



## PUBLICATIONS UPDATE

Operating System/3 (OS/3)

Report Program Generator II (RPG II) User Guide

UP-8067 Rev. 6-E

This Library Memo announces the release and availability of Update E to "SPERRY® Operating System/3 (OS/3) Report Program Generator II (RPG II) User Guide", UP-8067 Rev. 6.

This manual is one of a series describing the SPERRY Report Program Generator II. Other manuals that describe RPG II are:

- Report Program Generator II (RPG II) Programmer Reference, UP-8044
- Report Program Generator II (RPG II) Summary, UP-8253
- Report Program Generator II (RPG II) Editor User Guide/Programmer Reference, UP-9981
- Information Management System (IMS) Action Programming in RPG II User Guide, UP-9206.

RPG II is a programming language used in business data processing applications. You code programs by making entries on preprinted forms and entering the statements at a workstation with the OS/3 RPG II editor. The number and type of forms you use depend on the application. The auto report feature simplifies writing RPG II programs and output specifications.

This manual describes RPG II concepts and data handling capabilities, tells how to complete the RPG II specifications forms and process an RPG II program, and explains the simplified output specification using auto report.

This update for release 10.0 describes how to use the figurative constants \*BLANK, \*BLANKS, \*ZERO, and \*ZEROS as factors in a calculations specification.

All other changes are minor technical or editorial corrections.

Copies of Update E are now available. You can order the update only or the complete manual with the update through your local Sperry representative. To receive only the update, order UP-8067 Rev. 6-E. To receive the complete manual, order UP-8067 Rev. 6.

### LIBRARY MEMO ONLY

Mailing Lists  
BZ, CZ, and MZ

### LIBRARY MEMO AND ATTACHMENTS

Mailing Lists AF01, A00, A10, B00,  
B10, MBW, 18, 18U, 19, 19U, 20, 20U,  
21, 21U, 28U, 29U, 75, 75U,  
76, and 76U  
(Package E to UP-8067 Rev. 6,  
97 pages plus Memo)

### THIS SHEET IS

Library Memo for  
UP-8067 Rev. 6-E

RELEASE DATE:

May 1986



**PUBLICATIONS  
UPDATE**

**Operating System/3 (OS/3)**

**Report Program Generator II  
(RPG II)**

**User Guide**

**UP-8067 Rev. 6-D**

**This Library Memo announces the release and availability of Update Package D to "SPERRY Operating System/3 (OS/3) Report Program Generator II (RPG II) User Guide", UP-8067 Rev. 6.**

RPG II is a programming language used in business data processing applications. You code programs by making entries on preprinted forms and entering the statements at a workstation with the OS/3 RPG II editor. The number and type of forms you use depend on the application. The auto report feature simplifies writing RPG II programs and output specifications.

This manual describes RPG II concepts and data handling capabilities, tells how to complete the RPG II specifications forms and process an RPG II program, and explains the simplified output specification using auto report.

Update D documents:

- The REFER operation, which lets you retrieve records from indexed multikey files in IMS action programs.
- Additional compilation time error messages.

All other changes are minor technical or editorial corrections.

Copies of Update Package D are now available for requisitioning. Either the update package only or the complete manual with the update package may be requisitioned by your local Sperry representative. To receive only the update package, order UP-8067 Rev. 6-D. To receive the complete manual, order UP-8067 Rev. 6.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ, and MZ	Mailing Lists A00, A10, B00, B10, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76, and 76U (Package D to UP-8067 Rev. 6, 49 pages plus Memo)	Library Memo for UP-8067 Rev. 6-D  RELEASE DATE:  December, 1984





<b>PUBLICATIONS UPDATE</b>	
<b>Operating System/3 (OS/3)</b>	
<b>Report Program Generator II (RPG II)</b>	
<b>User Guide</b>	
UP-8067 Rev. 6-C	

This Library Memo announces the release and availability of Updating Package C to "SPERRY Operating System/3 (OS/3) Report Program Generator II (RPG II) User Guide", UP-8067 Rev. 6.

This update for release 8.2 incorporates additional information about the following features:

- Field location description
- Interactive data entry screen formats
- PROGID keyword parameter
- Changed and expanded diagnostic messages

Sample programs for creating, updating, and adding records to a keyed MIRAM file are added. The new sample programs and other corrections and expanded descriptions apply to the software prior to the 8.2 release.

Copies of Updating Package C are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry representative. To receive only the updating package, order UP-8067 Rev. 6-C. To receive the complete manual, order UP-8067 Rev. 6.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
<p>Mailing Lists BZ, CZ and MZ</p>	<p>Mailing Lists A00, A10, B00, B10, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76 and 76U (Package C to UP-8067 Rev. 6, 85 pages plus Memo)</p>	<p>Library Memo for UP-8067 Rev. 6-C</p> <hr/> <p>RELEASE DATE:  February, 1984</p>

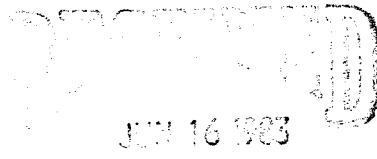


**PUBLICATIONS  
UPDATE**

Operating System/3 (OS/3)

Report Program Generator II  
(RPG II)

User Guide



William  
A. J. J. J.

