
       char  blkctrl_ind;      /*   BLKCTRL-indicator              */
       char  ioperf_ind;       /*   IOPERF-indicator               */
       char  iousage_ind;      /*   IOusage-indicator              */
       char  catalog_ind;      /*   catalog-indicator              */
       char  filler2;          /*   reserved, must be 0            */
       short a_ind_count;      /*   alternate index count          */
       char  filesize_32[4];   /*   file size >= 32GB,
                                    filesize must be X'FFFFFF'     */
       char  filler3[86];      /*   reserved, must be 0            */
       struct
       {      char  keyname[8]; /*   keyname                       */
              short keypos;     /*   keypos                        */
              char  keylen;     /*   keylen                        */
              char  indicator;  /*   dupkey=yes/no                 */
       }     aix[LMSREC_AIXMAX];
};
#define LMSC_AIX_PROTO {X0_8,0,'\0','\0'}
#define LMSC_REC164_PROTO \
{\
       540,                    /*   record length                  */\
       '\0',                   /*   reserved                       */\
       164,                    /*   identification of plam record  */\
       '\2',                   /*   version of specified PLAM record*/\
       '\1',                   /*   record describes format of \
                                    PLAM record of record type \
                                    with specified number          */\
       {BLANK18,BLANK18,BLANK18},  /*  file name taken from FCB    */\
       '\0',                   /*   fcbtype (SET/RESET)            */\
       '\0',                   /*   share (SET,RESET)              */\
       {'\0','\0','\0'},       /*   file size < 32GB               */\
       {'\0','\0'},            /*   secondary allocation           */\
       '\0',                   /*   recform (SET,RESET)            */\
       0,                      /*   blksize                        */\
       0,                      /*   recsize                        */\
       0,                         /*   keypos                          */\
       '\0',                      /*   keylen                          */\
       '\0',                      /*   pad                             */\
       '\0',                      /*   loglen                          */\
       '\0',                      /*   vallen                          */\
       ' ',                       /*   does key exists in member       */\
       ZERO4,                     /*   cfid                            */\
       '\0',                      /*   BLKCTRL-indicator               */\
       '\0',                      /*   IOPERF-indicator                */\
       '\0',                      /*   IOusage-indicator               */\
       '\0',                      /*   catalog-indicator               */\
       '\0',                      /*   reserved, must be 0             */\
       0,                         /*   alternate index count           */\
       {'\0','\0','\0','\0'},     /*   file size >= 32GB               */\
       {X0_32,X0_32,X0_8,X0_8,    /*   reserved, must be 0             */\
        X0_4,'\0','\0'},\
       {LMSC_AIX_PROTO,LMSC_AIX_PROTO,LMSC_AIX_PROTO,   \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,                  \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,LMSC_AIX_PROTO,   \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,                  \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,LMSC_AIX_PROTO,   \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,                  \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,LMSC_AIX_PROTO,   \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,                  \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,LMSC_AIX_PROTO,   \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,                  \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO,LMSC_AIX_PROTO,   \
        LMSC_AIX_PROTO,LMSC_AIX_PROTO}                  \
}
static struct lmsc_rec164 lmsc_rec164_proto = LMSC_REC164_PROTO;

/* -------------- fcbtype (SET/RESET) -------------------------------*/
#define LMSREC_FTYPES      '\xC0'  /* R SAM                          */
#define LMSREC_FTYPEI      '\x40'  /* S ISAM                         */
#define LMSREC_FTYPEP      '\xC0'  /* S PAM                          */
#define LMSREC_VMIN        '\x01'  /* R VALPROP MIN. FUNCT.          */
#define LMSREC_VMAX        '\x01'  /* S VALPROP MAX. FUNCT.          */
/*--------------- share (SET,RESET) -------------------------------- */
#define LMSREC_SHAREY      '\x04'  /* S YES                          */
#define LMSREC_ACCESS      '\x08'  /* S ACCESS=READ                  */
#define LMSREC_SHCCNO      '\xC0'  /* R NO CONTROL CHAR              */
#define LMSREC_SHCCM       '\x40'  /* S MACHINE CODE CONTROL CHAR    */
#define LMSREC_SHCCA       '\xC0'  /* S ASA CONTROL CHAR             */
/* -------------- recform (SET,RESET) ------------------------------ */
#define LMSREC_RECFF       '\x04'  /* S FIXED                        */
#define LMSREC_RECFV       '\x02'  /* S VARIABLE                     */
#define LMSREC_RECFU       '\x06'  /* S UNDEFINED                    */
/* -------------- does key exists in member ? ---------------------- */
```

```
#define LMSREC_KEYY          'Y'   /* YES                              */
#define LMSREC_KEYN          'N'   /* NO                               */
/* -------------- BLKCTRL-indicator -------------------------------- */
#define LMSREC_CTRLN       '\x80'  /* S BLKCTRL=NO                     */
#define LMSREC_CTRLP       '\x40'  /* S BLKCTRL=PAMKEY                 */
#define LMSREC_CTRLD       '\x20'  /* S BLKCTRL=DATA                   */
#define LMSREC_CTRLO       '\x10'  /* S BLKCTRL=NULL                   */
#define LMSREC_CTRLR       '\xF0'  /* R BLKCTRL=NOT SPECIFIED          */
#define LMSREC_BCF4K       '\x08'  /* S BLOCK CONTROL FIELD 4K         */
#define LMSREC_BCF2K       '\x04'  /* S BLOCK CONTROL FIELD 2K         */
#define LMSREC_CTRLU       '\x03'  /* R -- RESERVED, MUST BE 0 --      */
/* -------------- IOPERF-indicator --------------------------------- */
#define LMSREC_PFVH        '\x03'  /* S  IOPERF=VERY-HIGH              */
#define LMSREC_PFHI        '\x02'  /* S  IOPERF=HIGH                   */
#define LMSREC_PFST        '\x01'  /* S  IOPERF=STD                    */
#define LMSREC_PFNS        '\x00'  /* S  IOPERF NOT SPECIFIED          */
/* -------------- IOUSAGE-indicator -------------------------------- */
#define LMSREC_USRW        '\x03'  /* S  IOUSAGE=RDWRT                 */
#define LMSREC_USWR        '\x02'  /* S  IOUSAGE=WRITE                 */
#define LMSREC_USRD        '\x01'  /* S  IOUSAGE=READ                  */
#define LMSREC_USNS        '\x00'  /* S  IOUSAGE NOT SPECIFIED         */
/* -------------- CATALOG-indicator (IDCEX) ------------------------ */
#define LMSREC_ESPEC       '\x08'  /* S  PLAM FILE INDICATOR           */
/* -------------- BLKCTRL-indicator -------------------------------- */
#define LMSREC_AIXIDK      '\x80'  /* S  DUPKEY = YES                  */
                                   /* R  DUPKEY = NO                   */
/* ---------------------------------------------------------------- */
#endif
```

6.3 Example

The following C program has the functions listed below:

1. Open a subroutine access (INIT)
2. Incorporate a file as a member (ADD)
3. Search the directory for a member (TOCPRIM)
4. Open a member (OPENGET)
5. Read a member record by record (GET)
6. Close the member (CLOSE)
7. Terminate a subroutine access (END)

To make the example easier to understand, comments have been included.

```
/* ********************************************************************* */
/*                                                                       */
/*                    EXAMPLE OF LMS AS A SUBROUTINE                     */
/*                                                                       */
/* ********************************************************************* */

#include <stdlib.h>
#include <stdio.h>
#include <string.h>    /* INCLUDE member for copy and other functions */
#include <lms.h>       /* INCLUDE member for LMS structures */

main()
{
  /* Assign and initialize parameter structures */

   struct lmsc_cb cb;  /* Assign parameter structures for cb */
   struct lmsc_ed ed;  /* Assign parameter structures for ed */
   struct lmsc_em em;  /* Assign parameter structures for em */
   struct lmsc_ei ei;  /* Assign parameter structures for ei */
   struct lmsc_fd fd;  /* Assign parameter structures for fd */
   struct lmsc_ld ld;  /* Assign parameter structures for ld */
   struct lmsc_rd rd;  /* Assign parameter structures for rd */

   int tid=1;           /* Initialize the TOC identification */
   char buffer[200];    /* Buffer length of ER; required for GET */
   char temp[100];      /* Buffer for copying for output */
   char * ptemp;        /* Pointer to this buffer */

   cb = lmsc_cb_proto;   /* Initialize control block CB */
   ed = lmsc_ed_proto;   /* Initialize member description */
   em = lmsc_em_proto;   /* Initialize member mask */
   ei = lmsc_ei_proto;   /* Initialize member information */
   fd = lmsc_fd_proto;   /* Initialize file information */
   ld = lmsc_ld_proto;   /* Initialize library description */
   rd = lmsc_rd_proto;   /* Initialize record description */

/* ******************************************************************** */
/*                                                                      */
/*  1. Open a subroutine access with INIT                               */
/*                                                                      */
/* ******************************************************************** */

    cb.function = LMSUP_INIT ;
    cb.subcode  = LMSUP_UNUSED ;
    lmsup1(&cb);

     /* Evaluate return code */

     if (cb.retcode != LMSUP_OK)
       {
        /* An error has occurred -  issue message and terminate program */
```

```
          printf("Error during initialization \n");
          exit(1);
        }
      else
          printf("Initialization successfully terminated \n");


/* ******************************************************************* */
/*                                                                     */
/*  2. Incorporate a file with ADD                                     */
/*                                                                     */
/* ******************************************************************* */

    cb.function=LMSUP_ADD;
    cb.subcode =LMSUP_UNUSED;

    /*     Assign the necessary values for ADD                        */
    /*                                                                */
    /* Note that the arguments to be copied must end with a blank     */
    /* if they are shorter than the target field.                     */

     strfill(ld.name,"#BSP.LIB.C",sizeof(ld.name));
                                /* Assign library name            */
     strfill(fd.name,"#BSP.IN.INPUT",sizeof(fd.name));
                                /* Assign file name               */
     strfill(ed.name,"BSP.ELEMENT",sizeof(ed.name));
                                /* Assign member name             */
     strfill(ed.version,"001",sizeof(ed.version));
                                /* Assign member version          */
     strfill(ed.typ,"S",sizeof(ed.typ));
                                /* Assign member type             */

     /* Program call */

     lmsup1(&cb,&fd,&ld,&ed);

     /* Evaluate return code */

     if (cb.retcode != LMSUP_OK)
      {
        /* An error has occurred - issue message and terminate program */

         printf("Error when incorporating member \n");
         exit(1);
      }
     else
          printf("Member incorporated \n");
```

```
/* ******************************************************************** */
/*                                                                      */
/*  3. Search for a member in the directory with TOCPRIM                */
/*                                                                      */
/* ******************************************************************** */

    cb.function = LMSUP_TOCPRIM;
    cb.subcode = LMSUP_LONG;
    cb.overwrite =LMSUP_YES;

