

# ASG-DataManager™ Mark IV Interface

Version: 2.5

Publication Number: DMR0200-25-MKIV

Publication Date: June 1983

The information contained herein is the confidential and proprietary information of Allen Systems Group, Inc. Unauthorized use of this information and disclosure to third parties is expressly prohibited. This technical publication may not be reproduced in whole or in part, by any means, without the express written consent of Allen Systems Group, Inc.

© 1998-2001 Allen Systems Group, Inc. All rights reserved.

All names and products contained herein are the trademarks or registered trademarks of their respective holders.



ASG Worldwide Headquarters Naples, Florida USA | [asg.com](http://asg.com)

1333 Third Avenue South, Naples, Florida 34102 USA Tel: 941.435.2200 Fax: 941.263.3692 Toll Free: 1.800.932.5536







# ASG Support Numbers

ASG provides support throughout the world to resolve questions or problems regarding installation, operation, or use of our products. We provide all levels of support during normal business hours and emergency support during non-business hours. To expedite response time, please follow these procedures.

## **Please have this information ready:**

- Product name, version number, and release number
- List of any fixes currently applied
- Any alphanumeric error codes or messages written precisely or displayed
- A description of the specific steps that immediately preceded the problem
- The severity code (ASG Support uses an escalated severity system to prioritize service to our clients. The severity codes and their meanings are listed below.)

## **If You Receive a Voice Mail Message:**

- 1 Follow the instructions to report a production-down or critical problem.
- 2 Leave a detailed message including your name and phone number. A Support representative will be paged and will return your call as soon as possible.
- 3 Please have the information described above ready for when you are contacted by the Support representative.

## **Severity Codes and Expected Support Response Times**

Severity	Meaning	Expected Support Response Time
1	Production down, critical situation	Within 30 minutes
2	Major component of product disabled	Within 2 hours
3	Problem with the product, but customer has work-around solution	Within 4 hours
4	"How-to" questions and enhancement requests	Within 4 hours

ASG provides software products that run in a number of third-party vendor environments. Support for all non-ASG products is the responsibility of the respective vendor. In the event a vendor discontinues support for a hardware and/or software product, ASG cannot be held responsible for problems arising from the use of that unsupported version.

## ***Business Hours Support***

<b>Your Location</b>	<b>Phone</b>	<b>Fax</b>	<b>E-mail</b>
<b>United States and Canada</b>	800.354.3578 1.941.435.2201 <b>Secondary Numbers:</b> 800.227.7774 800.525.7775	941.263.2883	support@asg.com
<b>Australia</b>	61.2.9460.0411	61.2.9460.0280	support.au@asg.com
<b>England</b>	44.1727.736305	44.1727.812018	support.uk@asg.com
<b>France</b>	33.141.028590	33.141.028589	support.fr@asg.com
<b>Germany</b>	49.89.45716.300	49.89.45716.400	support.de@asg.com
<b>Singapore</b>	65.224.3080	65.224.8516	support.sg@asg.com
<b>All other countries:</b>	1.941.435.2201		support@asg.com

## ***Non-Business Hours - Emergency Support***

<b>Your Location</b>	<b>Phone</b>	<b>Your Location</b>	<b>Phone</b>
<b>United States and Canada</b>	800.354.3578 1.941.435.2201 <b>Secondary Numbers:</b> 800.227.7774 800.525.7775 <b>Fax:</b> 941.263.2883		
<b>Asia</b>	011.65.224.3080	<b>Japan/Telecom</b>	0041.800.9932.5536
<b>Australia</b>	0011.800.9932.5536	<b>New Zealand</b>	00.800.9932.5536
<b>Denmark</b>	00.800.9932.5536	<b>South Korea</b>	001.800.9932.5536
<b>France</b>	00.800.9932.5536	<b>Sweden/Telia</b>	009.800.9932.5536
<b>Germany</b>	00.800.9932.5536	<b>Switzerland</b>	00.800.9932.5536
<b>Hong Kong</b>	001.800.9932.5536	<b>Thailand</b>	001.800.9932.5536
<b>Ireland</b>	00.800.9932.5536	<b>United Kingdom</b>	00.800.9932.5536
<b>Israel/Bezeq</b>	014.800.9932.5536		
<b>Japan/IDC</b>	0061.800.9932.5536	<b>All other countries</b>	1.941.435.2201

## ASG Web Site

Visit <http://www.asg.com>, ASG's World Wide Web site.

Submit all product and documentation suggestions to ASG's product management team at <http://www.asg.com/products/suggestions.asp>

If you do not have access to the web, FAX your suggestions to product management at (941) 263-3692. Please include your name, company, work phone, e-mail ID, and the name of the ASG product you are using. For documentation suggestions include the publication number located on the publication's front cover.



## PREFACE

This manual is one of a series describing DATAMANAGER, the data dictionary software developed by MSP for use on IBM System/360, System/370, 30xx and 4300 series, and plug compatible, machines. The manual describes the MARK IV Interface facility, which, in combination with the Source Language Generation facility (in addition to the basic DATAMANAGER set up, maintenance, and interrogation capabilities), enables the user:

- to include MARK IV file and segment data definitions in the data dictionary and
- to produce MARK IV File Definition forms directly from DATAMANAGER data definitions.

The MARK IV Interface facility is also applicable in conventional file and in IMS (DL/I) database environments.

This first definitive edition of the DATAMANAGER MARK IV Interface manual supersedes the second provisional edition together with all Amendment Lists and User Notices issued during the currency of that edition. All MARK IV Interface User Notices up to and including notice number 4 are therefore cancelled. This edition is relevant to all MARK IV releases up to and including Release 7.0. With Amendment Lists 1 and 2 incorporated it relates to DATAMANAGER Release 4.0 and subsequent releases.

It is assumed that the reader has a knowledge of DATAMANAGER to the extent covered by the User's Guide and the Source Language Generation manual, and is familiar with the MARK IV File Management System.

Chapter 1 of this manual summarises the interface between DATAMANAGER and MARK IV.

Chapter 2 discusses briefly the concept of MARK IV files and segment hierarchies and provides an example of their definition to the data dictionary.

Chapter 3 states the specifications of the DATAMANAGER data definition statements for MARK IV files and segments.

Chapter 4 describes the interface between the DATAMANAGER Source Language Generation facility and MARK IV, in particular providing those specifications of the PRODUCE command which enable DATAMANAGER to generate MARK IV File Definition forms from the data dictionary, along with a sample listing of output from a PRODUCE MARKIV command.

Chapter 5 tabulates the relationships which exist between the entries of a MARK IV File Definition form and the DATAMANAGER data definitions from which they are generated.

PREFACE  
(continued)

Appendix 1 describes the macro DGMIV, which can be used to tailor some of the standard MARK IV Interface options to meet an installation's particular requirements.

For the storage and job control requirements for installing and running DATAMANAGER with the MARK IV Interface, reference should be made to the Installation in OS Environments manual or Installation in DOS Environments manual, as appropriate.

Other manuals in the DATAMANAGER series are the User's Guide; the Messages Manual, in which all diagnostic messages that can be output by DATAMANAGER are listed; the Controller's Manual, circulation of which is restricted to those responsible for the administration of a data dictionary; and a separate manual for each of the separate DATAMANAGER facilities (Audit and Security, Automation of Set Up, Source Language Generation User Defined Syntax, various interface facilities, etc.).

CONTENTS

PREFACE . . . . . iii

NOTATION FOR STATEMENT FORMATS . . . . . vi

CHAPTER 1 DATAMANAGER/MARK IV INTERFACE FACILITIES . 1-1

CHAPTER 2 MARK IV DEFINITIONS AND DATAMANAGER . . . . 2-1

2.1 Overview . . . . . 2-1

2.2 Further Considerations . . . . . 2-5

CHAPTER 3 DATAMANAGER DATA DEFINITION STATEMENTS FOR  
A MARK IV FILE MANAGEMENT ENVIRONMENT . . . 3-1

3.1 Introduction . . . . . 3-1

3.2 The FILE Data Definition Statement  
for a MARK IV File . . . . . 3-2

3.3 The MARKIV-SEGMENT Data Definition  
Statement . . . . . 3-7

3.4 MARK IV Related Aspects of the ITEM  
Data Definition Statement . . . . . 3-9

CHAPTER 4 GENERATION OF MARK IV FILE DEFINITION  
FORMS FROM DATAMANAGER . . . . . 4-1

4.1 Introduction . . . . . 4-1

4.2 Specification of the PRODUCE Command  
for MARK IV File Definition Form  
Generation . . . . . 4-3

4.3 Supporting Information . . . . . 4-16

4.4 Example Output . . . . . 4-21

CHAPTER 5 MARK IV/DATAMANAGER CORRESPONDENCE TABLES . 5-1

APPENDIX 1 THE MACRO DGMIV . . . . . App.1-1

App.1.1 Implementation of the Macro DGMIV . . . App.1-1

App.1.2 DGMIV Macro Keyword Definitions . . . App.1-1

AMENDMENT RECORD . . . . . AR-1

NOTATION FOR  
STATEMENT  
FORMATS

In all manuals relating to DATAMANAGER, the following notation is used in the specification of statement formats (for commands and data definition statements):

- all words printed in capitals are statement identifiers or keywords that must be present in full or truncated form in the circumstances stated in the statement specification. The extent beyond which a word must not be truncated is indicated by underlining of the characters that must be retained.
- all words printed in lower case are variables for which the user must substitute a value consistent with the specification
- material enclosed in square brackets [ ] is an option which may be included or omitted as required
- braces { } indicate that a choice must be made of one of the options enclosed within them
- three full stops ... indicate that the material they immediately follow may be repeated. Where ... immediately follows a closing square bracket or brace, the material that can be repeated is bounded by that square bracket or brace and the corresponding opening square bracket or brace. If material can be repeated only a limited number of times, the repetition permitted is stated in the specification.
- other punctuation marks and symbols must be coded as shown, subject to the implications of any square brackets or braces enclosing them; except that where a single quote, ', is shown, a double quote, ", can alternatively be used, provided that the opening and closing quotes of any pair of quotes are the same character (single quote or double quote).

DATAMANAGER's MARE IV Interface provides the following facilities for users in a MARE IV environment:

- the ability to define MARE IV files and segments to DATAMANAGER, to hold the definitions in the data dictionary, and to process them by the standard DATAMANAGER commands
- the ability to generate from the data dictionary, and to insert into the required source library, sets of File Definition forms, for one or more MARE IV files, ready for direct input to the MARE IV File Management System.

The ability to define a MARE IV segment requires the new DATAMANAGER member type, MARKIV-SEGMENT, which in the member type hierarchy comes between FILE and GROUP. Also required are specific modifications of the FILE data definition statement, to cater for MARE IV files. The MARKIV-SEGMENT data definition statement and the relevant formats of the FILE data definition statement are further discussed in Chapter 2 and specified in Chapter 3.