UP-8067 Rev. 6-B

This Library Memo announces the release and availability of Updating Package B to "SPERRY Operating System/3 (OS/3) Report Program Generator II (RPG II) User Guide", UP-8067 Rev. 6.

This update includes the following:

- a reference to assigned DVC numbers;
- a correction as to how many continuation specifications may be used (five);
- an expanded explanation of the AR068 diagnostic message;
- data structures that begin on a double-word boundary in main storage;
- a correction of the number of workstations (255) used with workstation files; and
- an explanation for placement of input format specifications for data structures.

All other changes are corrections or expanded descriptions.

Copies of Updating Package B are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry representative. To receive only the updating package, order UP-8067 Rev. 6-B. To receive the complete manual, order UP-8067 Rev. 6.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ, and MZ	Mailing Lists A00, A10, B00, B10, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76 and 76U (Package B to UP-8067 Rev. 6, 94 pages plus Memo)	Library Memo for UP-8067 Rev. 6-B  RELEASE DATE:  June, 1983



*clout.*

**RECEIVED**  
 APR - 5 1983  
 District of Columbia  
 Administration

<b>PUBLICATIONS UPDATE</b>
Operating System/3 (OS/3)
Report Program Generator II (RPG II)
User Guide
UP-8067 Rev. 6-A

This Library Memo announces the release and availability of Updating Package A to "SPERRY UNIVAC Operating System/3 (OS/3) Report Program Generator II (RPG II) User Guide", UP-8067 Rev. 6.

This update incorporates additional information about RPG II for release 8.0:

- Using RPG II interactively
- /SPACE n directive
- Error messages
- Defining a PAGE field
- Job control for interactive data entry
- SETK operation
- RCB continuation line for MIRAM files
- File retrieval
- Data structures
- Multikey MIRAM files
- Adding/deleting records
- MIRAM parameter

All other changes are corrections or expanded descriptions applicable to features present in RPG II prior to the 8.0 release.

Copies of Updating Package A are now available for requisitioning. Either the updating package only or the complete manual with the updating package may be requisitioned by your local Sperry Univac representative. To receive only the updating package, order UP-8067 Rev. 6-A. To receive the complete manual, order UP-8067 Rev. 6.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists A00, A10, B00, B10, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76 and 76U (Package A to UP-8067 Rev. 6. 173 pages plus Memo)	Library Memo for UP-8067 Rev. 6-A
		RELEASE DATE: March, 1983



**PUBLICATIONS  
REVISION**

Operating System/3 (OS/3)

Report Program Generator II  
(RPG II)

User Guide

UP-8067 Rev. 6

This Library Memo announces the release and availability of "SPERRY UNIVAC® Operating System/3 (OS/3) Report Program Generator II (RPG II) User Guide", UP-8067 Rev. 6.

This revision describes the following RPG II features for release 8.0:

- Error file processing
- AUTO, AUTRPG, AUTRPGL, and AUTRPGLG jprocs
- MIRAM multikey support
- Interactive data entry
- Currency sign other than dollar sign
- SHTDN, SETK, and NEXT operations
- /EJECT, /SPACE, and /TITLE directives
- Function key indicators
- Error log access
- Data structures allowing multiple definitions of data
- Multiple workstation support
- Workstation file continuation statements

All other changes are corrections or expanded descriptions applicable to RPG II prior to the 8.0 release.

Destruction Notice: If you are going to OS/3 release 8.0, use this revision and destroy all previous copies. If you are not going to OS/3 release 8.0, retain the copy you are now using and store this revision for future use.

Copies of UP-8067 Rev. 5 and UP-8067 Rev. 5-A will be available for 6 months after the release of 8.0. Should you need additional copies of this edition, you should order them within 90 days of the release of 8.0. When ordering the previous edition of a manual, be sure to identify the exact revision and update packages desired and indicate that they are needed to support an earlier release.

Additional copies may be ordered by your local Sperry Univac representative.

LIBRARY MEMO ONLY	LIBRARY MEMO AND ATTACHMENTS	THIS SHEET IS
Mailing Lists BZ, CZ and MZ	Mailing Lists A00, A10, B00, B10, 18, 18U, 19, 19U, 20, 20U, 21, 21U, 28U, 29U, 75, 75U, 76, and 76U (Cover and 749 pages)	Library Memo for UP-8067 Rev. 6  RELEASE DATE:  September, 1982





**UNISYS**

**OS/3**

**Report Program  
Generator II  
(RPG II)**

**Programming  
Guide**

Relative to Release  
Level 11.0

Priced Item

August 1987

Printed in U S America  
UP-8067 Rev. 6



**UNISYS**

**OS/3**

**Report Program  
Generator II  
(RPG II)**

**Programming  
Guide**

Copyright© 1987 Unisys Corporation  
All Rights Reserved  
Unisys is a trademark of Unisys Corporation.  
Previous Title: OS/3 Report Program Generator II (RPG II)  
User Guide

Relative to Release  
Level 11.0

Priced Item

August 1987

Printed in U S America  
UP-8067 Rev. 6

The names, places, and/or events used in this publication are not intended to correspond to any individual, group, or association existing, living, or otherwise. Any similarity or likeness of the names, places, and/or events with the names of any individual living or otherwise, or that of any group or association is purely coincidental and unintentional.