    /*     Assign the necessary values for TOCPRIM                      */
    /*                                                                  */
    /* Note that the arguments to be copied must end with a blank       */
    /* if they are shorter than the target field.                       */

     strfill(em.name,"BSP.ELEMENT",sizeof(em.name));
                                /* Assign member name               */
     strfill(em.version,"001",sizeof(em.version));
                                /* Assign member version            */
     strfill(em.typ,"S",sizeof(em.typ));
                                /* Assign member type               */

     /* Program call */

     lmsup1(&cb,&tid,&ei,&ld,&em);

     /* Evaluate return code */

     if (cb.retcode != LMSUP_OK)
      {
        /* An error has occurred - issue message and terminate program */

         printf("Error when searching for member \n");
         exit(1);
      }
     else
      {
          printf("Member found:\n");
                                               /* Display type      */
          strncpy(temp,ei.typ,sizeof(ei.typ));
          ptemp = index(temp,' ');
          if (ptemp) *ptemp = '\0';
          printf("typ          %s\n",temp);
                                               /* Display name      */
          strncpy(temp,ei.name,sizeof(ei.name));
          ptemp = index(temp,' ');
          if (ptemp) *ptemp = '\0';
          printf("name         %s\n",temp);
```

```
                                                    /* Display version   */
          strncpy(temp,ei.version,sizeof(ei.version));
          ptemp = index(temp,' ');
          if (ptemp) *ptemp = '\0';
          printf("version      %s\n",temp);
                                                    /* Display date      */
          strncpy(temp,ei.user_date,sizeof(ei.user_date));
          ptemp = index(temp,' ');
          if (ptemp) *ptemp = '\0';
          printf("user-date    %s\n\n",temp);
        }

/* ***************************************************************** */
/*                                                                   */
/*  4. Open a member with OPENGET                                    */
/*                                                                   */
/* ***************************************************************** */

    cb.function = LMSUP_OPEN_GET;
    cb.subcode  = LMSUP_UNUSED ;

    /*     Assign the necessary values for TOCPRIM                   */
    /*                                                               */
    /* Note that the arguments to be copied must end with a blank    */
    /* if they are shorter than the target field.                    */

     strfill(ld.name,"#BSP.LIB.C",sizeof(ld.name));
                                /* Assign library name           */
     strfill(ed.name,"BSP.ELEMENT",sizeof(ed.name));
                                /* Assign member name            */
     strfill(ed.version,"001",sizeof(ed.version));
                                /* Assign member version         */
     strfill(ed.typ,"S",sizeof(ed.typ));
                                /* Assign member type            */

     /* Program call */

     lmsup1(&cb,&rd,&ld,&ed);

     /* Evaluate return code */

     if (cb.retcode != LMSUP_OK)
      {
        /* An error has occurred - issue message and terminate program */

         printf("Error when opening member \n");
         exit(1);
      }
     else
          printf("Member opened \n");

/* ****************************************************************** */
/*                                                                    */
/*  5. Read a record with GET                                         */
/*                                                                    */
/* ****************************************************************** */

     do
      {
       cb.function = LMSUP_GET;
       cb.subcode = LMSUP_SEQ;

       /* Assign the necessary values for GET  */

        rd.buffer_len = sizeof(buffer)-1;

        /* Program call */

        lmsup1(&cb,&rd,buffer);

        switch (cb.retcode)    /* Evaluate return code */
         {
          case LMSUP_OK:       /* Output record */
                         buffer[rd.record_len]='\0';
                         printf("%s\n",buffer+4);
                         break;

          case LMSUP_TRUNC:    /* Record truncated */
                         printf("Record buffer too short\n");
                         break;

          case LMSUP_EOF:      /* Member end */
                         break;

          default:             /* An error has occurred - issue message */
                         printf("Error when reading record \n");
                         break;
         }
      }
     while (cb.retcode == LMSUP_OK);
```

```
/* ******************************************************************* */
/*                                                                     */
/*  6. Close a member with CLOSE                                       */
/*                                                                     */
/* ******************************************************************* */

    cb.function = LMSUP_CLOSE;
    cb.subcode = LMSUP_UNUSED;

     /* Program call */

     lmsup1(&cb,&rd);

     /* Evaluate return code */

     if (cb.retcode != LMSUP_OK)
      {
        /* An error has occurred - issue message and terminate program */

         printf("Error when closing member \n");
         exit(1);
      }
     else
          printf("Member closed \n");


/* ******************************************************************* */
/*                                                                     */
/*  7. Terminate subroutine access with END                            */
/*                                                                     */
/* ******************************************************************* */

    cb.function = LMSUP_END;
    cb.subcode  = LMSUP_UNUSED ;

     /* Program call */

     lmsup1(&cb);

     /* Evaluate return code */

     if (cb.retcode != LMSUP_OK)
      {
        /* An error has occurred - issue message and terminate program */

         printf("Error during termination \n");
         exit(1);
      }
```

```
        else
             printf("Subroutine access terminated \n");


    }   /* End of the main program */
```

7 Assembler interface

7.1 Linkage module LMSUP1

LMS functions are called from Assembler programs via the linkage module LMSUP1, which is invoked as follows:

Name	Operation	Operands
	L BALR	15,=V(LMSUP1) 14,15

For the entries in the parameter list, see the description of the various functions (page 17ff).

Module LMSUP1 is fetched from library SYSLNK.LMS.034 and permanently linked to the main program.

For the subroutine branch to LMS, the registers must be loaded as follows:

Register 1:	Address of the parameter list
Register 13:	Address of the save area
Register 14:	Return address
Register 15:	Entry address

Parameter list

The parameter list contains a sequence of addresses which must be supplied by the caller. The relevant addresses for a call depend on the function called and the requisite control blocks.

Save area

The save area is an 18-word register save area which must be supplied by the caller.

Return address

This is the address in the main program to which LMS returns after having executed a function.

Entry address:

LMSUP1

7.2 Generation of the parameter structures for Assembler

In order for the Assembler parameter structures to be generated, library SYSLIB.LMS.033 (containing the macros) must first be assigned. Assignment is possible using the following command:

```
/ADD-FILE-LINK FILE-NAME = $.SYSLIB.LMS.034, LINK-NAME = ALTLIB
```

The installation location of this library is freely selectable via IMON.

The installation location of SYSLIB.LMS.034 can be determined and stored into an S variable using the builtin function INSTALLATION-PATH:

```
/SET-VARIABLE LIBRARY-NAME =INSTALLATION-PATH             -
                            (LOGICAL-ID = ´SYSLIB´        -
                            ,INSTALLATION-UNIT = ´LMS´    -
                            ,VERSION = ´3.4´              -
                            ,DEFAULT-PATH-NAME = ´$.SYSLIB.LMS.034´)
```

Then the library can be assigned as follows:

```
/ADD-FILE-LINK FILE-NAME=&(LIBRARY-NAME),LINK-NAME=ALTLIB
```

No macros are available for parameter structures TID and ER.

LMSASSCB

LMSASSCB generates the function control block.

Name	Operation	Operands
name	LMSASSCB	[P=prefix,] MF = $\left\{\begin{array}{c}D\\C\\L\end{array}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string CB.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSCB

```
***********************************************************************
*                                                                     *
*        FUNCTION CONTROL BLOCK (DSECT)                               *
*                                                                     *
***********************************************************************
         SPACE
&NAME    DSECT
&P.SCBV  DS    CL2                        VERSION OF INTERFACE MACROS   IN  CL2'04'
&P.FUNC  DS    XL1                        FUNCTION CODE                 IN  XL1'01'
&P.SUBC  DS    CL1                        SUBCODE                       IN  CL1' '
&P.ACC   DS    F                          INIT ACCESS ID             INOUT  X'FFFFFFFF'
&P.RTC   DS    XL1                        MAIN RETURNCODE              OUT  XL1'00'
&P.RES1  DS    XL1                        FREE                              XL1'00'
&P.LMSM  DS    H                          LMS-MSG-CODE                 OUT  H'0'
&P.DMSM  DS    H                          DMS-MSG-CODE                 OUT  H'0'
&P.PLAM  DS    H                          PLAM-MSG-CODE                OUT  H'0'
&P.LMSV  DS    CL12                       LMS VERSION                  OUT  CL12' '
&P.RES2  DS    XL4                        FREE                              XL4'00'
*
********************    LMS PARAMETER   *********************
*
&P.DEST  DS    CL1                        DESTROY                    INOUT  CL1' '
&P.FCB   DS    CL1                        FCBTYPE                    INOUT  CL1' '
&P.KEY   DS    CL1                        KEY                        INOUT  CL1' '
&P.OV    DS    CL1                        OVERWRITE                  INOUT  CL1' '
&P.COL   DS    H                          COLUMN                     INOUT  H'0'
&P.LINE  DS    H                          LINE                       INOUT  H'0'
&P.PI    DS    CL1                        PROTECTION INDICATOR       INOUT  CL1' '
&P.ATTR  DS    CL1                        COPYLIB WITH ATTRIBUTES    INOUT  CL1' '
&P.INFO  DS    CL1                        RECORD TYPE FOR ....       INOUT  CL1' '
&P.LDRT  DS    CL1                        DMS-FILENAME IN LD         INOUT  CL1' '
&P.RES3  DS    CL4                        FREE                              CL4' '
&P.PLNG  EQU   *-&P.SCBV                  CB LENGTH
```

LMSASSEA

LMSASSEA generates the member attributes.

Name	Operation	Operands
name	LMSASSEA	[P=prefix,] MF = { D / C / L }

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string EA.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSEA

```
***********************************************************************
*                                                                     *
*        ELEMENT ATTRIBUTES  (DSECT)                                  *
*                                                                     *
***********************************************************************
         SPACE
&NAME    DSECT
&P.UDAT  DS    CL14                   USER DATE                    IN  CL14' '
&P.UTIM  DS    CL8                    USER TIME                    IN  CL8' '
&P.CCSN  DS    CL8                    CODED CHARACTER SET NAME     IN  CL8' '
*
***********    SOURCE CODE CONTROL    *******************************
*
&P.HOSTA DS    CL1                    HOLD FLAG                    IN  CL1' '
.*                                    ' ': UNCHANGE
.*                                    '-': FREE
.*                                    'H': IN HOLD
&P.RES1  DS    CL8                    FREE (HOLDER USERID)         IN  CL8' '
*
&P.MDATI DS    CL1                    MODIFICATION DATE INDICATOR  IN  CL1'O'
.*                                    'O': OLD
.*                                    'S': SYSTEM DATE
&P.RES2  DS    CL56                   FREE                             CL56' '
*
&P.PLNG  EQU   *-&P.UDAT              EA LENGTH
```

LMSASSED

LMSASSED generates the member description.

Name	Operation	Operands
name	LMSASSED	[P=prefix,] MF = $\left\{\begin{array}{c}D\\C\\L\end{array}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string ED.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSED

```
***********************************************************************
*                                                                     *
*        ELEMENT DESCRIPTION (DSECT)                                  *
*                                                                     *
***********************************************************************
         SPACE
&NAME    DSECT
&P.ELEM  DS    0CL96                    ELEMENT IDENTIFIER            IN
&P.TYPE  DS    CL8                      ELEMENT TYPE                  IN  CL8' '
&P.NAME  DS    CL64                     ELEMENT NAME                  IN  CL64' '
&P.VERS  DS    CL24                     ELEMENT VERSION               IN  CL24' '
&P.STOR  DS    CL1                      STORAGE FORM                  IN  CL1'V'
&P.UDAT  DS    CL14                     USER DATE                     IN  CL14' '
&P.UTIM  DS    CL8                      USER TIME                     IN  CL8' '
&P.PLNG  EQU   *-&P.ELEM                ED LENGTH
```

LMSASSEI

LMSASSEI generates the member information.

Name	Operation	Operands
name	LMSASSEI	[P=prefix,] MF = $\left\{\begin{array}{c}D\\C\\L\end{array}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string EI.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSEI

```
***********************************************************************
*                                                                     *
*        ELEMENT INFORMATION (DSECT)                                  *
*                                                                     *
***********************************************************************
         SPACE
&NAME    DSECT
&P.ELEM  DS    0CL96                   ELEMENT IDENTIFIER           OUT
&P.TYPE  DS    CL8                     ELEMENT TYPE                 OUT  CL8' '
&P.NAME  DS    CL64                    ELEMENT NAME                 OUT  CL64' '
&P.VERS  DS    CL24                    ELEMENT VERSION              OUT  CL24' '
&P.STOR  DS    CL1                     STORAGE FORM                 OUT  CL1' '
&P.UDAT  DS    CL14                    USER DATE                    OUT  CL14' '
&P.UTIM  DS    CL8                     USER TIME                    OUT  CL8' '
&P.CDAT  DS    CL14                    CREATION DATE                OUT  CL14' '
&P.CTIM  DS    CL8                     CREATION TIME                OUT  CL8' '
&P.MDAT  DS    CL14                    MODIFICATION DATE            OUT  CL14' '
&P.MTIM  DS    CL8                     MODIFICATION TIME            OUT  CL8' '
&P.SECN  DS    CL32                    SECONDARY NAME               OUT  CL32' '
&P.SECA  DS    CL8                     SECONDARY ATTRIBUTE          OUT  CL8' '
&P.RES1  DS    CL5                     FREE                              CL5' '
*
```

```
***********    PROTECTION ATTRIBUTES   ********************************
*
&P.PTRD  DS    CL1                     PROT. TYPE INDI. FOR READ    OUT  CL1' '
.*                                     'N': NO SPECIAL PROTECTION
.*                                     'Y': SPECIAL PROTECTION
.*                                     'G': PROTECTION BY GUARD
&P.PRDU  DS    CL1                     READ INDICATOR FOR OWNER     OUT  CL1' '
&P.PRDG  DS    CL1                     READ INDICATOR FOR GROUP     OUT  CL1' '
&P.PRDO  DS    CL1                     READ INDICATOR FOR OTHERS    OUT  CL1' '
&P.PRDP  DS    CL1                     READ PASSWORD INDICATOR      OUT  CL1' '
&P.RES2  DS    FL4                                                       FL4'0'
*
&P.PTWR  DS    CL1                     PROT. TYPE INDI. FOR WRITE   OUT  CL1' '
&P.PWRU  DS    CL1                     WRITE INDICATOR FOR OWNER    OUT  CL1' '
&P.PWRG  DS    CL1                     WRITE INDICATOR FOR GROUP    OUT  CL1' '
&P.PWRO  DS    CL1                     WRITE INDICATOR FOR OTHERS   OUT  CL1' '
&P.PWRP  DS    CL1                     WRITE PASSWORD INDICATOR     OUT  CL1' '
&P.RES3  DS    FL4                                                       FL4'0'
*
&P.PTEX  DS    CL1                     PROT. TYPE INDI. FOR EXEC    OUT  CL1' '
&P.PEXU  DS    CL1                     EXEC INDICATOR FOR OWNER     OUT  CL1' '
&P.PEXG  DS    CL1                     EXEC INDICATOR FOR GROUP     OUT  CL1' '
&P.PEXO  DS    CL1                     EXEC INDICATOR FOR OTHERS    OUT  CL1' '
&P.PEXP  DS    CL1                     EXEC PASSWORD INDICATOR      OUT  CL1' '
&P.RES4  DS    FL4                                                       FL4'0'
*
&P.PGRD  DS    CL18                    READ  GUARD                  OUT  CL18' '
&P.PGWR  DS    CL18                    WRITE GUARD                  OUT  CL18' '
&P.PGEX  DS    CL18                    EXEC  GUARD                  OUT  CL18' '
&P.CCSN  DS    CL8                     CODED CHARACTER SET NAME     OUT  CL8' '
*
***********    SOURCE CODE CONTROL     ********************************
*
&P.PTHO  DS    CL1                     IND. FOR HOLDER-AUTHORIZATION UT  CL1' '
.*                                     'N': NO SPECIAL AUTHORIZATION
.*                                     'Y': SPECIAL AUTHORIZATION
.*                                     'G': AUTHORIZATION BY GUARD
&P.PHOU  DS    CL1                     HOLDER INDICATOR FOR OWNER   OUT  CL1' '
&P.PHOG  DS    CL1                     HOLDER INDICATOR FOR GROUP   OUT  CL1' '
&P.PHOO  DS    CL1                     HOLDER INDICATOR FOR OTHERS  OUT  CL1' '
&P.PHOP  DS    CL1                     HOLDER PASSWORD INDICATOR    OUT  CL1' '
&P.RES5  DS    FL4                                                       FL4'0'
&P.PGHO  DS    CL18                    HOLDER GUARD                 OUT  CL18' '
*
&P.HOSTA DS    CL1                     HOLD FLAG                    OUT  CL1' '
.*                                     '-': FREE
.*                                     'H': IN HOLD
&P.HOLD  DS    CL8                     HOLDER USERID                OUT  CL8' '
&P.ADAT  DS    CL14                    ACCESS DATE                  OUT  CL14' '
&P.ATIM  DS    CL8                     ACCESS TIME                  OUT  CL8' '
*
&P.RES6  DS    CL1                                                       CL1' '
```

```
&P.ESIZE DS    F                       ELEMENT-SIZE                 OUT  FL4'0'
*
&P.RES7  DS    CL40                    FREE                              CL40' '
&P.PLNG  EQU   *-&P.ELEM                EI LENGTH
```

LMSASSEM

LMSASSEM generates the member mask.

Name	Operation	Operands
name	LMSASSEM	[P=prefix,] MF = $\left\{\begin{matrix}D\\C\\L\end{matrix}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string EM.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSEM

```
*********************************************************************
*                                                                   *
*        ELEMENT MASK (DSECT)                                       *
*                                                                   *
*********************************************************************
         SPACE
&NAME    DSECT
&P.TYPE  DS    CL20                     ELEMENT TYPE            IN  CL20' '
&P.NAME  DS    CL132                    ELEMENT NAME            IN  CL132' '
&P.VERS  DS    CL52                     ELEMENT VERSION         IN  CL52' '
&P.STOR  DS    CL6                      STORAGE FORM            IN  CL6' '
&P.UDAT  DS    CL32                     USER DATE               IN  CL32' '
&P.UTIM  DS    CL20                     USER TIME               IN  CL20' '
&P.CDAT  DS    CL32                     CREATION DATE           IN  CL32' '
&P.CTIM  DS    CL20                     CREATION TIME           IN  CL20' '
&P.MDAT  DS    CL32                     MODIFICATION DATE       IN  CL32' '
&P.MTIM  DS    CL20                     MODIFICATION TIME       IN  CL20' '
&P.SECN  DS    CL68                     SECONDARY NAME          IN  CL68' '
&P.SECA  DS    CL20                     SECONDARY ATTRIBUTE     IN  CL20' '
&P.RES1  DS    CL14                     FREE                        CL14' '
*
```

```
***********    PROTECTION ATTRIBUTES   ********************************
*
&P.PTRD  DS    CL1                     PROT. TYPE INDICATOR FOR READ IN  CL1' '
.*                                     'A': ANY PROTECTION
.*                                     'N': NO SPECIAL PROTECTION
.*                                     'Y': SPECIAL PROTECTION
.*                                     'G': PROTECTION BY GUARD
&P.PRDU  DS    CL1                     READ INDICATOR FOR OWNER      IN  CL1' '
&P.PRDG  DS    CL1                     READ INDICATOR FOR GROUP      IN  CL1' '
&P.PRDO  DS    CL1                     READ INDICATOR FOR OTHERS     IN  CL1' '
&P.PRDP  DS    CL1                     READ PASSWORD INDICATOR       IN  CL1' '
&P.RES2  DS    FL4                                                       FL4'0'
*
&P.PTWR  DS    CL1                     PROT. TYPE IND. FOR WRITE     IN  CL1' '
&P.PWRU  DS    CL1                     WRITE INDICATOR FOR OWNER     IN  CL1' '
&P.PWRG  DS    CL1                     WRITE INDICATOR FOR GROUP     IN  CL1' '
&P.PWRO  DS    CL1                     WRITE INDICATOR FOR OTHERS    IN  CL1' '
&P.PWRP  DS    CL1                     WRITE PASSWORD INDICATOR      IN  CL1' '
&P.RES3  DS    FL4                                                       FL4'0'
*
&P.PTEX  DS    CL1                     PROT. TYPE INDICATOR FOR EXEC IN  CL1' '
&P.PEXU  DS    CL1                     EXEC INDICATOR FOR OWNER      IN  CL1' '
&P.PEXG  DS    CL1                     EXEC INDICATOR FOR GROUP      IN  CL1' '
&P.PEXO  DS    CL1                     EXEC INDICATOR FOR OTHERS     IN  CL1' '
&P.PEXP  DS    CL1                     EXEC PASSWORD INDICATOR       IN  CL1' '
&P.RES4  DS    FL4                                                       FL4'0'
*
&P.PGRD  DS    CL40                    READ  GUARD                   IN  CL40' '
&P.PGWR  DS    CL40                    WRITE GUARD                   IN  CL40' '
&P.PGEX  DS    CL40                    EXEC  GUARD                   IN  CL40' '
&P.CCSN  DS    CL20                    CODED CHARACTER SET NAME      IN  CL20' '
*
***********    SOURCE CODE CONTROL     ********************************
*
&P.PTHO  DS    CL1                     IND. FOR HOLDER-AUTHORIZATION IN  CL1' '
.*                                     'N': NO SPECIAL AUTHORIZATION
.*                                     'Y': SPECIAL AUTHORIZATION
.*                                     'G': AUTHORIZATION BY GUARD
&P.PHOU  DS    CL1                     HOLDER INDICATOR FOR OWNER    IN  CL1' '
&P.PHOG  DS    CL1                     HOLDER INDICATOR FOR GROUP    IN  CL1' '
&P.PHOO  DS    CL1                     HOLDER INDICATOR FOR OTHERS   IN  CL1' '
&P.PHOP  DS    CL1                     HOLDER PASSWORD INDICATOR     IN  CL1' '
&P.RES5  DS    FL4                                                       FL4'0'
&P.PGHO  DS    CL40                    HOLDER GUARD                  IN  CL40' '
*
&P.HOSTA DS    CL1                     HOLD FLAG                     IN  CL1' '
.*                                     ' ': ANY
.*                                     '-': FREE
.*                                     'H': IN HOLD
&P.HOLD  DS    CL20                    HOLDER USERID                 IN  CL20' '
&P.ADAT  DS    CL32                    ACCESS DATE                   IN  CL32' '
&P.ATIM  DS    CL20                    ACCESS TIME                   IN  CL20' '
```