So that the definitions of MARE IV files and segments can be processed by DATAMANAGER in the same way as other members of the data dictionary, the keywords MARKIV-FILES and MARKIV-SEGMENTS are added to the member-type keywords available for use in the following DATAMANAGER commands (see User's Guide for definitions):

- BULK
- GLOSSARY
- LIST
- PERFORM
- REPORT
- WHICH.

In addition, the keywords FATHERS and FATHERED-BY are available for use in the relation clause of the WHICH and WHAT commands, and the keywords FATHERS, PARENTS and COUNT-FIELD are available within the VIA clause of the WHICH and WHAT commands. These keywords are additional to those described in the specifications of the WHICH and WHAT commands in section 3.2 of the User's Guide. The keywords FATHERS and PARENTS can also be used in VIA interrogations with IMS and System 2000/80. The following points regarding the use of these keywords should be noted:

1  
(continued)

- within the VIA clause, the keyword FATHERS has the same meaning as the keyword PARENTS-
- the condition FATHERS member-name is equivalent to USES member-name VIA PARENTS
- the condition FATHERED-BY member-name is equivalent to CONSTITUTES member-name VIA PARENTS

The ability to generate File Definition forms from data dictionary members requires the use of the Source Language Generation facility, which is described in a separate manual; The Source Language Generation manual describes the basic version of the facility, which can, via the PRODUCE command, output data descriptions in COBOL, PL/I, or Assembler. The enhancements to the facility to enable it to generate MARE IV File Definition forms are discussed in Chapter 4 of this manual. Tables are presented in Chapter 5 indicating the relationships that exist between the entries of a File Definition form and the DATAMANAGER data definitions used to generate them.

Since DATAMANAGER is able to hold IMS database data definitions as members of the data dictionary, the ability of MARE IV to access these databases is reflected in the DATAMANAGER/MARK IV Interface facility. Throughout this manual, any mention of the member type IMS-DATABASE should be construed as including the synonymous member type DL/I-DATABASE.

2.1

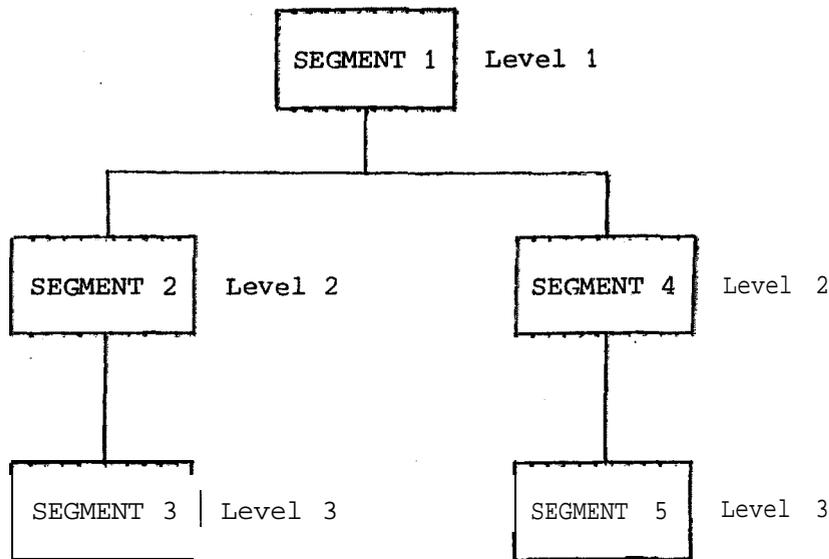
OVERVIEW

The ability provided by DATAMANAGER to define item, group, segment, file, and database members to the data dictionary permits generation of MARK IV File Definition forms through use of the **DATAMANAGER/MARK** IV Interface facility.

In the MARK IV File Management System, it is not the physical record structure of a file that is significant, but rather the logical structure of the fields within a record as indicated for specific applications. The significant structural element in the MARK IV approach to defining a logical file is the segment, a collection of one or more fields from a physical record. One or more distinct, but hierarchically related, segments may be defined from (some or all of) the fields of a physical record to form a logical record.

In the simplest case, a logical file will consist of **single-segment fixed length records**, but structures can be rather more complex. MARK IV provides the ability to define and process hierarchical record structures, where the logical record defined contains a nesting of distinct segments, with multiple occurrence of lower level (in the hierarchy) segments permitted, either a fixed or a variable number of times in a logical record. MARK IV uses level numbers and segment numbers to describe such a hierarchical structure, where the level one segment is referred to as the base segment of the logical record; for example:

2.1  
(continued)



Up to nine levels of nesting are permitted in a hierarchy for a logical record and up to 99 distinct segments, each able to occur as many times as required. Level numbers are assigned consecutively with the base segment at level 1. Segment numbers are assigned hierarchically, in ascending order from top to bottom and left to right (as in the example above) but need not be consecutive.

To control the occurrences of variably repeating segments, MARK IV maintains count fields, defined by the user, which contain, for each logical record, the number of occurrences of subordinate segments. The count field for a variably repeating segment is kept in the segment at the next higher level than that of the repeating segment which it controls.

To take a particular example (from the MARK IV Reference Manual), consider the three-level five-segment hierarchy shown above as a logical record of a personnel file where the base segment is the employee segment, the level two segments contain data describing the employee's education (segment 2) and employment history (segment 4), and the corresponding level three segments describe the degrees (segment 3) obtained by the employee at each school and his references (segment 5) for each previous job. Since each of the level two segments can occur a variable number of times from employee to employee and, for a given employee, a level three segment can occur a variable number of times from occurrence to occurrence of its parent level two segment, count fields must be held in the appropriate higher level segments; that is, for a given logical record:

- the segment (named) Employee will include the count fields for the (variable) number of occurrences of the segment School and the segment Employer

2.1  
(continued)

- each occurrence of the segment School will include a count field for the segment Degree
- each occurrence of the segment Employer will include a count field for the segment Reference.

Figure 2.1 indicates the occurrences of the above segments for a typical logical record of the personnel file, where:

- the assumed fields of each segment are named,
- illustrative specific values have been assigned to the count fields (**thus determining** the number of occurrences of lower level segments), and
- an assumed key field is indicated in each segment.

The following DATAMANAGER data definitions could be inserted into the data dictionary (by INSERT or ADD commands! see DATAMANAGER User's Guide, section 3.2) to define the MARK IV personnel file described in the example above:

<u>Member Name</u>	<u>Data Definition</u>
PERSONNEL	FILE MARK-IV HELD-AS CONTAINS EMPLOYEE, SCHOOL PARENT EMPLOYEE COUNT-FIELD SCHCOUNT, DEGREE PARENT SCHOOL COUNT-FIELD DEGCOUNT, EMPLOYER PARENT EMPLOYEE COUNT-FIELD JOBCOUNT, REFRENCE PARENT EMPLOYER COUNT-FIELD REFCOUNT ;
EMPLOYEE	MARKIV-SEGMENT HELD-AS KEYS EMPLNO CONTAINS <b>EMPLNO, NAME, JOBTITLE, CLASS, DIVISION, DEPT, GROUP, HIREDATE, SALARY, DEPENDTS, SOCSECNO, JOBS, SCHCOUNT, JOBCOUNT</b> ;
SCHOOL	MARKIV-SEGMENT HELD-AS KEYS <b>SCHNAME</b> CONTAINS <b>SCHNAME, SCHADDR, YEARS, HONORS, DEGCOUNT, REMARKS</b> ;
DEGREE	MARKIV-SEGMENT HELD-AS KEYS DEGTYPE CONTAINS <b>DEGTYPE, GRDPTAVG, DATE</b> ;

EMPL. NO.	NAME	JOB TITLE	CLASS	DIVISION	DEPT
key field					
GROUP	HIRE DATE	DEPEND.	SOC. SEC.#	COUNT FIELDS	
				School	Jobs
				2	1

Level 1  
Segment  
EMPLOYEE

SCHOOL#1	ADDRESS	YEARS	HONORS	DEGREE COUNT	REMARKS
key field				1	

Level 2  
Segment  
SCHOOL

DEGREE	GPA	DATE
key field		

Level 3  
Segment  
DEGREE

SCHOOL#2	ADDRESS	YEARS	HONORS	DEGREE COUNT	REMARKS
key field				0	

Level 2  
Segment  
SCHOOL

PREVIOUS EMPLOYER#1	ADDRESS	TITLE	REF. COUNT	SALARY
			2	
TERM. DATE	START DATE			
	key field			

Level 2  
Segment  
EMPLOYER

REFERENCE#1	TITLE	YEARS KNOWN	EVALUATION
key field			

Level 3  
Segment  
REFERENCE

REFERENCE#2	TITLE	YEARS KNOWN	EVALUATION
key field			

Figure 2.1 Example of a Three-Level Variable Hierarchical Record

**2.1**  
(continued)

Member Name Data Definition

```
EMPLOYER      MARKIV-SEGMENT HELD-AS
              KEYS  STRTDATE
              CONTAINS
                EMPLNAME,EMPADDR,TITLE,REFCOUNT,SALARY,
                TERMDATE,STRTDATE
              ;

REFERENCE     MARKIV-SEGMENT  HELD-AS
              KEYS  REFNO
              CONTAINS
                REFNO,REFTITLE,YRSKNOWN,EVALTION
              ;
```

A MARK IV File Definition form generated from these data definitions by a PRODUCE command is listed in section 4.4.

**2.2**                    **FURTHER CONSIDERATIONS**

Although a MARK IV specific member type, MARKIV-SEGMENT, has been provided in DATAMANAGER to enable users to define a MARK IV file as containing MARK IV segments, DATAMANAGER also accepts the names of ordinary GROUP members and **ITEM** members. (MARK IV fields) in the CONTAINS clause of the **FILE** MARKIV data definition. When the Source Language Generation facility's PRODUCE command is used to generate MARK IV File Definition forms for an entire segment hierarchy from a FILE MARKIV member, any GROUP and ITEM members whose names appear in that member's CONTAINS clause are treated as though they were MARKIV-SEGMENT members. Furthermore, a PRODUCE command can be used to generate a File Definition form for a particular segment directly from such a GROUP or ITEM member, as well as from a MARKIV-SEGMENT member.

In contrast to the case of a FILE MARKIV member's data definition, DATAMANAGER accepts the names of GROUP and ITEM members in the CONTAINS clause of a MARKIV-SEGMENT member, but not the names of other MARKIV-SEGMENT members.

The full specifications of the MARKIV-SEGMENT data definition statement and of the MARK IV variant of the FILE data definition statement are given in Chapter **3**.



3.1

INTRODUCTION

To enable MARK IV files to be fully reflected in the data dictionary maintained by DATAMANAGER, the DATAMANAGER MARK IV Interface provides two additional member types:

- the member type FILE MARKIV (specified in section 3.2), in order to define a MARK IV file
- the member type MARKIV-SEGMENT (specified in section 3.3), in order to define the segments of a MARK IV file.