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THE DOCUMENT. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed Program Product License or Agreement to purchase or lease equipment. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such License or Agreement. Unisys cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

FASTRAND, ✦SPERRY, SPERRY✦UNIVAC, SPERRY, SPERRY UNIVAC, UNISCOPE, UNISERVO, UNIS, UNIVAC, and ✦ are registered trademarks of Unisys Corporation. ESCORT, PAGEWRITER, PIXIE, PC/IT, PC/HT, PC/microIT, SPERRYLINK, and USERNET are additional trademarks of Unisys Corporation. MAPPER is a registered trademark and service mark of Unisys Corporation. CUSTOMCARE is a service mark of Unisys Corporation.

## PAGE STATUS SUMMARY

ISSUE: Update F – UP-8067 Rev. 6  
RELEASE LEVEL: 11.0 Forward

Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level	Part/Section	Page Number	Update Level
Cover		F	5 (cont)	19	F	PART 3	Tab Breaker	E
Title Page/Disclaimer		F*		20, 21	F		Title Page	Orig.**
PSS	1, 2	F		22	F	12	Tab Breaker	E
Preface	1 thru 4	E		23	F		1 thru 10	Orig.
Contents	1	E		24, 25	A		11	C
	2	Orig.		26	D		12	F
	3	D		26a	D		12a	C
	4	C		27	C		13 thru 26	Orig.
	5	F		28	A		27, 28	E
	6	F		28a thru 28c	A	13	Tab Breaker	E
	7	Orig.		29	Orig.		1, 2	Orig.
	8	C	6	30 thru 33	B		3	B
	9	Orig.					4 thru 8	Orig.
	10	A		Tab Breaker	E		9, 10	B
	11	Orig.		1 thru 3	Orig.		11 thru 13	Orig.
	12	A		4, 5	B		14, 15	B
	13 thru 16	Orig.		6 thru 12	Orig.		16, 17	Orig.
	17, 18	C		13, 14	C		18, 19	B
	19 thru 21	E		14a	C		20 thru 22	Orig.
	22 thru 25	C		15 thru 26	Orig.		23, 24	C
	26	C					25 thru 27	B
PART 1	Tab Breaker	E	7	Tab Breaker	E		28	D
	Title Page	Orig.**		1 thru 4	Orig.		29 thru 43	Orig.
1	Tab Breaker	E		5, 6	E		44	B
	1 thru 27	A		7 thru 19	A		45 thru 48	Orig.
2	Tab Breaker	E		20, 21	Orig.		49	D
	1 thru 12	Orig.		22	A		50 thru 54	Orig.
PART 2	Tab Breaker	E		23 thru 27	Orig.		55	A
	Title Page	Orig.**		28	A		56 thru 69	Orig.
3	Tab Breaker	E		30 thru 33	Orig.		70	B
	1	A		34	F		71 thru 77	Orig.
	2 thru 4	Orig.		34a	D		78	D
	5	A		35 thru 37	Orig.		79 thru 82	Orig.
	6	Orig.		38, 39	A		83	F
4	Tab Breaker	E		40 thru 46	Orig.		84	Orig.
	1	A					85 thru 87	A
	2 thru 8	Orig.		8	Tab Breaker	E	88, 89	B
5	Tab Breaker	E		1 thru 3	Orig.		90	A
	1, 2	Orig.		4	F		90a, 90b	D
	3	C		5 thru 12	A		91	B
	4	Orig.		13	Orig.		92	D
	5	A		14	A		93	Orig.
	6	Orig.					94	C
	7	A		9	Tab Breaker	E	94a thru 94d	C
	8 thru 12	Orig.		1	A		95	C
	13 thru 16	D		2 thru 4	Orig.		96, 97	Orig.
	17	Orig.		5	F		98	B
	18	A		6 thru 10	Orig.		99	Orig.
			10	Tab Breaker	E	14	Tab Breaker	E
				1 thru 5	Orig.		1 thru 8	Orig.
			11	Tab Breaker	E		9	B
				1 thru 7	Orig.		10, 11	Orig.

\*New Pages

\*\*Deleted Pages

All the technical changes are denoted by an arrow ( $\Rightarrow$ ) in the margin. A downward pointing arrow ( $\Downarrow$ ) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow ( $\Uparrow$ ) is found. A horizontal arrow ( $\Rightarrow$ ) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.