```
*
&P.RES6  DS    CL3                                                    CL3' '
&P.ESMIN DS    F                        ELEMENT-SIZE    MINIMUM       IN  FL4'0'
&P.ESMAX DS    F                        ELEMENT-SIZE    MAXIMUM       IN  X'FFFFFFFF'
*
&P.RES7  DS    CL64                      FREE                             CL64' '
&P.PLNG  EQU   *-&P.TYPE                 EM LENGTH
```

LMSASSFD

LMSASSFD generates the file description.

Name	Operation	Operands
name	LMSASSFD	[P=prefix,] MF = $\begin{Bmatrix} D \\ C \\ L \end{Bmatrix}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string FD.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSFD

```
**********************************************************************
*                                                                    *
*        FILE DESCRIPTION (DSECT)                                    *
*                                                                    *
**********************************************************************
         SPACE
&NAME    DSECT
&P.PSWD  DS    F                DMS PASSWORD                   IN  F'0'
&P.LINK  DS    CL8              LINK NAME                      IN  CL8' '
&P.NAME  DS    CL54             FILE NAME                   INOUT  CL54' '
&P.PLNG  EQU   *-&P.PSWD        FD LENGTH
```

LMSASSLA

LMSASSLA generates the administration privilege for the library.

Name	Operation	Operands
name	LMSASSLA	[P=prefix,] MF = $\left\{\begin{matrix}\text{D}\\\text{C}\\\text{L}\end{matrix}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string LA.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSLA

```
**********************************************************************
*                                                                    *
*        LIBRARY ATTRIBUTES (DSECT)                                  *
*                                                                    *
**********************************************************************
         SPACE
&NAME    DSECT
&P.PTAD  DS    CL1                     PROT. TYPE INDI. FOR ADMIN    IN  CL1' '
.*                                     'N': NO SPECIAL PROTECTION
.*                                     'Y': SPECIAL PROTECTION
.*                                     'G': PROTECTION BY GUARD
&P.PADU  DS    CL1                     ADMIN INDICATOR FOR OWNER     IN  CL1' '
&P.PADG  DS    CL1                     ADMIN INDICATOR FOR GROUP     IN  CL1' '
&P.PADO  DS    CL1                     ADMIN INDICATOR FOR OTHERS    IN  CL1' '
&P.PADP  DS    CL1                     ADMIN PASSWORD INDICATOR      IN  CL1' '
&P.PADW  DS    FL4                     ADMIN PASSWORD                IN  FL4'0'
&P.PGAD  DS    CL18                    ADMIN GUARD                   IN  CL18' '
*
&P.STOR  DS    CL1                     STORAGE FORM FOR LIBRARY      IN  CL1' '
.*                                     ' ': UNCHANGE
.*                                     'S': STD ( FULL OR DELTA )
.*                                     'V': FULL  ELEMENT
.*                                     'D': DELTA ELEMENT
&P.WRCT  DS    CL1                     WRITE-CONTROL FOR LIBRARY     IN  CL1' '
.*                                     ' ': UNCHANGE
.*                                     'A': ACTIVATE
.*                                     'D': DEACTIVATE
&P.ADAT  DS    CL1                     ACCESS DATE                   IN  CL1' '
.*                                     ' ': UNCHANGE
.*                                     'N': NONE (DONT KEEP)
.*                                     'K': KEEP
&P.RES1  DS    CL34                    FREE                              CL34' '
&P.PLNG  EQU   *-&P.PTAD                LA LENGTH
```

LMSASSLD

LMSASSLD generates the library description.

Name	Operation	Operands
name	LMSASSLD	[P=prefix,] MF = $\left\{\begin{matrix}D\\C\\L\end{matrix}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string LD.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSLD

```
*********************************************************************
*                                                                   *
*        LIBRARY DESCRIPTION (DSECT)                                *
*                                                                   *
*********************************************************************
         SPACE
&NAME    DSECT
&P.PSWD  DS    F                DMS PASSWORD                 IN  F'0'
&P.LINK  DS    CL8              LINK NAME                    IN  CL8' '
&P.RES1  DS    2F               FREE                             2F'0'
&P.LEN   DS    Y                MAX. LENGTH OF LIBRARY NAME  IN  Y(54)
&P.NAME  DS    CL54             LIBRARY NAME              INOUT  CL54' '
&P.PLNG  EQU   *-&P.PSWD        LD LENGTH
```

LMSASSLI

LMSASSLI generates the library information.

Name	Operation	Operands
name	LMSASSL	[P=prefix,] MF = { D / C / L }

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string LI.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSLI

```
**********************************************************************
*                                                                    *
*        LIBRARY INFORMATION (DSECT)                                 *
*                                                                    *
**********************************************************************
         SPACE
&NAME    DSECT
&P.PTAD  DS    CL1                     PROT. TYPE INDI. FOR ADMIN   OUT  CL1' '
.*                                     'N': NO SPECIAL PROTECTION
.*                                     'Y': SPECIAL PROTECTION
.*                                     'G': PROTECTION BY GUARD
&P.PADU  DS    CL1                     ADMIN INDICATOR FOR OWNER    OUT  CL1' '
&P.PADG  DS    CL1                     ADMIN INDICATOR FOR GROUP    OUT  CL1' '
&P.PADO  DS    CL1                     ADMIN INDICATOR FOR OTHERS   OUT  CL1' '
&P.PADP  DS    CL1                     ADMIN PASSWORD INDICATOR     OUT  CL1' '
&P.RES1  DS    FL4                                                       FL4'0'
&P.PGAD  DS    CL18                    ADMIN GUARD                  OUT  CL18' '
*
&P.STOR  DS    CL1                     STORAGE FORM FOR LIBRARY     OUT  CL1' '
.*                                     'S': STD ( FULL OR DELTA )
.*                                     'V': FULL  ELEMENT
.*                                     'D': DELTA ELEMENT
&P.WRCT  DS    CL1                     WRITE-CONTROL FOR LIBRARY    OUT  CL1' '
.*                                     'A': ACTIVE
.*                                     'D': DEACTIVATED
&P.ADAT  DS    CL1                     ACCESS DATE                  OUT  CL1' '
```

```
.*                                    'N': NONE (DONT KEEP)
.*                                    'K': KEEP
&P.RES2  DS    CL24                   FREE                              CL24' '
&P.LFORM DS    CL1                    LIBRARY FORMAT (NK2/NK4)     OUT  CL1' '
*                                     '2' : NK2 LIBRARY FORMAT
*                                     '4' : NK4 LIBRARY FORMAT
&P.UPROT DS    CL1                    UPAM PROTECTED (YES/NO)      OUT  CL1' '
*                                     'Y' : LIB IS UPAM PROTECTED
*                                     'N' : LIB IS NOT UPAM PROTECTED
&P.FILSZ DS    F                      FILE SIZE                    OUT  F'0'
&P.FRESZ DS    F                      FREE SIZE                    OUT  F'0'
*
***********    PROTECTION ATTRIBUTES   ********************************
*
&P.PTRD  DS    CL1                    PROT. TYPE INDI. FOR READ    OUT  CL1' '
&P.PRDU  DS    CL1                    READ INDICATOR FOR OWNER     OUT  CL1' '
&P.PRDG  DS    CL1                    READ INDICATOR FOR GROUP     OUT  CL1' '
&P.PRDO  DS    CL1                    READ INDICATOR FOR OTHERS    OUT  CL1' '
&P.PRDP  DS    CL1                    READ PASSWORD INDICATOR      OUT  CL1' '
&P.RES3  DS    FL4                                                      FL4'0'
*
&P.PTWR  DS    CL1                    PROT. TYPE INDI. FOR WRITE   OUT  CL1' '
&P.PWRU  DS    CL1                    WRITE INDICATOR FOR OWNER    OUT  CL1' '
&P.PWRG  DS    CL1                    WRITE INDICATOR FOR GROUP    OUT  CL1' '
&P.PWRO  DS    CL1                    WRITE INDICATOR FOR OTHERS   OUT  CL1' '
&P.PWRP  DS    CL1                    WRITE PASSWORD INDICATOR     OUT  CL1' '
&P.RES4  DS    FL4                                                      FL4'0'
*
&P.PTEX  DS    CL1                    PROT. TYPE INDI. FOR EXEC    OUT  CL1' '
&P.PEXU  DS    CL1                    EXEC INDICATOR FOR OWNER     OUT  CL1' '
&P.PEXG  DS    CL1                    EXEC INDICATOR FOR GROUP     OUT  CL1' '
&P.PEXO  DS    CL1                    EXEC INDICATOR FOR OTHERS    OUT  CL1' '
&P.PEXP  DS    CL1                    EXEC PASSWORD INDICATOR      OUT  CL1' '
&P.RES5  DS    FL4                                                      FL4'0'
*
&P.PGRD  DS    CL18                   READ  GUARD                  OUT  CL18' '
&P.PGWR  DS    CL18                   WRITE GUARD                  OUT  CL18' '
&P.PGEX  DS    CL18                   EXEC  GUARD                  OUT  CL18' '
*
&P.PTHO  DS    CL1                    IND. FOR HOLDER-AUTHORIZATION     CL1' '
.*                                    'N': NO SPECIAL AUTHORIZATION
.*                                    'Y': SPECIAL AUTHORIZATION
.*                                    'G': AUTHORIZATION BY GUARD
&P.PHOU  DS    CL1                    HOLDER INDICATOR FOR OWNER   OUT  CL1' '
&P.PHOG  DS    CL1                    HOLDER INDICATOR FOR GROUP   OUT  CL1' '
&P.PHOO  DS    CL1                    HOLDER INDICATOR FOR OTHERS  OUT  CL1' '
&P.PHOP  DS    CL1                    HOLDER PASSWORD INDICATOR    OUT  CL1' '
&P.RES6  DS    FL4                                                      FL4'0'
&P.PGHO  DS    CL18                   HOLDER GUARD                 OUT  CL18' '
*
&P.RES7  DS    CL68                   FREE                              CL68' '
&P.PLNG  EQU   *-&P.PTAD              LI LENGTH
```

LMSASSPA

LMSASSPA generates the administration privilege and default values for member protection in the library.

Name	Operation	Operands
name	LMSASSPA	[P=prefix,] MF = { D / C / L }

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string PA.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSPA

```
***********************************************************************
*                                                                     *
*        PROTECTION ATTRIBUTES (DSECT)                                *
*                                                                     *
***********************************************************************
         SPACE
&NAME    DSECT
&P.PTRD  DS    CL1                    PROT. TYPE INDI. FOR READ     IN  CL1' '
.*                                    'N': NO SPECIAL PROTECTION
.*                                    'Y': SPECIAL PROTECTION
.*                                    'G': PROTECTION BY GUARD
&P.PRDU  DS    CL1                    READ INDICATOR FOR OWNER      IN  CL1' '
&P.PRDG  DS    CL1                    READ INDICATOR FOR GROUP      IN  CL1' '
&P.PRDO  DS    CL1                    READ INDICATOR FOR OTHERS     IN  CL1' '
&P.PRDP  DS    CL1                    READ PASSWORD INDICATOR       IN  CL1' '
&P.PRDW  DS    FL4                    READ PASSWORD                 IN  FL4'0'
*
&P.PTWR  DS    CL1                    PROT. TYPE INDI. FOR WRITE    IN  CL1' '
&P.PWRU  DS    CL1                    WRITE INDICATOR FOR OWNER     IN  CL1' '
&P.PWRG  DS    CL1                    WRITE INDICATOR FOR GROUP     IN  CL1' '
&P.PWRO  DS    CL1                    WRITE INDICATOR FOR OTHERS    IN  CL1' '
&P.PWRP  DS    CL1                    WRITE PASSWORD INDICATOR      IN  CL1' '
&P.PWRW  DS    FL4                    WRITE PASSWORD                IN  FL4'0'
*
&P.PTEX  DS    CL1                    PROT. TYPE INDI. FOR EXEC     IN  CL1' '
```

```
&P.PEXU  DS    CL1                      EXEC INDICATOR FOR OWNER      IN  CL1' '
&P.PEXG  DS    CL1                      EXEC INDICATOR FOR GROUP      IN  CL1' '
&P.PEXO  DS    CL1                      EXEC INDICATOR FOR OTHERS     IN  CL1' '
&P.PEXP  DS    CL1                      EXEC PASSWORD INDICATOR       IN  CL1' '
&P.PEXW  DS    FL4                      EXEC PASSWORD                 IN  FL4'0'
*
&P.PGRD  DS    CL18                     READ  GUARD                   IN  CL18' '
&P.PGWR  DS    CL18                     WRITE GUARD                   IN  CL18' '
&P.PGEX  DS    CL18                     EXEC  GUARD                   IN  CL18' '
*
***********    SOURCE CODE CONTROL      ********************************
*
&P.PTHO  DS    CL1                      IND. FOR HOLDER-AUTHORIZATION IN  CL1' '
.*                                      'N': NO SPECIAL AUTHORIZATION
.*                                      'Y': SPECIAL AUTHORIZATION
.*                                      'G': AUTHORIZATION BY GUARD
&P.PHOU  DS    CL1                      HOLDER INDICATOR FOR OWNER    IN  CL1' '
&P.PHOG  DS    CL1                      HOLDER INDICATOR FOR GROUP    IN  CL1' '
&P.PHOO  DS    CL1                      HOLDER INDICATOR FOR OTHERS   IN  CL1' '
&P.PHOP  DS    CL1                      HOLDER PASSWORD INDICATOR     IN  CL1' '
&P.PHOW  DS    FL4                      HOLDER PASSWORD               IN  FL4'0'
&P.PGHO  DS    CL18                     HOLDER GUARD                  IN  CL18' '
&P.RES1  DS    CL84                     FREE                              CL84' '
&P.PLNG  EQU   *-&P.PTRD                PA LENGTH
```

LMSASSRD

LMSASSRD generates the record description.

Name	Operation	Operands
name	LMSASSRD	[P=prefix,] MF = $\left\{\begin{matrix}D\\C\\L\end{matrix}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string RD.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSRD

```
**********************************************************************
*                                                                    *
*        RECORD DESCRIPTION (DSECT)                                  *
*                                                                    *
**********************************************************************
         SPACE
&NAME    DSECT
&P.ACC   DS    F                       READ ACCESS ID        INOUT  X'FFFFFFFF'
&P.BLEN  DS    F                       BUFFER LENGTH            IN  F'0'
&P.RLEN  DS    F                       RECORD LENGTH         INOUT  F'0'
&P.RES1  DS    XL3                     FREE                         XL3'000000'
&P.RECID DS    0XL5                    RECORD ID             INOUT
&P.RTYP  DS    XL1                     RECORD TYPE           INOUT  XL1'01'
&P.RNUM  DS    F                       RECORD NUMBER         INOUT  F'0'
&P.RES2  DS    2F                      FREE                         2F'0'
&P.PLNG  EQU   *-&P.ACC                RD LENGTH
```

LMSASSTA

LMSASSTA generates the type attributes.

Name	Operation	Operands
name	LMSASSTA	[P=prefix,] MF = {D / C / L}

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string TA.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSTA

```
***********************************************************************
*                                                                     *
*        TYPE ATTRIBUTES (DSECT)                                      *
*                                                                     *
***********************************************************************
         SPACE
&NAME    DSECT
&P.CONV  DS    CL1                     TYPE CONVENTION             IN  CL1' '
.*                                     'N' - NONE
.*                                     'S' - STD-SEQUENCE
.*                                     'M' - MULTI-SEQUENCE
.*                                     'T' - STD-TREE
&P.RES1  DS    CL3                     FREE                            CL3' '
&P.EXPL  DS    CL24                    VERSION EXAMPLE (F. SEQ.)   IN  CL24' '
*
&P.PTAD  DS    CL1                     PROT. TYPE INDI. FOR ADMIN  IN  CL1' '
.*                                     'N': NO SPECIAL PROTECTION
.*                                     'Y': SPECIAL PROTECTION
.*                                     'G': PROTECTION BY GUARD
&P.PADU  DS    CL1                     ADMIN INDICATOR FOR OWNER   IN  CL1' '
&P.PADG  DS    CL1                     ADMIN INDICATOR FOR GROUP   IN  CL1' '
&P.PADO  DS    CL1                     ADMIN INDICATOR FOR OTHERS  IN  CL1' '
&P.PADP  DS    CL1                     ADMIN PASSWORD INDICATOR    IN  CL1' '
&P.PADW  DS    FL4                     ADMIN PASSWORD              IN  FL4'0'
&P.PGAD  DS    CL18                    ADMIN GUARD                 IN  CL18' '
*
```

```
&P.STOR  DS    CL1                     STORAGE FORM FOR LIBRARY      IN  CL1' '
.*                                     ' ': UNCHANGE
.*                                     'N': NONE
.*                                     'S': STD ( FULL OR DELTA )
.*                                     'V': FULL  ELEMENT
.*                                     'D': DELTA ELEMENT
&P.WRCT  DS    CL1                     WRITE-CONTROL FOR LIBRARY     IN  CL1' '
.*                                     ' ': UNCHANGE
.*                                     'N': NONE
.*                                     'A': ACTIVATE
.*                                     'D': DEACTIVATE
&P.STYP  DS    CL8                     SUPER TYPE                    IN  CL8' '
*
&P.RES2  DS    CL47                    FREE                              CL47' '
&P.PLNG  EQU   *-&P.CONV               TA LENGTH
```

LMSASSTD

LMSASSTD generates the member type.

Name	Operation	Operands
name	LMSASSTD	[P=prefix,] MF = {D / C / L}

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string TD.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSTD

```
**********************************************************************
*                                                                    *
*        TYPE DESCRIPTION (DSECT)                                    *
*                                                                    *
**********************************************************************
         SPACE
&NAME    DSECT
&P.TYPE  DS    CL8                     ELEMENT TYPE                IN  CL8' '
&P.RES1  DS    CL8                     FREE                            CL8' '
&P.PLNG  EQU   *-&P.TYPE               TD LENGTH
```

LMSASSTI

LMSASSTI generates the type information.

Name	Operation	Operands
name	LMSASSTI	[P=prefix,] MF = $\left\{\begin{matrix}D\\C\\L\end{matrix}\right\}$

prefix	Up to 3 characters to be prefixed to the field names. The default value is the string TI.
D	A dummy section (DSECT) is generated.
C	A storage area is generated (without CSECT statement).
L	Constants are defined which can be used for initialization of the control blocks.

Expansion of LMSASSTI

```
**********************************************************************
*                                                                    *
*        TYPE INFORMATION (DSECT)                                    *
*                                                                    *
**********************************************************************
         SPACE
&NAME    DSECT
&P.TYPE  DS    CL8                      ELEMENT TYPE              OUT  CL8' '
&P.RES1  DS    CL8                      FREE                           CL8' '
*
&P.CONV  DS    CL1                      TYPE CONVENTION           OUT  CL1' '
.*                                      'N' - NONE
.*                                      'S' - STD-SEQUENCE
.*                                      'M' - MULTI-SEQUENCE
.*                                      'T' - STD-TREE
&P.RES2  DS    CL3                      FREE                           CL3' '
&P.EXPL  DS    CL24                     VERSION EXAMPLE (F. SEQ.)  OUT  CL24' '
*
&P.PTAD  DS    CL1                      PROT. TYPE INDI. FOR ADMIN OUT  CL1' '
.*                                      'N': NO SPECIAL PROTECTION
.*                                      'Y': SPECIAL PROTECTION
.*                                      'G': PROTECTION BY GUARD
&P.PADU  DS    CL1                      ADMIN INDICATOR FOR OWNER  OUT  CL1' '
&P.PADG  DS    CL1                      ADMIN INDICATOR FOR GROUP  OUT  CL1' '
&P.PADO  DS    CL1                      ADMIN INDICATOR FOR OTHERS OUT  CL1' '
&P.PADP  DS    CL1                      ADMIN PASSWORD INDICATOR   OUT  CL1' '
&P.RES3  DS    FL4                                                     FL4'0'
```