MARK IV segments may also be defined by GROUP and ITEM data definition statements (see User's Guide, section 4.2, for specifications). In particular, the ITEM statement specification includes optional keywords specific to the MARK IV environment, as described in section 3.4.

The syntax for the FILE MARKIV data definition statement is the same as for the data definition statement of a conventional file except that:

- the keyword MARKIV (or MARK-IV) must appear immediately after the statement identifier FILE
- the CONTAINS clause is declared in a different format (see section 3.2).

For the MARKIV-SEGMENT member type, except for the statement identifier, the syntax follows that of the member type GROUP.

## THE FILE DATA DEFINITION STATEMENT FOR A MARK IV FILE

Format

```

FILE { MARKIV
      MARK-IV } [ ENTERED-AS
                  HELD-AS
                  REPORTED-AS
                  DEFAULTED-AS ] [ ALIGNED
                                    UNALIGNED
                                    NOT-ALIGNED ]

[ CONTAINS segment-name
  [, segment-name PARENT segment-name-p
    [ COUNT-FIELD item-name-c [...] ]
    OCCURS occurrences ]

[ SEQUENTIAL
  INDEXED
  DIRECT
  KEYED
  VSAM
  PARTITIONED ] ... [ NO-LABELS
                       STANDARD-LABELS
                       USER-LABELS module-name ]

[ FIXED [block-size [record-size]]
  VARIABLE [maximum-block-size [maximum-record-size]]
  UNDEFINED [maximum-block-size]
  SPANNED [maximum-block-size [maximum-record-size]] ]

[ SORT-KEY { item-name
            group-name } [ ASCENDING
                          DESCENDING ]

  [, { item-name
      group-name } [ ASCENDING
                    DESCENDING ] ] ... ]

[ GENERATION-CYCLE integer ]

[ RETENTION-PERIOD integer [ 'string'
                             DAYS
                             MONTHS
                             YEARS ] ]

[ GROWTH-RATE integer [ PERCENT ] [ 'string'
                                     MONTHLY
                                     YEARLY ] ]

[ DEVICE { device [ number
                  'model' ] } 1 [ DENSITY integer ]
         { number
           'model' } ]

[ VOLUME 'name' ]

[ SIZE size [ KBYTES
              TRACKS
              CYLINDERS
              'string' ] [, size [ KBYTES
                                  TRACKS
                                  CYLINDERS
                                  'string' ] ] ]

```

3.2  
(continued)

[common clauses]

{ ; }  
{ . }

where

segment-name is the name of a dictionary member which is  
a MARKIV-SEGMENT, a GROUP, or an ITEM

segment-name-p is a segment-name which has been declared  
previously in the CONTAINS clause

item-name-c is the name of the ITEM held as a count field in  
segment-name-p and which contains the (usually variable)  
number of occurrences of segment-name in the hierarchy  
of segments which form the MARK IV file being defined

occurrences is an unsigned positive integer indicating the  
(fixed) number of occurrences of segment-name in the  
given segment hierarchy

module-name is the name of a module in which the file's user  
labels are defined and/or processed

block-size is an unsigned integer not greater than 32767,  
being the number of characters per block

record-size is an unsigned integer not greater than 32767,  
being the number of characters per record

maximum-block-size is an unsigned integer not greater than  
32767, being the maximum number of characters per block

maximum-record-size is an unsigned integer not greater than  
32767, being the maximum number of characters per record

item-name is the name of an item

group-name is the name of a group

integer is an unsigned integer not greater than 32767

string is a-character string of not more than 256 printable  
characters. A space character (hexadecimal 40)  
is regarded as a printable character.

device can be one of the following keywords:

<u>A</u> UDIO	<u>D</u> RUM	<u>P</u> RINTER
<u>C</u> ARDS	<u>M</u> ICR	<u>R</u> EMOTE
<u>C</u> ONSOLE	<u>O</u> CR	<u>T</u> APE
<u>D</u> ASD	<u>P</u> APERTAPE	<u>T</u> ERMINAL
<u>D</u> ISC	<u>P</u> APER-TAPE	
<u>D</u> ISK	<u>P</u> LOTTER	

3.2  
(continued)

number is a manufacturer's type number or model number, with a maximum of eight digits. No validation of specific numbers is performed

model is a character string of not more than eight printable characters, being a manufacturer's model identifier. Spaces are regarded as printable characters.

name is a character string of not more than 256 printable characters, being any name that is appropriate to the operating system and to the physical storage medium

size is an unsigned integer of from one to ten digits, with a maximum value of 2147483647, being any indication of the file's size that is meaningful to the user

common clauses are any of the following clauses, as defined in section 4.3 of the User's Guide, in any order:

<u>ACCESS-AUTHORITY</u>	FREQUENCY
<u>ADMINISTRATIVE-DATA</u>	- <u>NOTE</u>
<u>ALIAS</u>	<u>OBSOLETE-DATE</u>
<u>CATALOGUE</u>	<u>QUERY</u>
<u>COMMENT</u>	<u>SECURITY-CLASSIFICATION</u>
<u>DESCRIPTION</u>	SEE
<u>EFFECTIVE-DATE</u>	

#### Remarks

- 1 The FILE statement identifier must be followed immediately by one of the synonymous keywords MARKIV or MARK-IV. Except for this and for changes in the CONTAINS clause, the format of the FILE MARKIV data definition statement is the same as that of the FILE data definition statement. The specification of the CONTAINS clause is given below; for the specification of the rest of the data definition statement, reference should be made to that of the FILE statement in section 4.2 of the User's Guide.
- 2 Members whose names appear (as segment-name or segment-name-p) in the CONTAINS clause can be of member type MARKIV-SEGMENT (or equivalently, MARK-IV-SEGMENT), GROUP, or ITEM. When the Source Language Generation facility is used to generate a MARK IV File Definition form from a FILE MARKIV member, any GROUP or ITEM member specified in the CONTAINS clause as segment-name or segment-name-p is treated as though it had been defined as a MARKIV-SEGMENT member.

3.2  
(continued)

- 3 The CONTAINS clause specifies (as segment-name) the names of from one to 99 members; representing the segments of a **MARK IV** file. Each segment-name specified, except the first in the list, must be preceded by a comma; the comma can optionally be preceded **and/or** followed by spaces. The segment-name specifications must appear in the top to bottom, left to right order of the segment hierarchy, as described in the examples of Chapter 2.
- 4 The first member name specified in the CONTAINS clause as segment-name must be the name of the base (level one) segment in the hierarchy. Any subsequent segment-name specification must have an associated subordinate **PARENT** clause, specifying as segment-name-p the name of the parent segment of segment-name. A segment-name-p specification must have been declared (as segment-name) earlier in the CONTAINS clause.
- 5 For each segment-name specification after the first, following the **PARENT** clause, an optional subordinate clause may be specified indicating possible multiple occurrence of the named segment, as follows:
  - if segment-name is a variably repeating segment, then a **COUNT-FIELD** item-name-c clause must be specified indicating the name of the item held as a count field in segment-name-p (the parent segment of segment-name) and containing the (variable) number of occurrences of segment-name in the hierarchy. Thus the integer held in item-name-c may vary for each occurrence of segment-name within a logical record.
  - if segment name occurs a fixed number of times in the hierarchy (for each occurrence of segment-name-p in the file), then the **OCCURS** integer clause must appear specifying the predetermined fixed number.
- 6 If a **SORT-KEY** clause is included in a **FILE MARKIV** statement, it is treated by **DATAMANAGER** as documentation only; the information is not used in processing a relevant **PRODUCE MARKIV** command. Therefore, since every **MARK IV** segment must contain a key field, it is necessary, if a **PRODUCE** command is to generate valid **MARK IV** definitions from a **FILE MARKIV** member, that the **KEYS** option (see section 3.3) be exercised in the data definition of each **MARKIV-SEGMENT** member and each **GROUP** member named in the **CONTAINS** clause of the relevant **FILE MARKIV** statement. (See further in remark 14 of the **PRODUCE** command specification in section 4.2.) If an **ITEM** member is named in a **FILE MARKIV CONTAINS** clause for use as a **MARK IV** segment, it will automatically be taken as the key for that segment; key order (that is, ascending or descending), however, will not be assigned automatically and, if the user wishes to specify a key order, he must further create a **MARKIV-SEGMENT**

3.2  
(continued)

or GROUP member containing the item and exercise the KEYS option for the item in the required MARKIV-SEGMENT or GROUP data definition statement. Segment keys (however defined and whatever the member type of the containing segment) are not automatically assigned a key order; if the user does not define a key as either ascending or descending, no default will be taken and the key order will remain unspecified.

- 7 Common clauses, listed under Format above, can be present in any type of data definition statement; they are therefore defined separately, in section 4.3 of the User's Guide. Not more than one of each of these clauses can be declared. If a common clause has a subordinate clause or keyword, the subordinate clause identifier or subordinate keyword must not be truncated to an extent where it becomes ambiguous with any other clause identifier or other keyword available in the data definition syntax for this member type.
- 8 The major clauses following the MARKIV (or MARK-IV) keyword can appear in any order, provided that subordinate clauses are not separated from the other elements of the major clause in which they are contained. For example, the CONTAINS clause may include subordinate PARENT, COUNT-FIELD, and OCCURS clauses, and no other major clause may be interposed between its subordinate clauses.
- 9 A record containing the MARK IV file data definition statement can be inserted into the data dictionary's source data set by a suitable command such as ADD or INSERT (see User's Guide, Chapter 31, and an encoded record can subsequently be generated and inserted into the data entries data set. If, when the encoded record is generated, any member whose name appears in the file's data definition has no data entries record, DATAMANAGER creates a dummy data entries record for that member. The dummy record is inserted as a dummy MARKIV-SEGMENT record if the member's name appears in the CONTAINS clause as segment-name or segment-name-p (but not as item-name-c). Otherwise the dummy record is created as a dummy ITEM record unless the name appears in a USER-LABELS clause, in which case a dummy MODULE record is created.

Format

```

{MARKIV-SEGMENT
 MARK-IV-SEGMENT}

[ENTERED-AS
 HELD-AS
 REPORTED-AS
 DEFAULTED-AS] [ALIGNED
 UNALIGNED
 NOT-ALIGNED] [CONTAINS content

[ELSE content [IF clause] } [ELSE content [IF clause]...
 [IF clause

[,content { [ELSE content [IF clause]
             [IF clause
             [ELSE content [IF clause]... I...}

[KEYS { item-name } { [UNIQUE
                       DUPLICATED] } { [ASCENDING
                                         DESCENDING] }
        [, { item-name } { [UNIQUE
                           DUPLICATED] } I... ]

[USER-EXIT module-name] [common clauses]

{ ;
  .