**PAGE STATUS SUMMARY**

**ISSUE: Update F – UP-8067 Rev. 6**  
**RELEASE LEVEL: 11.0 Forward**

Part/Section	Page Number	Update Level
14 (cont)	12 12a 13, 14	A A Orig.
15	Tab Breaker 1 thru 3 4 5, 6	E Orig. F Orig.
16	Tab Breaker 1 2, 3 4, 5 6	E F Orig. F E
PART 4	Tab Breaker Title Page	E Orig.**
17	Tab Breaker 1 thru 5 6 7 thru 12 13 14 thru 24	E Orig. B Orig. B Orig.
PART 5	Tab Breaker Title Page	E Orig.**
18	Tab Breaker 1, 2 3 4 thru 7 8 8a 9 thru 16 17 18, 19 20 21 22, 23 24 25 26 thru 31 32 33, 34 35 thru 39 40 41	E Orig. A Orig. A A Orig. A C Orig. C Orig. C Orig. D Orig. C Orig. C Orig. B Orig.
PART 6	Tab Breaker Title Page	E Orig.**
19	Tab Breaker 1 thru 4 5, 6	E Orig. A

Part/Section	Page Number	Update Level
19 (cont)	7, 8 9 10 thru 13 14, 15 16 thru 22 23 24 thru 51 52 53 thru 56 57 58 thru 81 82 83 thru 90 91 92, 93 94 95 thru 99 100 101 thru 108	Orig. A Orig. A Orig. A Orig. B Orig. A Orig. B Orig. A Orig. B Orig. A Orig.
Appendixes	Tab Breaker Title Page	E Orig.**
Appendix A	Tab Breaker 1 thru 8	E Orig.
Appendix B	Tab Breaker 1 thru 3 4 5 thru 12 13, 14 15, 16 17, 18 19 20 21 thru 23 24 25 thru 27 28, 29 30 thru 32 33 34 35, 36 37 thru 40 41 42 thru 46 47 48 thru 57 58 59 thru 67	E Orig. B Orig. B Orig. B Orig. C Orig. B Orig. C Orig. B Orig. B Orig. D C
Appendix C	Tab Breaker 1 thru 8 9 10 thru 55	E Orig. F Orig.

Part/Section	Page Number	Update Level
Appendix D	Tab Breaker 1 2 3 4 5 thru 10 11 12 thru 25 26, 27 28 29, 30 31	E Orig. A D A Orig. C Orig. A B A D
Appendix E	Tab Breaker 1 thru 4	E Orig.
Appendix F	Tab Breaker 1 2 3, 4	E E Orig. E
Appendix G	Tab Breaker 1 thru 6 7 8 thru 16 17 18 19 20	E Orig. C Orig. B Orig. C Orig.
Index	Tab Breaker 1 2 3 4 5 6 7 thru 9 10 11 12 13, 14 15 16 17 18, 19 20 21 22 23 24 25 thru 26 27 28 29	E F Orig. E Orig. A Orig. A C Orig. C Orig. Orig. B C Orig. C C Orig. A Orig. C B D C Orig. F A E
User Comment Form		

\*New Pages      \*\*Deleted Pages

All the technical changes are denoted by an arrow (⇒) in the margin. A downward pointing arrow (⇓) next to a line indicates that technical changes begin at this line and continue until an upward pointing arrow (⇑) is found. A horizontal arrow (⇒) pointing to a line indicates a technical change in only that line. A horizontal arrow located between two consecutive lines indicates technical changes in both lines or deletions.

# Preface

This manual is one of a series designed to instruct and guide the programmer in the use of the SPERRY Operating System/3 (OS/3). This manual specifically describes the OS/3 Report Program Generator (RPG II) and its effective use. Its intended audience is the novice programmer with a basic knowledge of data processing, but with limited programming experience, and the RPG II programmer whose experience is limited to systems other than those of Sperry.

Two other manuals are available that cover OS/3 RPG II; one is an introductory manual and the other is a programmer reference (PR) manual. The introductory manual, UP-8004 (current version), briefly describes OS/3 RPG II and its facilities. The PR, UP-8044 (current version), provides the characteristics of OS/3 RPG II in skeletal form and is intended as a quick-reference document for the programmer experienced in the use of OS/3 RPG II.

This user guide is subdivided into the following parts:

- PART 1. RPG II CONCEPTS AND DATA HANDLING CAPABILITIES

Introduces you to RPG II in terms of what it is, the way in which you use it, the general program logic, how your files can be organized, and the record and data formats that you can use with RPG II to implement your data processing applications.

- PART 2. RPG II SPECIFICATIONS FORMS

Describes the forms that you use to define your data processing application to RPG II.

- PART 3. USING RPG II

Describes the various indicators used to control RPG II, the file processing techniques that you can use to process your files, printing techniques, the method used to incorporate subroutines within your program, the way to utilize external subroutines, and how to write IMS action programs in RPG II.

- PART 4. RPG II TELECOMMUNICATIONS

Describes how you can communicate with remote terminals and use RPG II from a remote site.

- PART 5. PROCESSING YOUR RPG II PROGRAM

Describes how to compile, link, and execute your programs.

- PART 6. AUTO REPORT

Describes how to use auto report to produce formatted reports.

- PART 7. APPENDIXES

Contains additional information pertaining to RPG II.

Each of the foregoing parts consists of one or more sections that cover the different aspects of the subject matter covered in each part.

Other current OS/3 publications, referenced in this manual, are useful to the programmer working with RPG II. They are:

### **System 80**

- RPG II editor user guide/programmer reference, UP-8803

Describes how to program at a workstation.

- Information management system (IMS) action programming in RPG II user guide, UP-9206

Describes how to write action programs in RPG II.

- Screen format services concepts and facilities, UP-8802

Describes the screens and formats used on workstation terminals.

- Interactive services commands and facilities user guide/programmer reference, UP-8845

Describes the commands and operating procedure for workstation terminals.