```
&P.PGAD  DS    CL18                     ADMIN GUARD                 OUT  CL18' '
*
&P.STOR  DS    CL1                      STORAGE FORM FOR TYPE       OUT  CL1' '
.*                                      'N': NONE
.*                                      'S': STD ( FULL OR DELTA )
.*                                      'V': FULL  ELEMENT
.*                                      'D': DELTA ELEMENT
&P.WRCT  DS    CL1                      WRITE-CONTROL FOR TYPE      OUT  CL1' '
.*                                      'N': NONE
.*                                      'A': ACTIVE
.*                                      'D': DEACTIVATED
&P.STYP  DS    CL8                      SUPER TYPE                  OUT  CL8' '
&P.BTYP  DS    CL8                      BASIS TYPE                  OUT  CL8' '
*
&P.RES4  DS    CL39                     FREE                             CL39' '
*
***********    PROTECTION ATTRIBUTES    ********************************
*
&P.PTRD  DS    CL1                      PROT. TYPE INDI. FOR READ   OUT  CL1' '
.*                                      'N': NO SPECIAL PROTECTION
.*                                      'Y': SPECIAL PROTECTION
.*                                      'G': PROTECTION BY GUARD
&P.PRDU  DS    CL1                      READ INDICATOR FOR OWNER    OUT  CL1' '
&P.PRDG  DS    CL1                      READ INDICATOR FOR GROUP    OUT  CL1' '
&P.PRDO  DS    CL1                      READ INDICATOR FOR OTHERS   OUT  CL1' '
&P.PRDP  DS    CL1                      READ PASSWORD INDICATOR     OUT  CL1' '
&P.RES5  DS    FL4                                                       FL4'0'
*
&P.PTWR  DS    CL1                      PROT. TYPE INDI. FOR WRITE  OUT  CL1' '
&P.PWRU  DS    CL1                      WRITE INDICATOR FOR OWNER   OUT  CL1' '
&P.PWRG  DS    CL1                      WRITE INDICATOR FOR GROUP   OUT  CL1' '
&P.PWRO  DS    CL1                      WRITE INDICATOR FOR OTHERS  OUT  CL1' '
&P.PWRP  DS    CL1                      WRITE PASSWORD INDICATOR    OUT  CL1' '
&P.RES6  DS    FL4                                                       FL4'0'
*
&P.PTEX  DS    CL1                      PROT. TYPE INDI. FOR EXEC   OUT  CL1' '
&P.PEXU  DS    CL1                      EXEC INDICATOR FOR OWNER    OUT  CL1' '
&P.PEXG  DS    CL1                      EXEC INDICATOR FOR GROUP    OUT  CL1' '
&P.PEXO  DS    CL1                      EXEC INDICATOR FOR OTHERS   OUT  CL1' '
&P.PEXP  DS    CL1                      EXEC PASSWORD INDICATOR     OUT  CL1' '
&P.RES7  DS    FL4                                                       FL4'0'
*
&P.PGRD  DS    CL18                     READ  GUARD                 OUT  CL18' '
&P.PGWR  DS    CL18                     WRITE GUARD                 OUT  CL18' '
&P.PGEX  DS    CL18                     EXEC  GUARD                 OUT  CL18' '
*
&P.PTHO  DS    CL1                      IND. FOR HOLDER AUTHORIZATION OUT  CL1' '
.*                                      'N': NO SPECIAL AUTHORIZATION
.*                                      'Y': SPECIAL AUTHORIZATION
.*                                      'G': AUTHORIZATION BY GUARD
&P.PHOU  DS    CL1                      HOLDER INDICATOR FOR OWNER  OUT  CL1' '
&P.PHOG  DS    CL1                      HOLDER INDICATOR FOR GROUP  OUT  CL1' '
```

```
&P.PHOO  DS    CL1                        HOLDER INDICATOR FOR OTHERS  OUT  CL1' '
&P.PHOP  DS    CL1                        HOLDER PASSWORD INDICATOR    OUT  CL1' '
&P.RES8  DS    FL4                                                          FL4'0'
&P.PGHO  DS    CL18                       HOLDER GUARD                 OUT  CL18' '
&P.RES9  DS    CL52                       FREE                              CL52' '
&P.PLNG  EQU   *-&P.TYPE                  TA LENGTH
```

7.3 Programming aids

LMSASSEQ symbolic names

LMSASSEQ generates a number of equates which serve as a programming aid for processing operand values, function codes, subcodes, return codes and storage mode of members.

For the values of the processing operands see the meanings in [1].

Name	Operation	Operands
name	LMSASSEQ	[P=prefix]

prefix	Up to 3 characters to be prefixed to the field names. By default, the field names have no prefix.

Expansion of LMSASSEQ

```
 *****************   LMS PARAMETER VALUES   ****************************
*
&P.YES   EQU   'Y'                     YES
&P.NO    EQU   'N'                     NO
&P.NONE  EQU   'N'                     NONE
&P.ANY   EQU   ' '                     ANY
&P.UNCH  EQU   ' '                     UNCHANGED
&P.SAME  EQU   'M'                     SAME
&P.STD   EQU   'S'                     STD
*****************   FOR: CBFCB  *****************************
&P.ISAM  EQU   'I'                     ISAM
&P.SAM   EQU   'Q'                     SAM
&P.CAT   EQU   'C'                     CAT
*****************   FOR: CBOV   *****************************
&P.EXT   EQU   'E'                     EXTEND
&P.ONLY  EQU   'O'                     ONLY
&P.NAME  EQU   'A'                     NAME
*****************   FOR: CBINFO  ****************************
&P.TXT   EQU   X'01'                   TEXT ONLY
&P.COM   EQU   X'02'                   COMMENT / DOCUMENTATION ONLY
*
*****************   FUNCTION CODES  ***********************************
*
&P.INIT  EQU   X'01'   INIT              ( CB )
&P.END   EQU   X'02'   END               ( CB )
&P.TOCP  EQU   X'03'   TOCPRIM           ( CB, TID,EI, LD, EM )
&P.TOCS  EQU   X'04'   TOCSEC            ( CB, TID,EI, LD, EM )
```

```
&P.TOC   EQU   X'05'   TOC               ( CB, TID,EI )
&P.REN   EQU   X'06'   REN               ( CB, LD, ED1,ED2 )
&P.DEL   EQU   X'07'   DEL               ( CB, LD, ED )
&P.ADD   EQU   X'08'   ADD               ( CB, FD, LD, ED1 [,ED2] )
&P.SEL   EQU   X'09'   SEL               ( CB, LD, ED, FD )
&P.COPY  EQU   X'0A'   COPY              ( CB, LD1,ED1,LD2,ED2 [,ED3] )
&P.COPST EQU   X'0B'   COPYSTR           ( CB, LD1,ED1,LD2,ED2 )
&P.LOCK  EQU   X'0C'   LOCK              ( CB, LD, ED )
&P.UNLK  EQU   X'0D'   UNLOCK            ( CB, LD, ED )
&P.OPENG EQU   X'0E'   OPEN GET          ( CB, RD, LD, ED )
&P.OPENP EQU   X'0F'   OPEN PUT          ( CB, RD, LD, ED1 [,ED2] )
&P.OPENU EQU   X'10'   OPEN UPD          ( CB, RD, LD, ED )
&P.GET   EQU   X'11'   GET               ( CB, RD, ER )
&P.PUT   EQU   X'12'   PUT               ( CB, RD, ER )
&P.CLOSE EQU   X'13'   CLOSE             ( CB, RD )
&P.LST   EQU   X'14'   LIST ELEMENT      ( CB, LD, ED )
&P.MEP   EQU   X'15'   MODIFY PROTECTION ( CB, LD, ED, PA )
&P.MLA   EQU   X'16'   MODIFY LIB. ATTR. ( CB, LD, LA, PA )
&P.SLA   EQU   X'17'   SHOW   LIB. ATTR. ( CB, LD, LI )
&P.MTA   EQU   X'18'   MODIFY TYPE ATTR. ( CB, LD, TD, TA, PA )
&P.STA   EQU   X'19'   SHOW   TYPE ATTR. ( CB, LD, TD, TI )
&P.MEA   EQU   X'1A'   MODIFY ELEM. ATTR.( CB, LD, ED, EA )
&P.COPLB EQU   X'1B'   COPY LIBRARY      ( CB, LD1,LD2 )
&P.CLOLB EQU   X'1C'   CLOSE LIBRARY     ( CB, LD )
&P.PROVI EQU   X'1D'   PROVIDE ELEMENT   ( CB, LD1,ED1,LD2,ED2 )
&P.RETUR EQU   X'1E'   RETURN ELEMENT    ( CB, LD1,ED1,LD2,ED2,ED3 )
&P.GSYSE EQU   X'1F'   GET SYSELEM       ( CB, LD, ED )
&P.REOLB EQU   X'20'   REORGANIZE LIB    ( CB, LD )
*
*****************   SUBCODES   ****************************************
*
&P.UNUSE EQU   ' '                     SUBCODE UNUSED (DEFAULT)
&P.SHORT EQU   'S'                     TOC SHORT
&P.LONG  EQU   'L'                     TOC LONG
&P.DIR   EQU   'D'                     READ DIRECT
&P.SEQ   EQU   'S'                     READ SEQUENTIAL
&P.WRITE EQU   'W'                     CLOSE OUTPUT ELEMENT FOR WRITE
&P.RESET EQU   'R'                     FORGET OUTPUT ELEMENT
&P.SYM   EQU   'S'                     SHOW ELEMENT SYMBOLIC
&P.HEX   EQU   'H'                     SHOW-ELEMENT ALPHA+HEX
&P.INCP  EQU   'P'                     INCREMENT WITH PREFIX
&P.INCB  EQU   'B'                     INCREMENT WITH BASE
&P.HIGP  EQU   'H'                     HIGHEST EXISTING WITH PREFIX
&P.EXTRA EQU   'X'                     FORMAT-B RECORDS ALLOWED
*
```

```
*****************   RETURNCODES   ************************************
*
&P.OK    EQU   X'00'                    OK
&P.TRUNC EQU   X'04'                    RECORD TRUNCATED
&P.EOF   EQU   X'08'                    END OF GET/TOC
&P.LMSER EQU   X'0C'                    LMS ERROR
&P.PARER EQU   X'14'                    PARAMETER ERROR
&P.SEQER EQU   X'18'                    SEQUENCE ERROR
&P.INTER EQU   X'1C'                    LMS INTERNAL ERROR
*
*****************   STORAGE FORM   ***********************************
*
&P.FULL  EQU   'V'                      FULL  ELEMENT
&P.DELTA EQU   'D'                      DELTA ELEMENT
*
*****************  CONVENTIONS  **************************************
*
&P.CNONE EQU     'N'                    NONE
&P.CSEQ  EQU    'S'                     STD-SEQUENCE
&P.CMSEQ EQU    'M'                     MULTI-SEQUENCE
&P.CTREE EQU    'T'                     STD-TREE
*
*****************  PROTECTION INDICATORS  ****************************
*
&P.PNONE EQU   'N'                      NONE
&P.PSTD  EQU   'Y'                      STD-PROTECTION
&P.PGD   EQU   'G'                      PROTECTION BY GUARD
*
***********    SOURCE CODE CONTROL     ********************************
*
&P.FREE  EQU   '-'                      FREE
&P.INHLD EQU   'H'                      IN HOLD
*
***********    WRITE CONTROL           ********************************
*
&P.ACTIV EQU   'A'                      ACTIVATED
&P.DEACT EQU   'D'                      DEACTIVATED
*
***********    ACCESS DATE             ********************************
*
&P.KEEP  EQU   'K'                      KEEP
*
***********    MODIFICATION DATE INDICATOR   **************************
*
&P.OLD   EQU   'O'                      BY-SOURCE
&P.SDAT  EQU   'S'                      NEW (SYSTEM DATE)
.*
```

Format of a record of type 163

The LMSAS163 record is described below as an example of a record of type 163.

Name	Operation	Operands
name	LMSAS163	[P=prefix]

prefix	Up to 3 characters to be prefixed to the field names. By default, the field names have no prefix.

Expansion of LMSAS163