```

where

```

content      } are as defined for a GROUP data definition
IF clause   } statement in section 4.2 of the DATAMANAGER
item-name     } User's Guide
group-name    }
module-name   }

```

common clauses are any of the following clauses, as 'defined in section 4.3 of the User's Guide, in any order:

- |                              |                                  |
|------------------------------|----------------------------------|
| - <u>ACCESS-AUTHORITY</u>    | - <u>FREQUENCY</u>               |
| - <u>ADMINISTRATIVE-DATA</u> | - <u>NOTE</u>                    |
| - <u>ALIAS</u>               | - <u>OBSOLETE-DATE</u>           |
| - <u>CATALOGUE</u>           | - <u>QUERY</u>                   |
| - <u>COMMENT</u>             | - <u>SECURITY-CLASSIFICATION</u> |
| - <u>DESCRIPTION</u>         | - <u>SEE</u>                     |
| - <u>EFFECTIVE</u>           |                                  |

Remarks

- The member type identifiers MARKIV-SEGMENT and MARK-IV-SEGMENT are synonymous.

3.3

(continued)

- 2 With the exception of the KEYS clause, the format of the MARKIV-SEGMENT data definition statement following the statement identifier is the same as that of the GROUP data definition statement. In a MARKIV-SEGMENT member's KEYS clause, the optional keyword ASCENDING or DESCENDING, indicating key order, may be declared only for the first member named and, if declared, applies to all members named in the clause. If neither ASCENDING nor DESCENDING is declared, no default is taken and key order is unspecified for all members named in the clause.
- 3 The MARKIV-SEGMENT CONTAINS clause may contain only the names of GROUP or ITEM members. It must not contain the name of another MARKIV-SEGMENT member.

### 3.4

#### MARK IV RELATED ASPECTS OF THE ITEM DATA DEFINITION STATEMENT

The ITEM data definition statement (see DATAMANAGER User's Guide, section 4.2) can include the MARK IV related form-description keywords, ROUNDED and TRUNCATED, which may be specified for use in producing MARE IV File Definition forms.

A bit string item is generated in the MARK IV File Definition as a binary field, the length of which is the number of whole bytes required to contain the number of bits that constitute the item. If alignment is specified (see the FILE or GROUP data definition statements in the User's Guide, section 4.2) or RNDBIT=YES (see Appendix 1) then each bit string item will be byte aligned in the MARE IV File Definition. If a bit string item that does not end at a byte boundary is immediately followed by a bit string item that is not aligned, then the MARK IV definition for this second item will be generated so that the item is shown at the same byte offset as the last byte of the preceding item.



## 4.1

## INTRODUCTION

DATAMANAGER can generate, and optionally insert into an output source library data set, MARX IV File Definition forms from encoded FILE, IMS-DATABASE, SEGMENT (for IMS), MARKIV-SEGMENT, GROUP, and ITEM data dictionary members, where:

- FILE members can define conventional or MARX IV files,
- IMS-DATABASE members can be HDAM, HSAM, HIDAM, or HISAM databases,
- SEGMENT members are restricted to PHYSICAL segments that contain no logical relationships,
- a GROUP or ITEM member must have been named as segment-name in the CONTAINS clause of a FILE MARKIV member's data definition statement.

The File Definition forms generated are suitable for direct input to the MARX IV File Management System.

MARX IV File Definition form generation from members of the following member types is not yet available but will be made available later:

- IMS-DATABASE GSAM, LOGICAL, and SECONDARY-INDEX
- SEGMENT LOGICAL and INDEX-POINTER
- TOTAL-DATABASE
- FILE TOTAL.

Generation of File Definition forms is achieved through use of the PRODUCE- command. The basic form of the command, used to generate record layouts and COBOL, PL/I, or Assembler data descriptions for conventional file environments, is described in a separate DATAMANAGER manual, Source Language Generation. The modifications to the PRODUCE command required for MARX IV File Definition form generation are specified in section 4.2. Tables are presented in Chapter 5 indicating the File Definition form entries generated by DATAMANAGER and the corresponding DATAMANAGER entities and actions used in generating them.

4.1  
(continued)

The installation macro DGMIV (see Appendix 1), can be used to tailor the output from the MARK IV version of the PRODUCE command to conform to the standards of a particular installation. Tailoring is performed by Controller assignment of values to macro keywords (as alternatives to the default values assigned by MSP). In the specifications below, any condition or value that is initially defined by MSP but that can be tailored by the Controller is annotated "... (unless tailored, see xxxxxx)", where xxxxxx is the relevant DGMIV keyword. The installed values of certain DGMIV keywords (whether default values assigned by MSP or tailored values assigned by the Controller) may be overridden by the user for a particular PRODUCE command by specification of relevant clauses in the command; for example, the values assigned to the keywords FDGLOS and DDNAME may be overridden for a PRODUCE command by specification of the clauses GLOSSARY-CODE 'x' and ONTO file-name, respectively.

## SPECIFICATION OF THE PRODUCE COMMAND FOR MARK IV FILE DEFINITION FORM GENERATION

### Purpose

To generate MARK IV File Definition forms for subsequent use as input to the MARK IV File Management System.

### Format

PRODUCE { MARKIV  
MARK-IV } FROM member-name [AS library-name]

[,member-name [AS library-name]]...

[CONTAINED-BY markiv-file-name]

[control-options] { !  
.

where

member-name is the name of an encoded member of any of the following member types:

- FILE (defining a conventional file)
- FILE MARKIV
- IMS-DATABASE (defining an IMS HDAM, HSAM, **HIDAM**, or **HISAM** database)
- SEGMENT (defining an IMS PHYSICAL segment)
- MARKIV-SEGMENT
- GROUP or ITEM, provided that the GROUP or ITEM member's name has been specified as segment-name in the CONTAINS clause of an encoded FILE MARKIV member's data definition (see section 3.2) and thus is treated as if it had been defined as a MARKIV-SEGMENT member

library-name is the name to be given to:

- the generated library member in the output data set, and/or
- the File Name entry in the generated FD header record.

It is restricted to a maximum of eight characters, of which the first must be alphabetic and the rest must conform to the rules stated in the MARK IV Reference Manual for specifying the symbolic name, that is, the File Name of a MARK IV file.

markiv-file-name is the name of an encoded member of member type FILE MARKIV that contains all the members of member type MARKIV-SEGMENT, GROUP, or ITEM named in the FROM clause

4.2  
(continued)

control-options is as defined in section 2.2 of the Source Language Generation manual, except that the format of output-form-1 in the GIVING clause, subject to remarks 22 to 32 below, **is**:

```
{ RECORDS-ONLY  
  FD  
  FILE-IDENTITY 'xxxxxxx'  
  GLOSSARY-CODE 'x'  
  BUFFER-SIZE nnnnn  
  END-RECORDS  
  [s] NOTES  
  [s] DESCRIPTIONS  
  KNOWN-AS
```

where

xxxxxxx is a character string of up to eight characters

x is a character string consisting of exactly one of the characters A, B, N, 1, or space

nnnnn is an unsigned integer in the range 1 to 32767 for DOS users or 1 to 32760 for OS users, being the maximum number of bytes of main storage required to hold an entire logical record, that is, every occurrence of all sensitive segments

s is an unsigned integer specifying a number of consecutive delimited character strings commencing with the first such string of a member's NOTE or DESCRIPTION clause.

#### Remarks

- 1 The command identifier, PRODUCE, must be followed immediately by the context keyword MARKIV (or MARK-IV), which in turn must be followed by the FROM clause (with its optional subordinate AS clauses, if applicable). **Any** control-options specified may precede or follow the FROM clause, but must not precede the context keyword. The CONTAINED-BY clause, if declared, must immediately follow the last entry of the FROM clause.
- 2 Up to a maximum of 16 member names can be declared in the FROM clause. If two or more are declared, each except the first must be preceded by a comma; the comma can optionally be preceded and/or followed by spaces. Member names are processed in the order in which they appear in the FROM clause.

4.2  
(continued)

- 3 Acceptance of the PRODUCE command is in respect of each member-name individually. If member-name:
- is not encoded, or
  - is not present in the data dictionary, or
  - is of a member type that is not valid in the context, or
  - is protected against access by the user (see remark 4)

a message is output, no generation takes place in respect of that member-name, and processing continues with the next member-name or command.

- 4 Acceptance of the PRODUCE command is subject to access security levels (for each member-name individually, as stated in remark 3). See the specification of the PROTECT command in the User's Guide, and the description of the security system in section 1.7 of the User's Guide. If a member-name has an access security level higher than the user's (general or specific) security level, the command is rejected in respect of that member-name. If the command can be accepted in respect of member-name, but a reference path from member-name includes a protected member with an access level higher than the user's security level, the reference path is, if appropriate, followed to its end to determine the total storage space required, but the name of no member in that reference path beyond the last **member** to which the user has access is given; instead, a filler name is generated.
- 5 If member-name is the name of an encoded MARKIV-SEGMENT, GROUP or ITEM **member**, and:

- no encoded directly containing FILE MARKIV **member** exists in the data dictionary; or
- a CONTAINED-BY markiv-file-name clause follows the FROM clause in the command, but the encoded FILE MARKIV member named in that clause does not directly contain member-name

then the error message:

member-type member-name NOT CONTAINED BY MARKIV FILE

is output, and DATAMANAGER proceeds to the next **member-name** or command.

- 6 If member-name is the name of an encoded SEGMENT member, but no encoded directly containing IMS-DATABASE member exists in the data dictionary, the error message:

member-type member-name NOT CONTAINED BY IMS DATABASE

4.2  
(continued)

is output, and **DATAMANAGER** proceeds to the next member-name or command.

7 Subject to remarks 3 to 6 above, a single MARK IV File Definition form is generated:

- for the complete segment hierarchy specified in member-name's CONTAINS clause, if member-name is a FILE MARKIV or an IMS-DATABASE; or
- for the complete segment hierarchy determined in accordance with the rules stated in remarks 8 and 9, if member-name is a conventional FILE; or
- for the segment member-name only, if member-name is a MARKIV-SEGMENT, a GROUP, an ITEM or a SEGMENT.

8 When member-name is a conventional FILE, then a GROUP or ITEM member directly or indirectly contained by member-name is treated as a contained MARK IV segment if:

- the GROUP or ITEM member is described in its containing GROUP or FILE member's data definition as subject to a name occurs bound, that is, as occurring in a variable length array; or
- the GROUP is described in its containing GROUP or FILE member's data definition as occurring in a fixed length array, and it directly or indirectly contains a key field; or
- the ITEM is described in its containing GROUP or FILE member's data definition as occurring in a fixed length array, and is itself a key field.

A key field in this context is a member which is named in the SORT-KEY clause of its directly or indirectly containing FILE member's data definition.

When an ITEM member is treated as a MARK IV segment in the MARK IV file defined by member-name, both LS segment and LO field records appear for the ITEM member in the File Definition form generated from member-name. (See section 4.3.)

It is recommended that all MARK IV files be defined using FILE MARKIV (or IMS-DATABASE) members rather than conventional FILE members, so that all constituent segments can be named directly in the CONTAINS clause of the member's data definition (see section 3.2).