- Consolidated data management macroinstructions user guide/programmer reference, UP-9979

Describes the data management macroinstructions.



- **SORT3 user guide/programmer reference, UP-8836**  
Describes how to sort records and manipulate the fields within them.
- **Data utilities user guide/programmer reference, UP-8834**  
Describes how to reproduce and maintain data files.
- **System installation user guide/programmer reference, UP-8839**  
Describes how to generate, install, and tailor your operating system.
- **System messages programmer/operator reference, UP-8076**  
Lists and describes the system console messages issued during compilation.
- **Job control user guide, UP-8065**  
Provides information on the format and usage of job control statements.
- **ICAM concepts and facilities, UP-8194**  
Describes how to prepare the system to receive or send data to another system.
- **ICAM direct data interface (DDI) user guide, UP-8459**  
Describes the instructions needed to write a communications program for transmission to another system.
- **ICAM utilities user guide, UP-8552**  
Describes how to make the 90/30 system operate as a terminal.

### **Series 90 Systems**

- **Basic data management user guide, UP-8068**  
Describes the data management macroinstructions.
- **Sort/merge user guide, UP-8342**  
Describes how to sort records and manipulate the fields within them.
- **Data utilities user guide/programmer reference, UP-8069**  
Describes how to reproduce and maintain data files.

- System installation user guide/programmer reference, UP-8074  
Describes how to generate, install, and tailor your operating system.
- System messages programmer/operator reference, UP-8076  
Lists and describes the system console messages issued during compilation.
- Job control user guide, UP-8065  
Provides information on the format and usage of job control statements.
- ICAM concepts and facilities, UP-8194  
Describes how to prepare the system to receive or send data to another system.
- ICAM direct data interface (DDI) user guide, UP-8459  
Describes the instructions needed to write a communications program for transmission to another system.
- ICAM utilities user guide, UP-8552  
Describes how to make the 90/30 system operate as a terminal.

Throughout this manual, reference is made to the Series 90 environment. This is a catch-all phrase for the 90/25, 90/30, 90/30 B, and 90/40 operating systems.

# Contents

## USER COMMENT FORM

## PAGE STATUS SUMMARY

## PREFACE

## CONTENTS

### PART 1. RPG II CONCEPTS AND DATA HANDLING CAPABILITIES

#### 1. CONCEPTS

1.1.	WHAT IS RPG II?	1-1
1.2.	HOW TO USE RPG II WITH CARDS	1-1
1.3.	HOW TO USE RPG II ON A WORKSTATION	1-5
1.4.	RPG II GENERAL PROGRAM LOGIC	1-6
1.4.1.	Detail Time	1-7
1.4.2.	Total Time	1-7
1.4.3.	General Program Cycle	1-7

#### 2. RPG II FILE AND DATA HANDLING CAPABILITIES

2.1.	FILE CHARACTERISTICS	2-1
2.1.1.	File Type	2-1
2.1.2.	File Designation	2-2
2.2.	FILE ORGANIZATION	2-2
2.2.1.	Sequential Files	2-3
2.2.2.	Indexed Sequential Files	2-3
2.2.3.	Direct Files	2-3
2.2.4.	MIRAM Files	2-7
2.2.5.	IRAM Files	2-7
2.3.	FILE FORMAT	2-7
2.3.1.	Fixed-Length Records	2-8
2.3.2.	Variable-Length Records	2-9

2.4.	DATA FORMATS	2-11
2.4.1.	Alphanumeric Format	2-11
2.4.2.	Binary Format	2-11
2.4.3.	Packed Numeric Format	2-12
2.4.4.	Unpacked Numeric Format	2-12

## PART 2. RPG SPECIFICATIONS FORMS

### 3. GENERAL SPECIFICATION INFORMATION

3.1.	TYPE AND NUMBER OF SPECIFICATIONS FORMS REQUIRED FOR A PROGRAM	3-1
3.2.	COMMON FIELDS	3-2
3.2.1.	Page Number (Columns 1 and 2)	3-2
3.2.2.	Line Number (Columns 3 through 5)	3-2
3.2.3.	Form Type (Column 6)	3-4
3.2.4.	Compiler Directives (Columns 7 through 74)	3-5
3.2.5.	Comments (Columns 7 through 74)	3-5
3.2.6.	Program Identification (Columns 75 through 80)	3-6

### 4. CONTROL CARD SPECIFICATIONS FORM

4.1.	GENERAL DESCRIPTION	4-1
4.2.	FORM ENTRIES	4-1
4.2.1.	Compilation Mode (Column 7)	4-1
4.2.2.	Error Analysis Dump (Column 8)	4-3
4.2.3.	Operator Control (Column 9)	4-3
4.2.4.	Generate Debug Code (Column 15)	4-4
4.2.5.	Currency Sign (Column 18)	4-4
4.2.6.	Inverted Print (Column 21)	4-5
4.2.7.	Alternate Collating Sequence (Column 26)	4-5
4.2.8.	Binary Search (Column 31)	4-5
4.2.9.	Sign Handling (Column 40)	4-6
4.2.10.	Forms Alignment (Column 41)	4-6
4.2.11.	Indicator Initialization (Column 42)	4-6
4.2.12.	File Translation (Column 43)	4-7
4.2.13.	CCA Name (Columns 70 through 73)	4-7
4.2.14.	Subroutine or Action Program (Column 74)	4-7
4.2.15.	Program Identification (Columns 75 through 80)	4-7
4.3.	EXAMPLES OF ENTRIES ON THE CONTROL CARD SPECIFICATIONS FORM	4-8