```
**********************************************************************
*                                                                    *
*        RECORD TYPE :  1 6 3  (DSECT)                               *
*                                                                    *
**********************************************************************
         SPACE
&NAME    DSECT
&P.163LL DS    H                 RECORD LENGTH                       H'0'
         DC    AL1(0)                                                AL1(0)
&P.163RT DC    AL1(163)          RECORD TYPE                         AL1(163)
&P.SECNA DC    CL32' '           SECONDARY NAME : 1 - 32             CL32' '
&P.SECAT DC    CL8' '            SECONDARY ATTRIBUTE : 0 - 8         CL8' '
*              '0       ' :  -   CSECT
*              '1       ' :  -   ENTRY
&P.SFIND DC    AL1(0)            FORMAT INDICATOR                    AL1(0)
&P.SNAML DC    CL1' '            LONG SEC NAME : -  32K-45           CL1' '
&P.163LG EQU   *-&P.163LL
```

Format of a record of type 164

The LMSAS164 record is described below as an example of a record of type 164.

Name	Operation	Operands
name	LMSAS164	[P=prefix]

prefix	Up to 2 characters to be prefixed to the field names. By default, the field names have no prefix.

Expansion of LMSAS164

```
***********************************************************************
*                                                                     *
*        RECORD TYPE :  1 6 4  (DSECT)                                *
*                                                                     *
***********************************************************************
         SPACE
&NAME    DSECT
&P.RECLEN DS    H                LENGTH OF PLAM RECORD              H'540'
         DC    AL1(0)                                               AL1(0)
&P.RECID  DC    AL1(164)         IDENTIFICATION OF PLAM RECORD      AL1(164)
&P.VERS   DC    AL1(2)           VERSION OF SPECIFIED PLAM RECORD   AL1(2)
&P.RECNUM DS    AL1              RECORD DESCRIBES FORMAT            AL1(1)
*                                OF PLAM RECORD OF RECORD TYPE
*                                WITH SPECIFIED NUMBER
&P.FNAME  DS    CL54             FILE NAME         TAKEN FROM FCB   CL54' '
&P.FTYPE  DS    X                FCBTYPE (SET/RESET)             )  X'00'
&P.FTYPES EQU   X'C0'              R SAM                         )
&P.FTYPEI EQU   X'40'              S ISAM                        )
&P.FTYPEP EQU   X'C0'              S PAM                         )
&P.VMIN   EQU   X'01'              R VALPROP MIN. FUNCT.         )
&P.VMAX   EQU   X'01'              S VALPROP MAX. FUNCT.         )
&P.SHARE  DS    X                SHARE (SET,RESET)             CAT  X'00'
&P.SHAREY EQU   X'04'              S YES                         )
&P.ACCESS EQU   X'08'              S ACCESS=READ                 )
&P.SHCCNO EQU   X'C0'              R NO CONTROL CHAR             )
&P.SHCCM  EQU   X'40'              S MACHINE CODE CONTROL CHAR   )
&P.SHCCA  EQU   X'C0'              S ASA CONTROL CHAR            )
&P.SIZE   DS    XL3              FILE SIZE <32GB                 )  XL3'00'
&P.SALL   DS    XL2              SECONDARY ALLOCATION            )  XL2'00'
&P.RECF   DS    X                RECFORM (SET,RESET)             )  X'00'
&P.RECFF  EQU   X'04'              S FIXED                       )
&P.RECFV  EQU   X'02'              S VARIABLE                    )
&P.RECFU  EQU   X'06'              S UNDEFINED                   )
&P.BLKSIZ DS    H                BLKSIZE                         )  H'0'
&P.RECSIZ DS    H                RECSIZE                         )  H'0'
```

```
&P.KEYPOS DS    H                 KEYPOS                          )     H'0'
&P.KEYLEN DS    X                 KEYLEN                          )     X'00'
&P.PAD    DS    X                 PAD                           FCB     X'00'
&P.LOGLN  DS    X                 LOGLEN                          )     X'00'
&P.VALLN  DS    X                 VALLEN                          )     X'00'
&P.KEY    DS    C                 DOES KEY EXISTS IN MEMBER ?           C' '
&P.KEYY   EQU   C'Y'                YES
&P.KEYN   EQU   C'N'                NO
&P.CFID   DS    XL4               CFID                            )     XL4'00'
&P.CTRLI  DS    X                 BLKCTRL-INDICATOR               )     X'00'
&P.CTRLN  EQU   X'80'               S BLKCTRL=NO                  )
&P.CTRLP  EQU   X'40'               S BLKCTRL=PAMKEY              )
&P.CTRLD  EQU   X'20'               S BLKCTRL=DATA                )
&P.CTRLO  EQU   X'10'               S BLKCTRL=NULL                )
&P.CTRLR  EQU   X'F0'               R BLKCTRL=NOT SPECIFIED       )
&P.BCF4K  EQU   X'08'               S BLOCK CONTROL FIELD 4K      )
&P.BCF2K  EQU   X'04'               S BLOCK CONTROL FIELD 2K      )
&P.CTRLU  EQU   X'03'               R -- RESERVED, MUST BE 0 --   )
*
&P.PERF   DS    X                 IOPERF-INDICATOR                )     X'00'
&P.PFVH   EQU   X'03'               S  IOPERF=VERY-HIGH           )
&P.PFHI   EQU   X'02'               S  IOPERF=HIGH                )
&P.PFST   EQU   X'01'               S  IOPERF=STD                 )
&P.PFNS   EQU   X'00'               S  IOPERF NOT SPECIFIED       )
&P.USAG   DS    X                 IOUSAGE-INDICATOR               )     X'00'
&P.USRW   EQU   X'03'               S  IOUSAGE=RDWRT              )
&P.USWR   EQU   X'02'               S  IOUSAGE=WRITE              )
&P.USRD   EQU   X'01'               S  IOUSAGE=READ               )
&P.USNS   EQU   X'00'               S  IOUSAGE NOT SPECIFIED      )
&P.EDMS3  DS    X                 CATALOG-INDIC                   )     X'00'
&P.ESPEC  EQU   X'08'               S  PLAM FILE INDICATOR        )
*
          DS    X                 FREE ( 0 )                      )     X'00'
&P.AIXCNT DS    H                 ALTERNATE INDEX COUNT           )     H'0'
&P.AIXMAX EQU   30                  MAX. NR. OF AIX ENTRIES       )
&P.FSIZ   DS    XL4               FILE SIZE >= 32GB,
*                                 &P.SIZE MUST BE X‘FFFFFF‘
          DS    CL86              RESERVED, MUST BE 0                   XL86'00'
*
&P.AIXNAM DS    CL8               KEYNAME                         )     XL8'00'
&P.AIXKPO DS    H                 KEYPOS                          )     H'0'
&P.AIXKLE DS    AL1               KEYLEN                          )     AL1(0)
&P.AIXIND DS    XL1               INDICATOR                       )     XL1'0'
&P.AIXIDK EQU   X'80'               S  DUPKEY=YES                 )
*                                   R  DUPKEY=NO                  )
&P.AIX#   EQU   *-&P.AIXNAM       LENGTH OF AIX ENTRY             )
          DS    (&P.AIXMAX-1)CL(&P.AIX#)                                 29XL12'00'
*
&P.LEN    EQU   *-&P.RECLEN        LENGTH OF PLAM RECORD
```

7.4 Example

The following Assembler program contains the functions listed below:

– Open a subroutine access (INIT)

– Incorporate a file as a member (ADD)

– Search for a member in a directory (TOCPRIM)

– Open a member (OPENGET)

– Read the first record (GET)

– Close the member (CLOSE)

– Terminate the subroutine access (END)

To make the example easier to understand, it includes comments in the form of messages

```
***********************************************************************
*                                                                     *
*                   EXAMPLE OF LMS AS A SUBROUTINE                    *
*                                                                     *
***********************************************************************
*
LMSUP    CSECT
R1       EQU   1                          ADDRESS OF THE PARAMETER LIST
R2       EQU   2                          TEMPORARY WORK REGISTER
R3       EQU   3                          TEMPORARY WORK REGISTER
R4       EQU   4                          TEMPORARY WORK REGISTER
R5       EQU   5                          TEMPORARY WORK REGISTER
R10      EQU   10                         BASE REGISTER
R11      EQU   11                         BASE REGISTER
R13      EQU   13                         ADDRESS OF SAVE AREA
R14      EQU   14                         RETURN ADDRESS
R15      EQU   15                         ENTRY ADDRESS
         LMSASSEQ
*
         BALR  R10,0
         USING *,R10,R11
BASIS    LA    R11,BASIS+4095             2. BASE REGISTER
         LA    R11,1(R11)
         LA    R13,SAVEAREA
```

```
***********************************************************************
*                                                                     *
*          CALLING INIT                                               *
*                                                                     *
***********************************************************************
*
*          ADDRESS LMSASSCB TO FIRST WORD OF PARAMETER LIST
*
         LA    R1,SBCB
         ST    R1,PARAM1
*
*          PREPARE CB FOR INIT CALL
*
         MVC   SBCB(CBPLNG),DEFCB
*
*          INIT FUNCTION AND SUBCODE UNUSED SET IMPLICITLY
*
         LA    R1,PARAM
         L     R15,=V(LMSUP1)
         BALR  R14,R15
*
*          EVALUATE RETURN CODE
*
INITCL   CLI   CBRTC,OK
         BNE   RCPROC
*
***********************************************************************
*                                                                     *
*          CALLING ADD                                                *
*                                                                     *
***********************************************************************
*
*          PREPARE CB FOR ADD CALL
*
         MVI   CBFUNC,ADD                  FUNCTION CODE
         MVI   CBSUBC,UNUSE                SUBCODE UNUSED (DEFAULT)
         MVI   CBOV,YES                    OVERWRITE=YES
*
*          ALL OTHER FIELDS SAME AS FOR INIT
*
*          PREPARE FD FOR ADD CALL
*
         MVC   SBFD(FDPLNG),DEFFD          DEFINE FD AREA
         MVC   FDLINK,FILELINK             LINK NAME TO FD
*
*          PREPARE LD FOR ADD CALL
*
         MVC   SBLD(LDPLNG),DEFLD          DEFINE LD AREA
         MVC   LDLINK,LIBLINK              LINK NAME TO LD
*
*          PREPARE ED FOR ADD CALL
*
```