4.2  
(continued)

- 9 For a GROUP member directly or indirectly contained by member-name, where member-name is a conventional FILE, ELSE clauses in the GROUP member's data definition represent redefinitions of the MARK IV file structure and produce alternative File Definition forms.
- 10 If member-name is a FILE MARKIV, a conventional FILE or an IMS-DATABASE, DATAMANAGER generates a segment number and a level number for each segment in the segment hierarchy. Segment numbers are assigned hierarchically as described in section 2.1, beginning with 1 and incremented by 1 (unless tailored, see INCLEV) up to a maximum of 99 segments (unless tailored, see MAXSEG), but with the proviso that no segment may have a segment number greater than 99. Thus, for example, if the increment selected is 2, no more than 50 segments can be generated in the hierarchy. Level numbers are also assigned to the segments in hierarchical fashion as described in section 2.1.
- 11 If member-name is a MARKIV-SEGMENT, a GROUP or an ITEM, DATAMANAGER generates a segment number and a level number or the segment. The numbers generated are those that would be assigned to the segment if generation were from the containing FILE MARKIV member. If there are two or more containing FILE MARKIV members, a CONTAINED-BY clause must immediately follow the FROM clause in the PRODUCE command, to specify which of the containing members is relevant to this segment number and level number generation. If a CONTAINED-BY clause is present, each MARKIV-SEGMENT, GROUP and ITEM member named in the FROM clause must be contained in the same specified MARK IV file.
- 12 If member-name is a SEGMENT, DATAMANAGER generates a segment number and a level number for the (MARK IV) segment. The numbers generated are those that would be assigned to the segment if generation were from the containing IMS-DATABASE member. As there can only be one containing IMS-DATABASE member, a CONTAINED-BY clause is never required in the command for generation from SEGMENT members; although such a clause can be present in respect of other members named in the FROM clause, as stated in remark 11.
- 13 Key fields of generated segments are determined as follows:
  - if member-name is a conventional FILE, the key fields of component segments are determined from the SORT-KEY clause of member-name's data definition. If a KEYS clause is specified in a component GROUP member's data definition, it is treated as documentation only and is not used by DATAMANAGER to identify key fields.

4.2

(continued)

- if member-name is a FILE MARKIV, key fields are determined for its component segments as described in remark 6 of the FILE MARKIV statement specification in section 3.2
- if member-name is a MARKIV-SEGMENT or GROUP, the KEYS clause must be present in its data definition to identify the key fields
- if member-name is an ITEM, it is taken to be a key field as well as a segment, and an LO field record as well as an LS segment record are generated (see section 4.3). Key order (that is, ascending or descending), is not specified. (If the user wishes to specify a key order, he must:
  - ADD a new MARKIV-SEGMENT or GROUP member directly containing the ITEM member (preferably as the only contained member), and with a KEYS clause declaring the item's name and key order
  - MODIFY the containing FILE MARKIV member, to CONTAIN the new member instead of the ITEM member
  - issue a further PRODUCE command, with the new member's name in the FROM clause instead of the ITEM's name.)
- if member-name is an IMS-DATABASE, the key fields of generated segments are determined from the SEQUENCE-KEY clauses of contained SEGMENT members
- if member-name is a SEGMENT, key fields are determined from its SEQUENCE-KEY clause.

Segment keys are not automatically assigned a key order; if the user does not define a key as either ascending or descending in the relevant SORT-KEY or KEYS clause, no default is taken and the key order remains unspecified.

- 14 If DATAMANAGER cannot determine a segment's key field (for example, if member-name is a conventional FILE that contains a variable length array which does not directly or indirectly contain a member named in the SORT-KEY clause of member-name's data definition; or if member-name is a MARKIV-SEGMENT or a GROUP, and the KEYS clause has been omitted from its data definition) then the warning message:

```
AT LEAST 1 SEGMENT KEY MUST BE SPECIFIED FOR A
MARK IV SEGMENT
```

is output, and generation continues.

4.2  
(continued)

15 MARE IV **allows** no more than three key fields to constitute the key of a segment. Consequently, if more than three key fields have been defined for a segment, some reorganisation of the data definitions held in the dictionary is necessary: see section 4.3.

16 Each AS clause present in the command relates only to the member-name that immediately precedes it. It declares:

- the name under which the File Definition form generated from member-name is to be cataloged in the output source library data set. (This does not apply when the control-option NOGENERATION or NO-GENERATION is specified in the command.)
- the name to be output as the File Name entry in the FD record of the File Definition form. (This does not apply when member-name is a MARKIV-SEGMENT, a GROUP, an ITEM or a SEGMENT, unless the control option GIVING FD is specified in the command.)

17 For each member-name for which no AS clause is specified:

- in respect of the name under which the generated File Definition form is **catalogued**, library-name is defaulted to:
  - member-name, if member-name **is** a FILE MARKIV, an IMS-DATABASE or a conventional FILE
  - the name of the containing FILE MARKIV or IMS-DATABASE member, if member-name is a MARKIV-SEGMENT, a GROUP, an ITEM or a SEGMENT

provided that the name conforms to the length restriction on library-name. The length restriction on library-name is a maximum of eight characters. If the name of the relevant member is longer than eight characters, no generation takes place in respect of member-name, a message is output, and processing continues with the next member-name or command. (This does not apply when the **control-**option NOGENERATION or NO-GENERATION is specified in **the command.**)

- in respect of the name output as the File Name entry in the FD record of the File Definition form, **library-**name is defaulted to:
  - member-name, if member-name is a FILE MARKIV, an IMS-DATABASE or a conventional FILE;
  - the name of the containing FILE MARKIV or IMS-DATABASE member, if member-name is a MARKIV-SEGMENT, a GROUP, an ITEM or a SEGMENT,

4.2  
(continued)

and the control-option GIVING FD is specified in the command (FD records are not generated for members of these types unless GIVING FD is specified);

unless the member from which the library-name is taken has an ALIAS clause, and either:

- the ALIAS clause declares a MARKIV or MARK-IV specific alias, the value of the ALIAS keyword in the macro DGMIV is YES, and the control-option OMITTING ALIAS is not specified in the command; in which case library-name is defaulted to the MARKIV or MARK-IV specific alias

or:

- the ALIAS clause declares a MARKIV or MARK-IV specific alias, and the control-option WITH-ALIAS or ALIAS is specified in the command (with no associated specification of alias-type or number); in which case library-name is defaulted to the MARKIV or MARK-IV specific alias

or:

- the control option:

WITH-ALIAS number  
ALIAS alias-type

(see the Source Language Generation manual, section 2.2.3) is specified in the command, and the ALIAS clause includes an alias of the specified number or alias-type; in which case library-name is defaulted to the alias of the specified number or alias-type.

18 In respect of the names under which the generated File Definition forms are catalogued, library-names, whether declared or defaulted, are not subject to any name editing, ALIAS or WITH-ALIAS clauses (see control-options) that may be present in the command.

19 In respect of the names output as the File Name entries in the FD records of the File Definition forms:

- library-names declared in AS clauses are not subject to any name editing, ALIAS or WITH-ALIAS clauses (see control-options) that may be present in the command
- library-names defaulted 'as described in remark 17 are subject to any name editing clauses that may be present in the command.

4 . 2  
(continued)

20 Output control-options can be specified in the command as defined in the Source Language Generation manual, section 2.2.2, except that:

- if ONTO file-name SEQUENTIAL is specified without 'control-card', or if ONTO file-name PARTITIONED is specified, then an RC run control record is generated (unless tailored, see **RC**), except when GIVING RECORDS-ONLY is specified in the command and member-name is not a conventional FILE
- if ONTO file-name SEQUENTIAL 'control-card' is specified, the specified control card is output (instead of the RC run control record), except when GIVING RECORDS-ONLY is specified in the command and member-name is not a conventional FILE
- if ONTO file-name SEQUENTIAL 'control-card' is specified, the question-mark character must not be used to indicate the point within the control-card at which the generated library member name is to be inserted; the required library member name must be included in the character string of control-card.

If an RC run control record or a control card is output, it immediately precedes the generated File Definition form in the output library data set.

21 Generation control-options can be specified in the command as defined in the Source Language Generation manual, section 2.2.3, except that:

- of remark 9 of that section, only the first two sentences apply
- instead of remarks 10 to 22 of that section, the following remarks 22 to 32 apply
- remarks 27 to 31 of that section are irrelevant in the context of MARK IV File Definition form generation.

22 GIVING RECORDS-ONLY (except when member-name is a conventional FILE) suppresses the generation of RC run control records and RP run parameter records, so that only File Definition form records are produced. (The equivalent **macro** keyword usages are RC=NO and **RPNO=0.**) If GIVING RECORDS-ONLY is not specified, or if member-name is a conventional FILE, RC records are generated (unless tailored, see **RC**) and RP records are suppressed (unless tailored, see **RPNO**).

4.2  
(continued)

- 23 GIVING FD is relevant only when member-name is a MARKIV-SEGMENT, a GROUP, an ITEM or a SEGMENT. It specifies that an FD record is to be included in the File Definition form. If GIVING FD is omitted, no FD record is produced when member-name is of one of these member-types.
- 24 GIVING FILE-IDENTITY 'xxxxxxx' is relevant only when member-name is a FILE MARKIV, a MARKIV-SEGMENT, a GROUP or an ITEM. It specifies that the value xxxxxxx is to be the File Identification entry in the FD record, if there is an FD record in member-name's File Definition form. If GIVING FILE-IDENTITY 'xxxxxxx' is not specified, or if member-name is a conventional FILE, an IMS-DATABASE or a SEGMENT, then:
- the name of the member is used as the File Identification entry, if member name is a conventional FILE, a FILE MARKIV or an **IMS-DATABASE**; or
  - the name of the containing FILE MARKIV or **IMS-DATABASE** member is used as the File Identification entry, if member-name is a **MARKIV-SEGMENT**, a GROUP, an ITEM or a SEGMENT, and has an FD record in the generated File Definition form.
- 25 GIVING GLOSSARY-CODE 'x' is relevant only when member-name is a FILE MARKIV, a MARKIV-SEGMENT, a GROUP or an ITEM. It specifies that the value x is to be the Glossary entry in the FD record, if there is an FD record in member-name's File Definition form. (The equivalent **macro** keyword usage is **FDGLOS='x'**.) If GIVING GLOSSARY-CODE 'x' is not specified, or if member-name is a conventional FILE, an IMS-DATABASE or a SEGMENT, the Glossary entry is B (unless tailored, see **FDGLOS**).
- 26 GIVING BUFFER-SIZE nnnnn is relevant only when member-name is an IMS-DATABASE or a SEGMENT. It specifies that the value of nnnnn is to be the Buffer Size entry in the FD record, if there is an FD record in member-name's **File** Definition form. If GIVING BUFFER-SIZE nnnnn is omitted, and **member-name is** an IMS-DATABASE or a SEGMENT, the Buffer Size entry is left blank. If member-name is a conventional FILE or a FILE MARKIV, and its definition includes maximum-block-size, the maximum-block-size is used for the Buffer Size entry. In all other cases, the Buffer Size entry is left blank.
- 27 GIVING END-RECORDS specifies that, except when **member-name** is a conventional FILE, an End record is to be generated for each segment in the File Definition form. If GIVING END-RECORDS is not specified, or if member-name is a conventional FILE; End records are not generated.