### 5. FILE DESCRIPTION SPECIFICATIONS FORM

5.1.	GENERAL DESCRIPTION	5-1
------	---------------------	-----

<b>5.2</b>	<b>FORM ENTRIES</b>	5-1
5.2.1.	File Name (Columns 7 through 13)	5-1
5.2.2.	File Type (Column 15)	5-3
5.2.3.	File Designation (Column 16)	5-4
5.2.4.	End of File (Column 17)	5-6
5.2.5.	Sequence (Column 18)	5-6
5.2.6.	File Format (Column 19)	5-7
5.2.7.	Block Length (Columns 20 through 23)	5-7
5.2.8.	Record Length (Columns 24 through 27)	5-10
5.2.9.	File Processing Mode (Column 28)	5-13
5.2.10.	Key or Record Address Field Length (Columns 29 and 30)	5-13
5.2.11.	Record Address Type (Column 31)	5-13
5.2.12.	File Organization (Column 32)	5-14
5.2.13.	Overflow Indicator (Columns 33 and 34)	5-16
5.2.14.	Key Field Starting Location (Columns 35 through 38)	5-17
5.2.15.	Extension or Line Counter Code (Column 39)	5-17
5.2.16.	Device (Columns 40 through 46)	5-17
5.2.17.	Labels (Column 53)	5-20
5.2.18.	Continuation Lines (Columns 53 through 69)	5-20
5.2.18.1.	Continuation Line Indicator (Column 53)	5-22
5.2.18.2.	Option (Columns 54 through 59)	5-22
5.2.18.3.	Entry (Columns 60 through 65)	5-23
5.2.18.4.	Key Length (Columns 66 and 67)	5-26
5.2.18.5.	Duplicate Record (Column 68)	5-26
5.2.18.6.	Change Keys (Column 69)	5-26
5.2.19.	Name of Label Exit or Name of User Device Routine (Columns 54 through 59)	5-26a
5.2.20.	Number of Bytes in Main Storage to Be Reserved for Index (Columns 60 through 65)	5-27
5.2.21.	File Addition/Unordered Load (Column 66)	5-28
5.2.22.	Cylinder Overflow Space Percentage (X10) (Column 67)	5-28c
5.2.23.	Number of Extents (Columns 68 and 69)	5-29
5.2.24.	Tape Rewind Option (Column 70)	5-29
5.2.25.	File Conditioners (Columns 71 and 72)	5-30
5.3.	EXAMPLES OF ENTRIES ON THE FILE DESCRIPTION SPECIFICATIONS FORM	5-30

## 6. INPUT FORMAT SPECIFICATIONS FORM

6.1.	GENERAL DESCRIPTION	6-1
6.2.	RECORD IDENTIFICATION ENTRIES (COLUMNS 7 THROUGH 42)	6-1
6.2.1.	File Name (Columns 7 through 13)	6-3
6.2.2.	Sequence (Columns 15 and 16)	6-3
6.2.3.	Number (Column 17)	6-3
6.2.4.	Optional (Column 18)	6-4
6.2.5.	Record Identifying Indicator (Columns 19 and 20)	6-4
6.2.5.1.	Look-Ahead Feature	6-5
6.2.5.2.	Spread Card Feature	6-5

<b>6.2.6. Record Identification Codes (Columns 21 through 41)</b>	6-8
6.2.6.1. Position (Columns 21 through 24, 28 through 31, and 35 through 38)	6-10
6.2.6.2. N=NOT (Columns 25, 32, and 39)	6-10
6.2.6.3. C/Z/D - Characters, Zone, Digit (Columns 26, 33, and 40)	6-10
6.2.6.4. Character (Columns 27, 34, and 41)	6-11
<b>6.2.7. Stacker Select (Column 42)</b>	6-11
<b>6.2.8. AND/OR Relationship (Columns 14 through 16)</b>	6-12
<b>6.3. FIELD DESCRIPTION ENTRIES (COLUMNS 43 THROUGH 70)</b>	6-12
6.3.1. Data Format (Column 43)	6-12
6.3.2. Field Location (Columns 44 through 51)	6-13
6.3.3. Decimal Positions (Column 52)	6-14
6.3.4. Field Name (Columns 53 through 58)	6-14a
6.3.5. Control Level (Columns 59 and 60)	6-15
6.3.6. Matching Fields or Chaining Fields (Columns 61 and 62)	6-17
6.3.6.1. Matching Fields	6-17
6.3.6.2. Chaining Fields	6-18
6.3.7. Field Record Relation (Columns 63 and 64)	6-19
6.3.8. Field Indicators (Columns 65 through 70)	6-20
<b>6.4. EXAMPLES OF ENTRIES ON THE INPUT FORMAT SPECIFICATIONS FORM</b>	6-20