```
         MVC   SBED(EDPLNG),DEFED         DEFINE ED AREA
         MVI   EDTYPE,'S'                 STORAGE TYPE S
         MVC   EDNAME(9),ELNAME           MEMBER NAME
         MVC   EDVERS(1),ELVERS           MEMBER VERSION
*
*           SUPPLY PARAMETER LIST; PARAM1 CONTAINS A(CB)
*
         LA    R1,SBFD                    A(FD)
         ST    R1,PARAM2
         LA    R1,SBLD                    A(LD)
         ST    R1,PARAM3
         LA    R1,SBED                    A(ED)
         ST    R1,PARAM4
*
         LA    R1,PARAM
         L     R15,=V(LMSUP1)
         BALR  R14,R15
*
*           EVALUATE RETURN CODE
*
ADDCL    CLI   CBRTC,OK
         BNE   RCPROC
*
***********************************************************************
*                                                                     *
*           CALLING TOC                                               *
*                                                                     *
***********************************************************************
*           PREPARE CB FOR TOCPRIM CALL
*
         MVI   CBFUNC,TOCP                FUNCTION CODE
         MVI   CBSUBC,LONG                EXTENDED MEMBER INFO
*
*           LD FIELDS AS PREDEFINED
*
*           PREPARE EM FOR TOCPRIM CALL
*
         LA    R2,SBEM                    TARGET ADDRESS
         LA    R3,EMPLNG                  LENGTH OF TRANSFER
         LA    R4,DEFEM                   SOURCE ADDRESS
         LR    R5,R3
         MVCL  R2,R4
*
         MVI   EMTYPE,'S'
         MVC   EMNAME(9),ELNAME
*
*           SUPPLY PARAMETER LIST; PARAM1 CONTAINS A(CB)
*
```

```
         LA    R1,SBTID                        A(TID)
         ST    R1,PARAM2
         LA    R1,SBEI                         A(EI)
         ST    R1,PARAM3
         LA    R1,SBLD                         A(LD)
         ST    R1,PARAM4
         LA    R1,SBEM                         A(EM)
         ST    R1,PARAM5
*
         LA    R1,PARAM
         L     R15,=V(LMSUP1)
         BALR  R14,R15
*
*           EVALUATE RETURN CODE
*
TOCCL    CLI   CBRTC,OK
         BNE   RCPROC
*
*           OUTPUT RESULTING INFO
*
         MVC   OTYPF,EITYPE
         WROUT OTYP,TERM
*
         MVC   ONAMEF,EINAME
         WROUT ONAME,TERM
*
         MVC   OVERSF,EIVERS
         WROUT OVERS,TERM
*
         MVC   ODATEF,EIUDAT
         WROUT ODATE,TERM
*
```

```
***********************************************************************
*                                                                     *
*           CALLING OPENGET                                           *
*                                                                     *
***********************************************************************
*           PREPARE CB FOR OPENGET CALL
*
         MVI   CBFUNC,OPENG               FUNCTION CODE
         MVI   CBSUBC,UNUSE               SUBCODE UNUSED (DEFAULT)
*
*           LD FIELDS AS PREDEFINED
*
*           ED FIELDS AS PREDEFINED
*
*           PREPARE RD FOR OPENGET CALL
*
         MVC   SBRD(RDPLNG),DEFRD         DEFINE RD AREA
*
*           SUPPLY PARAMETER LIST; PARAM1 CONTAINS A(CB)
*
         LA    R1,SBRD                    A(RD)
         ST    R1,PARAM2
         LA    R1,SBLD                    A(LD)
         ST    R1,PARAM3
         LA    R1,SBED                    A(ED)
         ST    R1,PARAM4
*
         LA    R1,PARAM
         L     R15,=V(LMSUP1)
         BALR  R14,R15
*
*           EVALUATE RETURN CODE
*
OPENGCL  CLI   CBRTC,OK
         BNE   RCPROC
*
```

```
***********************************************************************
*                                                                     *
*           CALLING GET, READING RECORDS IN LOOP                      *
*                                                                     *
***********************************************************************
*           PREPARING CB FOR GET CALL
*
GETLOOP  MVI   CBFUNC,GET                FUNCTION CODE
         MVI   CBSUBC,SEQ                SEQUENTIAL READING
*
*           PREPARE RD FOR GET CALL
*
         MVC   RDBLEN,PLENGTH            BUFFER LENGTH OF INPUT AREA
*
*           SUPPLY PARAMETER LIST; PARAM1 CONTAINS A(CB)
*
*           A(RD) WAS ALREADY SUPPLIED WITH OPENGET
*
         LA    R1,SBER                   A(ER)
         ST    R1,PARAM3
*
         LA    R1,PARAM
         L     R15,=V(LMSUP1)
         BALR  R14,R15
*
*           EVALUATE RETURN CODE / OUTPUT RECORD
*
GETCL    CLI   CBRTC,OK
         BNE   GETEND
*
*           RECORD OUTPUT WITHOUT COLUMN 1 (CONTROL CHARACTER)
*
         WROUT SBER,TERM
         B     GETLOOP
*
*           EVALUATE END OF ELEMENT
*
GETEND   CLI   CBRTC,EOF
         BNE   RCPROC
*
```

```
***********************************************************************
*                                                                     *
*           CALLING CLOSE                                             *
*                                                                     *
***********************************************************************
*           PREPARE CB FOR CLOSE CALL
*
         MVI   CBFUNC,CLOSE               FUNCTION CODE
         MVI   CBSUBC,UNUSE               SUBCODE UNUSED (DEFAULT)
*
*           RD FIELDS AS PREDEFINED
*
*           SUPPLY PARAMER LIST; PARAM1 CONTAINS A(CB)
*
*           A(RD) WAS ALREADY SUPPLIED WITH OPENGET
*
         LA    R1,PARAM
         L     R15,=V(LMSUP1)
         BALR  R14,R15
*
*           EVALUATE RETURN CODE
*
CLOSECL  CLI   CBRTC,OK
         BNE   RCPROC
```

```
**********************************************************************
*                                                                    *
*           CALLING END                                              *
*                                                                    *
**********************************************************************
*           PREPARE CB FOR END CALL
*
         MVI   CBFUNC,END                 FUNCTION CODE
         MVI   CBSUBC,UNUSE               SUBCODE UNUSED (DEFAULT)
*
         LA    R1,PARAM
         L     R15,=V(LMSUP1)
         BALR  R14,R15
*
*           EVALUATE RETURN CODE
*
ENDCL    CLI   CBRTC,OK
         BNE   RCPROC
         TERM
*
**********************************************************************
*                                                                    *
*           ERROR HANDLING                                           *
*                                                                    *
**********************************************************************
RCPROC   EQU   *
         WROUT MESSAGE,ERROR
*
FEHLER   EQU   *
TERM     TERM
```

```
**********************************************************************
*                                                                    *
*           DEFINING CONSTANTS                                       *
*                                                                    *
**********************************************************************
DEFCB    LMSASSCB MF=L                   CONSTANTS FOR CB
*
DEFFD    LMSASSFD MF=L                   CONSTANTS FOR FD
*
DEFLD    LMSASSLD MF=L                   CONSTANTS FOR LD
*
DEFED    LMSASSED MF=L                   CONSTANTS FOR ED
*
DEFEI    LMSASSEI MF=L                   CONSTANTS FOR EI
*
DEFEM    LMSASSEM MF=L                   CONSTANTS FOR EM
*
DEFRD    LMSASSRD MF=L                   CONSTANTS FOR RD
*
FILELINK DC    'FILELINK'
LIBLINK  DC    'LIBLINK '
PLENGTH  DC    A(L'SBER)
ELNAME   DC    'PROBEELEM'
ELVERS   DC    '1'
*
MESSAGE  DC    Y(MESSAGEE-MESSAGE)
         DS    CL2
         DC    X'40'
         DC    'FUNCTION ERRONEOUS'
MESSAGEE EQU   *
*
```

```
**********************************************************************
*                                                                    *
*           STORAGE AREAS                                            *
*                                                                    *
**********************************************************************
SBCB     LMSASSCB MF=C                STORAGE AREA FOR CB
*
SBFD     LMSASSFD MF=C                STORAGE AREA FOR FD
*
SBLD     LMSASSLD MF=C                STORAGE AREA FOR LD
*
SBED     LMSASSED MF=C                STORAGE AREA FOR ED
*
SBEI     LMSASSEI MF=C                STORAGE AREA FOR EI
*
SBEM     LMSASSEM MF=C                STORAGE AREA FOR EM
*
SBRD     LMSASSRD MF=C                STORAGE AREA FOR RD
*
SBER     DS    CL256                  RECORD BUFFER
SBTID    DC    F'1'                   TOC IDENTIFICATION
*
OTYP     DC    Y(OTYPE-OTYP)          FOR TYPE OUTPUT
         DS    CL2
         DC    X'40'
         DC    'TYPE          '
OTYPF    DC    CL(L'EITYPE)' '
OTYPE    EQU   *
*
ONAME    DC    Y(ONAMEE-ONAME)        FOR NAME OUTPUT
         DS    CL2
         DC    X'40'
         DC    'NAME          '
ONAMEF   DC    CL(L'EINAME)' '
ONAMEE   EQU   *
*
OVERS    DC    Y(OVERSE-OVERS)        FOR VERSION OUTPUT
         DS    CL2
         DC    X'40'
         DC    'VERSION       '
OVERSF   DC    CL(L'EIVERS)' '
OVERSE   EQU   *
*
ODATE    DC    Y(ODATEE-ODATE)        FOR DATE OUTPUT
         DS    CL2
         DC    X'40'
         DC    'USER-DATE    '
ODATEF   DC    CL(L'EIUDAT)' '
ODATEE   EQU   *
```

```
************************************************************************
*                                                                      *
*           PARAMETER LIST                                             *
*                                                                      *
************************************************************************
PARAM    DS    0F
PARAM1   DS    F                             A(LMSASSCB)
PARAM2   DS    F
PARAM3   DS    F
PARAM4   DS    F
PARAM5   DS    F
PARAM6   DS    F
************************************************************************
*                                                                      *
*           SAVE AREA                                                  *
*                                                                      *
************************************************************************
SAVEAREA DS    18F
************************************************************************
         END
```

Related publications

You will find the manuals on the internet at *http://manuals.ts.fujitsu.com*. You can order printed copies of those manuals which are displayed with an order number.

[1] **LMS**
(BS2000/OSD)
SDF Format
User Guide

[2] **XHCS**
(BS2000/OSD)
8-Bit Code Processing in BS2000/OSD
User Guide

Index