4.2  
(continued)

- 28 GIVING s NOTES specifies that AA comment records are to be included in the File Definition form, to contain the first s delimited character strings of NOTE clauses from member-name and from each of member-name's directly or indirectly contained members (unless selectively tailored by member type, see FICOM, IMDBCOM, MIVSCOM, MSGCOM, GRCOM and ITCOM). (The equivalent macro keyword usage is **NOTE=s.**)
- 29 GIVING NOTES specifies that AA comment records are to be included in the File Definition form, to contain all delimited character strings of NOTE clauses from **member-name** and from each of member-name's directly or indirectly contained members (unless selectively tailored by member type, see FICOM, IMDBCOM, MIVSCOM, MSGCOM, GRCOM and ITCOM). (The equivalent macro keyword usage is **NOTE=ALL.**)
- 30 GIVING s DESCRIPTIONS specifies that AA comment records are to be included in the File Definition form, to contain the first s delimited character strings of DESCRIPTION clauses from member-name and from each of member-name's directly or indirectly contained members (unless selectively tailored by member type, see FICOM, IMDBCOM, MIVSCOM, MSGCOM, GRCOM and **ITCOM**). (The equivalent macro keyword usage is **DESC=s.**)
- 31 GIVING DESCRIPTIONS specifies that AA comment records are to be included in the File Definition form, to contain all delimited character strings of DESCRIPTION clauses from member-name and from each of member-name's directly or indirectly contained members (unless selectively tailored by member type, see FICOM, IMDBCOM, MIVSCOM, MSGCOM, GRCOM and ITCOM). (The equivalent macro keyword usage is **DESC=ALL.**)
- 32 GIVING KNOWN-AS specifies that generated data names of the following types are to be based wherever possible on local-names from containing members' KNOWN-AS clauses, instead of on the members' names or aliases:
- Field Name entries in LS records, only where the segment is a GROUP or ITEM member directly or indirectly contained in a conventional FILE
  - Field Name entries in LO and Ln records.
- (The equivalent macro keyword usage is **KNOWNAS=YES.**)
- 33 Data names that can be based on aliases are:
- the File Name entry in the FD record of the File Definition form, provided that it is not governed by an AS clause (see remarks 16 and 17); and hence the File Name entries in related LS, LO and Ln records

4.2  
(continued)

- the Field Name entries in the LS, LO and Ln records of the File Definition form.

34 Name editing control-options can be specified in the command as defined in the Source Language Generation manual, section 2.2.4. They can affect:

- File Name entries in FD records (and in the associated LS, LO and Ln records) to the extent defined in remark 19
- File Identification entries in FD records
- Field Name entries in LS, LO and Ln records.

35 After all editing specified by editing clauses has been completed, DATAMANAGER performs a final editing of each generated data name in the categories listed in remark 34, to ensure that each name conforms with MARK IV source language requirements; thus:

- if the first character is non-alphabetic, it is removed
- any occurrence of the MARK IV delimiting character (**#**, unless tailored, see DLIMIT) is removed
- if there are more than eight characters remaining in the name, middle characters are removed to shorten the name to eight characters
- if there are less than eight characters remaining in the name, the name is lengthened to eight characters by the addition of trailing space characters
- thereafter, if two or more Field Name entries in LS or LO records of the same File Definition are identical, (for example, if:
  - a number of LO entries are generated from a DATAMANAGER fixed length array; or
  - a **number** of LS entries are generated from an array as defined in remark 8 above; or
  - a GROUP or ITEM member appears in several segments)

then DATAMANAGER assigns numeric suffixes to distinguish between occurrences of the generated name; thus:

4.2  
(continued)

- the first occurrence of the name is left unaltered
- for each subsequent occurrence of the name, DATAMANAGER replaces the last n characters (where n is 2 unless tailored, see **REPBYTE**) with a right justified integer of n digits, including leading zeros if applicable, beginning at 2 for the second occurrence and incremented by 1 for each subsequent occurrence. If this leaves any space characters immediately preceding the suffix, such space characters are replaced by zeros. For example, if **GROUPA** occurs in an array three times and contains **ITEM1** and **ITEM2**, and if the value of **REPBYTE** in the macro **DGMIV** is 2, then the following Field Name entries are generated:

- **GROUPA**
- **ITEM1**
- **ITEM2**
- **GROUPA02**
- **ITEM1002**
- **ITEM2002**
- **GROUPA03**
- **ITEM1003**
- **ITEM2003**

- the number of digits, n, (as specified by **REPBYTE**) imposes a limit on the number of occurrences of a name that can be given a unique **suffix**, so that if, for example, the value of **REPBYTE** is 2 and there are more than 99 occurrences of a generated name, a warning message is output and the numbering sequence starts again, from 00 for the 100th occurrence of the name.

36 The length of a MARK IV field cannot exceed 255 characters. Therefore, if a relevant GROUP or ITEM member's specified length exceeds 255 characters, DATAMANAGER creates two or more Field Name entries, as required, each except (perhaps) the last representing a field of 'length 255 characters. Suffixes are applied to the generated names of these fields, as described in remark 35.

### 4.3

### SUPPORTING INFORMATION

A MARK IV File Definition form produced by DATAMANAGER can contain the **following** types of records:

- a Run Control record, identified by the Form Code entry RC. This contains:
  - a File Name entry identical with that in the FD Header record
  - the Form Code entry, RC
  - the delimiter (**#**, unless tailored, see **DLIMIT**) to be used in Ln records for spacing of column headings.

Alternatively, a control card can be output: see section 4.2, remark 20.

- one or two Run Parameter cards, identified by the Form Code entry RP. These contain:
  - a File Name entry identical with that in the FD Header record
  - the Form Code entry, RP
  - the parameter data specified by the value of the keyword **RP1** (for the first card) or **RP2** (for the second card) of the macro **DGMIV**

Whether Run Parameter cards are produced, and if so whether one or two are produced, depends on the value of the keyword **RPNO** of the macro **DGMIV** (except when **GIVING RECORDS-ONLY** is stated in the command).

- an FD Header record
- an LS Segment Name record for each segment in the hierarchy; or for member-name only, if member-name is a segment
- an LO Field Attributes record for each field of each segment. If a segment is an ITEM member, both an LS record **and** an LO record are generated in respect of that member
- one or more Ln (**n=1,2,...,9**) Column Heading records for each ITEM field of each segment
- one **or** more AA Comment records, subject to tailoring and to the **GIVING** clause of the command (see section 4.2, remarks 28 to **31**)

4.3  
(continued)

- an End record for each segment designating the last byte of the segment.

The table on page 4-18, with its associated notes, summarises the types of records that can be produced, by member type of member-name.

Chapter 5 contains a table describing the contents of the FD, LS, LO and Ln records and relating the generated output to the DATAMANAGER language.

If it is required to produce a single File Description form to contain entries for several but not all of the segments defined in the CONTAINS clause of an encoded FILE MARKIV or IMS-DATABASE member, this can be achieved by coding two successive PRODUCE MARKIV commands, as follows:

- a PRODUCE command with a single member-name specification (for a MARKIV-SEGMENT, GROUP, ITEM, or SEGMENT member), and with a GIVING FD clause
- a second command with member-name specifications only for the remaining required MARKIV-SEGMENT, GROUP, or ITEM members or for the remaining required SEGMENT members, with a GIVING RECORDS-ONLY clause.

The commands must conform to the requirements of remark 11 of section 4.2 with regard to CONTAINED-BY clauses.

MARK IV allows no more than three key fields to constitute the key of a segment. Consequently, if more than three key fields have been defined for a segment, some of the key fields must be combined into one or more new GROUP members (as GROUP rather than ITEM members so that the intended identity and function of the combining fields will not be lost, other than that of being distinct key fields) with the following provisos:

- all members combined to form a new member must be contiguous in the segment (thus retaining their function, but not their identity, as key fields)
- the combining members originally must all have been contained in a single member, that is, they must all have been named in the CONTAINS clause of the originally containing member's data definition.

A data definition must be **ADDED** to the data dictionary for each new GROUP member thus formed, and the new member must be named in place of the combining members, both:

(text continues on page 4-20)

4.3

(continued)

MARK IV File Definition Form Record Generation (Yes or No), by Record Type/Member Type Combination, for member-name					
Member Type Record Type	FILE MARKIV'	IMS- DATABAS E	MARKIV- SEGMENT, GROUP, or ITEM	SEGMENT	FILE (Conventional)
RC	Yes <sup>1</sup>	Yes <sup>1</sup>	Yes <sup>1</sup>	Yes <sup>1</sup>	Yes <sup>2</sup>
RP	No <sup>3</sup>	No <sup>3</sup>	No <sup>3</sup>	No <sup>3</sup>	No <sup>4</sup>
FD	Yes	Yes	No <sup>5</sup>	No <sup>5</sup>	Yes
LS	Yes <sup>6</sup>	Yes <sup>6</sup>	Yes <sup>6</sup>	Yes <sup>6</sup>	Yes <sup>6,7</sup>
LO	Yes <sup>8,9</sup>	Yes <sup>8</sup>	Yes <sup>8</sup>	Yes <sup>8</sup>	Yes <sup>8,9</sup>
Ln (n=1, 2, ...,9)	Yes <sup>10</sup>	Yes <sup>10</sup>	Yes <sup>10</sup>	Yes <sup>10</sup>	Yes <sup>10</sup>
AA	Yes <sup>11</sup>	Yes <sup>11</sup>	Yes <sup>11</sup>	Yes <sup>11</sup>	Yes <sup>11</sup>
End Record	No <sup>12</sup>	No <sup>12</sup>	No <sup>12</sup>	No <sup>12</sup>	No

### 4.3

(continued)

#### Notes on the Table on Page 4-18:

- 1 Except when GIVING RECORDS-ONLY is included in the PRODUCE command, or unless tailored, see RC. (If SEQUENTIAL 'control-card' is specified, a control card is produced no matter what the value of RC in the DGMIV macro, unless overridden by a GIVING RECORDS-ONLY clause).
- 2 Unless tailored, see RC.
- 3 Unless tailored, see RPNO, when RP records are generated unless overridden by inclusion of a GIVING RECORDS-ONLY clause in the PRODUCE command.
- 4 Unless tailored, see RPNO.
- 5 Except when overridden by inclusion of GIVING FD in the PRODUCE command.
- 6 Unless tailored, see LS.
- 7 See remark 8 of section 4.2 for generation of segments from GROUP and ITEM members directly or indirectly contained in a conventional FILE.
- 8 For each GROUP member (unless tailored, see GROUP) and each ITEM member directly or indirectly contained in member-name (or for member-name if itself an ITEM member but not if it is a GROUP member).
- 9 Not for directly or indirectly contained GROUP members taken as MARK IV segments.
- 10 For each ITEM member, directly or indirectly contained in member-name, whose data definition contains a HEADINGS clause (or for member-name if itself an ITEM member).
- 11 When tailored values (greater than 0) are in effect for the NOTE and/or DESC keywords in the DGMIV macro, or a GIVING [s] NOTES and/or a GIVING [s] DESCRIPTIONS clause is included in the PRODUCE command; unless selectively tailored, see FICOM, MIVSCOM, GRCOM, ITCOM, IMDBCOM, and IMSGCOM.
- 12 Except when GIVING END-RECORDS is included in the PRODUCE command.