## 7. CALCULATION SPECIFICATIONS FORM

<b>7.1. GENERAL DESCRIPTION</b>	7-1
<b>7.2. CONDITIONS ENTRIES (COLUMNS 7 THROUGH 17)</b>	7-1
7.2.1. Control Level (Columns 7 and 8)	7-1
7.2.2. Indicators (Columns 9 through 17)	7-3
7.2.3. AND/OR Relationship (Columns 7 and 8)	7-5
<b>7.3. CALCULATION ENTRIES (COLUMNS 18 THROUGH 53)</b>	7-6
7.3.1. Factor 1 and Factor 2 (Columns 18 through 27 and Columns 33 through 42)	7-6
7.3.2. Operation (Columns 28 through 32)	7-7
7.3.2.1. Arithmetic Operations	7-7
7.3.2.1.1. Add (ADD)	7-8
7.3.2.1.2. Zero and Add (Z-ADD)	7-8
7.3.2.1.3. Subtract (SUB)	7-9
7.3.2.1.4. Zero and Subtract (Z-SUB)	7-9
7.3.2.1.5. Multiply (MULT)	7-9
7.3.2.1.6. Divide (DIV)	7-10
7.3.2.1.7. Move Remainder (MVR)	7-10
7.3.2.1.8. Square Root (SQRT)	7-12
7.3.2.1.9. Cross-Foot (XFOOT)	7-12
7.3.2.2. Move Operations	7-12
7.3.2.2.1. Move (MOVE)	7-13
7.3.2.2.2. Move Array (MOVEA)	7-16
7.3.2.2.3. Move Left (MOVEL)	7-17
7.3.2.2.4. Move Low Zone to Low Zone (MLLZO)	7-20
7.3.2.2.5. Move Low Zone to High Zone (MLHZO)	7-21
7.3.2.2.6. Move High Zone to Low Zone (MHLZO)	7-22
7.3.2.2.7. Move High Zone to High Zone (MHHZO)	7-22

7.3.2.3.	Compare and Test Operations	7-23
7.3.2.3.1.	Compare (COMP)	7-23
7.3.2.3.2.	Test Bit (TESTB)	7-24
7.3.2.3.3.	Test Numeric (TESTN)	7-25
7.3.2.3.4.	Test Zone (TESTZ)	7-25
7.3.2.4.	Branching and Exit Operations	7-26
7.3.2.4.1.	Branch (GOTO)	7-26
7.3.2.4.2.	Tag (TAG)	7-26
7.3.2.4.3.	Begin Internal Subroutine (BEGSR)	7-27
7.3.2.4.4.	End Internal Subroutine (ENDSR)	7-27
7.3.2.4.5.	Execute Internal Subroutine (EXSR)	7-28
7.3.2.4.6.	Exit to Linked Subroutine (EXIT)	7-29
7.3.2.4.7.	External Subroutine Access to RPG II Program (RLABL)	7-29
7.3.2.4.8.	RPG II Program to External Subroutine Access (ULABL)	7-30
7.3.2.5.	Indicator Setting Operations	7-30
7.3.2.5.1.	Set On (SETON)	7-30
7.3.2.5.2.	Set Off (SETOF)	7-30
7.3.2.6.	Bit Setting Operations	7-31
7.3.2.6.1.	Set Bit On (BITON)	7-31
7.3.2.6.2.	Set Bit Off (BITOF)	7-32
7.3.2.7.	Look-Up Operations	7-32
7.3.2.7.1.	Look-Up (LOKUP)	7-32
7.3.2.8.	File Processing Operations	7-33
7.3.2.8.1.	Retrieve Record from a Chained File (CHAIN)	7-33
7.3.2.8.1A.	Retrieve Record from a Chained File (REFER) – IMS Action Programs Only	7-34 ←
7.3.2.8.2.	Select Key Structure (SETK)	7-34a
7.3.2.8.3.	Read Record (READ)	7-34a
7.3.2.8.5.	Override Normal Record Selection (FORCE)	7-35
7.3.2.8.6.	Display (DSPLY)	7-36
7.3.2.8.7.	Exception Lines (EXCPT)	7-36
7.3.2.8.8.	Debug (DEBUG)	7-37
7.3.2.8.9.	Next Workstation Input (NEXT)	7-37
7.3.2.9.	Time of Day Operations	7-38
7.3.2.9.1.	Time (TIME)	7-38
7.3.2.10.	System Shutdown (SHTDN)	7-38
7.3.2.11.	Operations Summary	7-38
<b>7.3.3.</b>	<b>Result Field (Columns 43 through 53)</b>	<b>7-40</b>
7.3.3.1.	Name (Columns 43 through 48)	7-40
7.3.3.2.	Field Length (Columns 49 through 51)	7-40
7.3.3.3.	Decimal Positions (Column 52)	7-41
7.3.3.4.	Half Adjust (Column 53)	7-41
<b>7.4.</b>	<b>RESULTING INDICATOR ENTRIES (COLUMNS 54 THROUGH 59)</b>	<b>7-42</b>
7.4.1.	+, 1>2, or High (Columns 54 and 55)	7-42
7.4.2.	-, 1<2, or Low (Columns 56 and 57)	7-43
7.4.3.	0, 1=2, or Equal (Columns 58 and 59)	7-43
7.4.4.	Comments (Columns 60 through 74)	7-44
<b>7.5.</b>	<b>EXAMPLES OF ENTRIES ON THE CALCULATION SPECIFICATIONS FORM</b>	<b>7-44</b>