4.3  
(continued)

- in the CONTAINS clause of the originally containing member's data definition and
- in the appropriate clause (SORT-KEY, KEYS, or SEQUENCE-KEY) in which the key fields for the segment were identified (see remark 13 of section 4.2).

## EXAMPLE OUTPUT

The output listing shown in figure 4.1 is an example File Definition form generated by a PRODUCE MARK IV command from encoded data definitions for the FILE MARKIV member, PERSONNEL, and its directly and indirectly contained MARKIV-SEGMENT and ITEM members, as defined in the example of section 2.1. (Additional data definitions were inserted in the data dictionary for the members named in the CONTAINS clauses of the five MARKIV-SEGMENT data definitions given in the example.)

The following points are illustrated by the output and the relevant data definitions:

- an RC record appears in the listing since:
  - no GIVING RECORDS-ONLY or SEQUENTIAL 'control-card' clauses are specified in the PRODUCE MARKIV command, and
  - the RC keyword in the macro DGMIV has not been tailored
- an FD record is generated automatically when a File Definition form is produced from a FILE MARKIV member. It is preceded by a dummy File Definition form containing only an FD record with the same File Name **entry**, which will serve to delete any existing input for that file from the MARK IV dictionary (see the Delete entry remarks in the Chapter 5 FD Header Record table).
- all File Name, File Identification, and Field Name entries are based on member names from the data dictionary, because the DGMIV macro keyword usages ALIAS=NO and **KNOWNAS=NO** are in effect and no WITH-ALIAS, GIVING KNOWN-AS, or GIVING FILE-IDENTITY clauses are specified in the PRODUCE command. (There are no ALIAS or KNOWN-AS clauses in the relevant members' data definitions, but if there were, they would in these circumstances be ignored.)
- the name, PERSONNEL, of the FILE MARKIV member has been shortened by DATAMANAGER to conform to the MARK IV requirement of eight characters for File Names, by removal of the middle character 0. (All Field Name entries listed are derived from members' names that meet the MARK IV name length requirement.)

00006 PRODUCE MARK-IV FROM PERSONNEL NOGEN PRINT;

```

PERSONNELRC #
PERSONNELFDPERSNNELY
PERSONNELFDPERSNNEL BV 563
DMC2275W PERSONNEL HAS BEEN SHORTENED TO PERSONNEL
PERSONNELLEMPLOYEE 1
PERSONNELLOEMPLNO 11 1 6C1
PERSONNELLONAME 11 7 30c
PERSONNELLOJOB TITLE 11 3 7 25C
PERSONNELLOCLASS 11 62 1C
PERSONNELLODIVISION 11 63 3C
PERSONNELLODEPT 11 66 4C
PERSONNELLOGROUP 11 70 2C
PERSONNELLOHIREDATE 11 72 7C
PERSONNELLOSALARY 11 79 4P 2
PERSONNELLODEPENOTS 11 83 2F 0
PERSONNELLOSOCSECNO 11 85 9C
PERSONNELLOJOBS 11 94 2F 0
PERSONNELLOSCHCOUNT 11 96 2F 0 2
PERSONNELLO JOBCOUNT 11 98 2F 0 4
PERSONNELLSCHOOL 2
PERSONNELLOSCHNAME 22 1 30C1
PERSONNELLOSCHADDR 22 31100C
PERSONNELLOYEARS 22 131 2F 0
PERSONNELLOHONOURS 22 133 2F 0
PERSONNELLODEGCOUNT 2 2 1 3 5 2 F 0 3
PERSONNELLOREMARKS 22 137100C
PERSONNELLSDEGREE 3
PERSONNELLODEGTYPE 33 1 4C1
PERSONNELLOGRDPTAVG 33 5 2F 0
PERSONNELLODATE 33 7 7c
PERSONNELLEHPLOYER 4
PERSONNELLOEMPLNAME 42 1 30c
PERSONNELLOEMPADDR 42 31100C
PERSONNELLOTITLE 42 131 30C
PERSONNELLOREFCOUNT 42 161 2F 0 5
DMO2712W DUPLICATE NAME SALARY CHANGED TO SALARY02
PERSONNELLOSALARY02 42 163 4P 2
PERSONNELLOTERMDATE 42 167 7C
PERSONNELLOSTRTOATE 42 1 7 4 7C1
PERSONNELLSREFERENCE 5
PERSONNELLOREFNO 53 1 2P1 0
PERSONNELLOREFTITLE 53 3 30c
PERSONNELLOYRSKNOWN 53 33 2F 0
PERSONNELLOEVALTION 53 35 1C
```

DMO2217I PERSONNEL SUCCESSFULLY GENERATED

Figure 4.1 Example of a DATAMANAGER Produced File Definition Form

The following tables indicate, for each FD, LO, Ln  
in = 1, 2, ..., 9), and LS record appearing in a DATAMANAGER  
produced MARK IV File Definition form, the following:

- the record entries listed by DATAMANAGER
- the corresponding DATAMANAGER dictionary members (by member-type) from which the entries are desired or to which they relate
- remarks relating to:
  - the data definition clauses (of the corresponding DATAMANAGER members) or the PRODUCE command clauses used, or
  - the procedures followed by DATAMANAGER in generating the entries.

MARK IV/DATAMANAGER Correspondence: FD Header Record		
MARK IV File Definition Form Entry	Relevant DATAMANAGER Member Type(s)	Remarks
File Name	FILE, FILE MARKIV, IMS-DATABASE (see Note 1)	library-name from AS clause of PRODUCE command, or its default value
FD Form Code		FD always generated
File Identification	FILE, FILE MARKIV, IMS-DATABASE (see Note 1)	From GIVING clause of PRODUCE command, or from name of the member of the relevant type
Delete?		Blank (if the FD record is produced, then a preceding <b>dummy</b> File Definition form is generated consisting only of an FD record which contains the same File Name entry, an FD Form Code entry, and the Delete entry <b>Y</b> )
<b>Glossary</b>		From GIVING clause of PRODUCE command or FDGLOS keyword of DGMIV macro
Record Format	FILE, FILE MARKIV, IMS-DATABASE <b>(see Note 1)</b>	From data definition: F for Fixed Sequential V for Variable Sequential U for Undefined Sequential I for Fixed Indexed Sequential J for Variable Indexed <b>Sequential</b> K for Key-sequenced VSAM E for Entry-sequenced VSAM D for IMS-DATABASE
Record Size	FILE, FILE MARKIV, IMS-DATABASE <b>(see Note 1)</b>	From data definition: Record length in bytes for F and I formats. Maximum record size for unblocked V formats. Blank otherwise.

(continued)

(continued)

(continued)

MARK IV/DATAMANAGER Correspondence: FD Header Record		
MARK IV File Definition Form Entry	Relevant DATAMANAGER Member Type(s)	Remarks
Records per Block	FILE, FILE MARKIV, IMS-DATABASE (see Note 1)	From record length and block length for blocked F and I <b>formats.</b> Blank otherwise.
Buffer Size:	FILE, FILE MARKIV, IMS-DATABASE (see Note 1)	From GIVING clause of PRODUCE <b>command</b> , else omitted, if member-name is an IMS-DATABASE or SEGMENT member. Otherwise: Block size (from data definition) for blocked V, U, J, K and E formats. Blank in all other cases.

Note:

- 1 Member type of member-name or of the member which directly  
contains member-name.

5  
 (continued)

MARK IV/DATAMANAGER Correspondence: LS Segment Name Record		
MARK IV File Definition Form Entry	Relevant DATAMANAGER Member Type(s)	Remarks
File Name	FILE, FILE MARKIV, IMS-DATABASE	Same as File Name in FD record
Form Code		LS always generated
Field Name	MARKIV-SEGMENT, GROUP, ITEM, SEGMENT	Member name or alias or <b>local-name</b>
Delete?		<b>Blank</b>
Segment Number	MARKIV-SEGMENT, GROUP, ITEM, SEGMENT	Number assigned to segment (see section 4.2, remarks 10 <b>to 12)</b>
Segment Key	MARKIV-SEGMENT, GROUP, ITEM, SEGMENT	Single character indicating key order for all key fields <b>in segment.</b> A if ASCENDING is specified D if DESCENDING is specified Blank if neither is specified. See section 4.2, remark 13.

5  
(continued)

MARK IV/DATAMANAGER Correspondence: LO Field Attributes Record		
MARK IV File Definition Form Entry	Relevant DATAMANAGER Member Type(s)	Remarks
File Name	FILE, FILE MARKIV, IMS-DATABASE	Same as File Name in FD record
Form Code		LO always generated
Field Name	ITEM, GROUP	local-name, ITEM or GROUP name, or alias
Delete?		Blank
Segment Number	MARKIV-SEGMENT, SEGMENT, GROUP, ITEM (see Note 1)	Number assigned to containing segment (see section 4.2, remarks 10 to 12)
Level Number	MARKIV-SEGMENT, SEGMENT, GROUP, ITEM (see Note 1)	Level number assigned to containing segment (see section 4.2, remarks 10 to 12)
Field Location	ITEM, GROUP	From length of preceding ITEM members directly or indirectly contained in segment being defined
Field Length	ITEM, GROUP	From data definition of ITEM or data definitions of contained ITEMS
Field Type	ITEM, GROUP	For an ITEM member, from data definition: C for HELD-AS CHARACTER Z for HELD-AS NUMERIC P for HELD-AS PACKED F for HELD-AS BINARY E for HELD-AS FLOATING For a GROUP member, C always used

(continued)

(continued)

(continued)

MARK IV/DATAMANAGER Correspondence: LO Field Attributes Record		
MARK IV File Definition Form Entry	Relevant DATAMANAGER Member Type(s)	Remarks
Segment Key	ITEM, GROUP	<b>Blank if field not a segment key.</b> Number 1 or 2 or 3 for a segment key field (see section 4.2, remark 13).
Field Rounding	ITEM	Y entered if <b>keyword ROUNDED</b> in form-description clause of ITEM member's data definition. Blank for any other ITEM member.
Decimal Places	ITEM	Number from 0 to 9 from ITEM member's data definition
Count Field for Segment Number	ITEM	Blank if not a count field. Otherwise, segment number of segment for which this is a count field (see Note 2).
This segment Occurs n Times		Blank if this segment does not occur a fixed number of times. <b>Otherwise</b> , the fixed number of times this segment occurs for each occurrence of its controlling segment (see <b>Note 2</b> ).
Output Edit	ITEM'	From REPORTED-AS PICTURE clause of ITEM member's data definition. Blank if no REPORTED-AS PICTURE clause.
Floating		<b>\$, £, +, -, or Z</b> if appearing as leading character in REPORTED-AS PICTURE-clause. Blank otherwise.

(continued)

(continued)

**(continued)**

MARK IV/DATAMANAGER Correspondence: LO Field Attributes Record		
MARK IV File Definition Form Entry	Relevant DATAMANAGER Member Type(s)	Remarks
Filling		Value specified or defaulted for the keyword FCHAR in DGMIV macro, if senior *(s) specified in REPORTED-AS PICTURE clause. Blank otherwise.
Trailing		+, -, C, or D if +, -, CR, or DB, respectively, appears as trailing character in REPORTED-AS PICTURE clause. Value of the keyword ECHAR in DGMIV macro, if S appears as trailing character in REPORTED-AS PICTURE clause. Blank otherwise.
Edit length		Blank

Notes:

- 1 Member type of the segment-forming member which directly or indirectly contains the relevant field member
- 2 From the CONTAINS clause in a relevant FILE MARKIV member's data definition, or from the CONTAINS clause in the data definition of a relevant FILE member or one of its directly or indirectly contained GROUP members.

5  
 (continued)

MARK IV/DATAMANAGER Correspondence: Ln Column Heading Record (n=1,2,...,9)		
MARK IV File Definition Form Entry	Relevant DATAMANAGER Member Type(s)	Remarks
File Name	FILE, FILE MARKIV, IMS-DATABASE	Same as File Name in FD Record
Form Code		L1, L2, ..., L9, depending upon the number of preceding Ln records appearing for this field
Field Name	ITEM	Same as Field Name in preceding L0 record for the ITEM member
Column Heading Text	ITEM	From HEADINGS clause in REPORTED-AS version in ITEM member's data definition (no Ln record appears for an ITEM member defined without a REPORTED-AS version that has a HEADINGS clause). DATAMANAGER automatically inserts required delimiters if CENTRHD=NO applies in DGMIV macro.

**App.1.1**           IMPLEMENTATION OF THE MACRO DGMIV

The macro DGMIV enables the generation of MARK IV File Definition forms by the PRODUCE command of DATAMANAGER to be tailored to conform to a particular installation's standards.

The macro DGMIV is supplied as a source module on the installation magnetic tape. The table in App.1.2 lists the keywords of this macro, for which values can be specified when DATAMANAGER is installed. If the supplied default values of all of these keywords are acceptable, no further action need be taken in respect of this macro. If any values are to be changed, the procedure described in section 2.1 of the Installation in OS Environments or Installation in DOS Environments manual must be followed. The macro assembles as the DATAMANAGER module DFU15.

**App.1.2**           DGMIV MACRO KEYWORD DEFINITIONS

The keywords of the macro DGMIV are defined in the following table.

The Macro DGMIV: Keywords Specifiable on Installation			
Keyword	Specifies	Default Value	Alternative Value
ACHAR	The hexadecimal values of any additional characters that are to be accepted for output in names produced by the Source Language Generation facility, to enable characters not in the standard source language character set to be output (see Note 1)	No default	Any valid hexadecimal value, or a sub-list of such values
ACSMETH	The type of output source library data set generated by a PRODUCE command	BPAM	QSAM
ALIAS	Whether MARK IV File Names and Field Names are to be based on relevant members' aliases	NO	YES
ATRUNK	The character part of an ALPHABETIC ITEM FILLER name	'ALPHA-FILLER'	'name' (see Note 2)
BTRUNK	The character part of a, CHARACTER ITEM FILLER name	'BIN-FILLER'	'name' (see Note 2)
CENTRHD	Whether Column Heading Text entries in Ln records for a field are to be centred	NO	YES
CTRUNK	The character part of a CHARACTER ITEM FILLER name	'FILLER'	'name' (see Note 2)
DDNAME	The name of the output source library data set	'GENLIB'	A delimited string of 1 to 8 characters

(continued)

App.1.2  
(continued)

(continued)

The Macro DGMIV: Keywords Specifiable on Installation			
Keyword	Specifies	Default Value	Alternative Value
DESC	Maximum number of character strings of DATAMANAGER DESCRIPTION clauses to be used to generate AA comment records	0	Up to 32767 or ALL
DLIMIT	MARK IV delimiting character	'#'	Any character, within quotes
DTRUNK	The character part of a PACKED-DECIMAL ITEM FILLER name	'DEC-FILLER'	'name' (see Note 2)
ESIGN	Trailing character to be used in MARK IV editing when a trailing S appears in the REPORTED-AS PICTURE clause of a DATAMANAGER ITEM member	')'	Any character, within quotes
FCHAR	Filling character to be used in MARK IV editing when senior *(s) appear in the REPORTED-AS PICTURE clause of a DATAMANAGER ITEM member	'*'	Any character, within quotes
FDGLOS	MARK IV FD Glossary code	'B'	'A', 'N', '1', or ''
FICOM	Whether AA comment records are required from NOTE and/or DESCRIPTION clauses specified in FILE or FILE MARKIV members' data definitions	YES	NO
FTRUNK	The character part of a FLOATING-POINT ITEM FILLER name	'FLOAT-FILLER'	'name' (see Note 2)

(continued1)

App. 1.2  
 (continued)

(continued)

The Macro DGMIV: Keywords Specifiable on Installation			
Keyword	Specifies	Default Value	Alternative Value
GFNL	The number of digits in the number part of a GROUP FILLER name	0	Up to 14
GRCOM	Whether AA comment records are required from NOTE and/or DESCRIPTION clauses specified in GROUP members' data definitions	YES	NO
GROUP	Whether LO records are to be generated from GROUP members directly or indirectly contained in segment-forming members	YES	NO
GTRUNK	The character part of a GROUP FILLER name	'GROUP-FILLER'	'name' (see Note 2)
IFNL	The number of digits in the number part of an ITEM FILLER name	0	Up to 14
MDBCOM	Whether AA comment records are required from NOTE and/or DESCRIPTION clauses specified in IMS-DATABASE members' data definitions	YES	NO
IMSGCOM	Whether AA comment records are required from NOTE and/or DESCRIPTION clauses specified in SEGMENT members' data definitions	YES	NO
CNCLEV	Segment number increment	1	Up to 98

(continued)

App. 1. 2  
 (continued)

continued)

The Macro DGMIV: Keywords Specifiable on Installation			
Keyword	Specifies	Default Value	Alternative Value
ITCOM	Whether AA comment records are required from NOTE and/or DESCRIPTION clauses specified in ITEM members' data definitions	YES	NO
KNOWNAS	Whether MARK IV Field Names are to be based on KNOWN-AS local-name specifications assigned to relevant GROUP and ITEM members	NO	YES
LS	Whether LS segment name records are to be generated	YES	NO
MAXSEG	Maximum number of segments to be generated in a segment hierarchy described by any one File Definition form	9	1 to 98
IIVSCOM	Whether AA comment records are required from NOTE and/or DESCRIPTION clauses specified in MARKIV-SEGMENT members' data definitions	YES	NO
IOTE	Maximum number of character strings of DATAMANAGER NOTE clauses to be used to generate AA comment records	0	Up to 32767 or ALL
TRUNK	The character part of a NUMERIC-CHARACTER ITEM FILLER name	'NC-FILLER'	'name' (see Note 2)
C	Whether an RC run control record is to be generated	YES	NO

(continued )

App. 1.2  
(continued)

The Macro DGMIV: Keywords Specifiable on Installation			
Keyword	Specifies	Default Value	Alternative Value
REPBYTE	When two or more LS or LO Field Name entries are based on the same GROUP or ITEM member's name or alias, REPBYTE is the number of trailing characters in the generated eight-character Field Name to be replaced by distinguishing numeric suffixes	2	up to 7
RNDBIT	Whether bit string fields should be generated with byte alignment (see Note 3)	YES	NO
RP1	Image of RP parameter data to be inserted in position 11 onward in the first RP record	' '	A delimited string of up to 20 characters
RP2	Image of RP parameter data to be inserted in position 11 onward in the second RP record	' '	A delimited string of up to 20 characters
RPNO	Number of RP records required	0	1 or 2

Notes :

- 1 The standard Source Language Generation facility output character set for the MARK IV Interface is that defined in the MARK IV data definition language specification. This character set can be extended to allow non-standard characters to be output in names, by entering the hexadecimal value of each required character as a value to ACHAR. The user should ensure that any extra characters that are added to the output character set in this way are used only in ways that are permitted by the software with which DATAMANAGER is used.

**App. 1.2**

(continued)

- 2 name defines part of a member name. It must be stated within single quotes, must not be more than 16 characters in length, and must conform to DATAMANAGER rules for member names stated in Chapter 2 of the User's Guide. The values declared for the keywords **ATRUNK**, **BTRUNK**, **CTRUNK**, **DTRUNK**, **FTRUNK**, **GTRUNK** and **NTRUNK** should correspond to those of the same keywords defined in association with the Automation of Set Up facility for the generation of members from COBOL source statements, if that facility is also used.
- 3 The effect of the **RNDBIT** parameter is overridden by any alignment specification stated in the data definition of any group or file that contains the bit string item.



AMENDMENT  
RECORD

This manual when issued should contain 64 pages (32 sheets), comprising:

Preliminary pages	6 pages ( 3 sheets)
Chapter 1	2 pages ( 1 sheet)
Chapter 2	6 pages ( 3 sheets)
Chapter 3	10 pages ( 5 sheets)
Chapter 4	22 pages (11 sheets)
Chapter 5	8 pages ( 4 sheets)
Appendix 1	8 pages ( 4 sheets)
Amendment Record	2 pages ( 1 sheet)

(Blank pages occurring at the end of chapters or appendices are included in the page counts.)

Please check that all pages are present immediately on receipt. If any pages are missing, please inform your local MSP Product Supplier without delay.

If any updates to the information contained in this manual become necessary, additional or replacement pages will be issued. Such pages will be accompanied by a front sheet, the amendment list, detailing the action required to update the manual. On completion of the updating, the amendment list should be filed behind this page. An entry should be made in the table overleaf to record the action taken.

Each page of this manual is annotated (immediately above the page reference) with the publication date of the page. Update pages will be similarly annotated. It is thus always possible to check that the latest version of a page is held.

Amendment List No.	Date of Incorporation	Initials	Notes
1 } 2 }	08.81	MSP	Incorporated in August 1981 reprint
3	12.82	MSP	Incorporated in December 1982 reprint

12 82





ASG Worldwide Headquarters Naples Florida USA | [asg.com](http://asg.com)