**8. OUTPUT FORMAT SPECIFICATIONS FORM**

<b>8.1.</b>	<b>GENERAL DESCRIPTION</b>	<b>8-1</b>
<b>8.2.</b>	<b>OUTPUT FILE IDENTIFICATION AND CONTROL ENTRIES (COLUMNS 7 THROUGH 31)</b>	<b>8-1</b>
8.2.1.	File Name (Columns 7 through 13)	8-1
8.2.2.	Type (Column 15)	8-3
8.2.3.	Stacker Select/Fetch Overflow (Column 16)	8-3
8.2.4.	AND/OR Relationship (Columns 14 through 16)	8-4
8.2.5.	Adding and Deleting Existing Indexed Sequential File Records and Deleting Transaction Buffers (Columns 16 through 18)	8-4
8.2.6.	Space (Columns 17 and 18)	8-5
8.2.7.	Skip (Columns 19 through 22)	8-5
8.2.8.	Output Indicators - Records (Columns 23 through 31)	8-7
<b>8.3.</b>	<b>FIELD DESCRIPTION AND CONTROL ENTRIES (COLUMNS 23 THROUGH 70)</b>	<b>8-8</b>
8.3.1.	Output Indicators - Fields (Columns 23 through 31)	8-8
8.3.2.	Field Name (Columns 32 through 37)	8-8
8.3.3.	Edit Codes (Column 38)	8-9
8.3.4.	Blank After (Column 39)	8-9
8.3.5.	End Position in Output Record (Columns 40 through 43)	8-10
8.3.6.	Length of Screen Format Name (Columns 42 and 43)	8-10
8.3.7.	Data Format (Column 44)	8-10
8.3.8.	Constant (Columns 45 through 70)	8-11
8.3.9.	Edit Word (Columns 45 through 70)	8-11
8.3.10.	Format Name (Columns 45 through 54)	8-13
<b>8.4.</b>	<b>EXAMPLE OF ENTRIES ON THE OUTPUT FORMAT SPECIFICATIONS FORM</b>	<b>8-13</b>

**9. FILE EXTENSION SPECIFICATIONS FORM**

<b>9.1.</b>	<b>GENERAL DESCRIPTION</b>	<b>9-1</b>
<b>9.2.</b>	<b>FORM ENTRIES</b>	<b>9-1</b>
9.2.1.	Record Sequence of Chaining File (Columns 7 and 8)	9-1
9.2.2.	Number of the Chaining Field (Columns 9 and 10)	9-1
9.2.3.	From File Name (Columns 11 through 18)	9-3
9.2.4.	To File Name (Columns 19 through 26)	9-3
9.2.5.	Table or Array Name (Columns 27 through 32)	9-4
9.2.6.	Number of Entries per Record (Columns 33 through 35)	9-5
9.2.7.	Number of Entries per Table or Array (Columns 36 through 39)	9-5
9.2.8.	Length of Entry (Columns 40 through 42)	9-5
9.2.9.	Data Format (Column 43)	9-6
9.2.10.	Decimal Positions (Column 44)	9-6
9.2.11.	Sequence (Column 45)	9-6
9.2.12.	Table or Array Name (Columns 46 through 51)	9-7
9.2.13.	Length of Entry (Columns 52 through 54)	9-7
9.2.14.	Data Format (Column 55)	9-7
9.2.15.	Decimal Positions (Column 56)	9-8
9.2.16.	Sequence (Column 57)	9-8
9.2.17.	Comments (Columns 58 through 74)	9-8
<b>9.3.</b>	<b>EXAMPLES OF ENTRIES ON THE FILE EXTENSION SPECIFICATIONS FORM</b>	<b>9-8</b>



**10. LINE COUNTER SPECIFICATIONS FORM**

10.1.	GENERAL DESCRIPTION	10-1
10.2.	FORM ENTRIES FOR ALL MODES OTHER THAN THE IBM SYSTEM/3 MODE	10-1
10.2.1.	File Name (Columns 7 through 14)	10-1
10.2.2.	Line Number (Columns 15 through 17, 20 through 22, 25 through 27, ..., 70 through 72)	10-3
10.2.3.	Channel Number (Columns 18 and 19, 23 and 24, 28 and 29, ..., 73 and 74)	10-3
10.3.	FORM ENTRIES FOR THE IBM SYSTEM/3 MODE	10-3
10.3.1.	File Name (Columns 7 through 14)	10-3
10.3.2.	Line Number - Number of Lines per Page (Columns 15 through 17)	10-3
10.3.3.	Channel Number - Form Length Indicator (Columns 18 and 19)	10-3
10.3.4.	Line Number - Overflow Line (Columns 20 through 22)	10-3
10.3.5.	Channel Number - Overflow Line Indicator (Columns 23 and 24)	10-4
10.3.6.	Line Number (Columns 25 through 27, ..., 70 through 72) and Channel Number (Columns 28 and 29, ..., 73 and 74)	10-4
10.4.	EXAMPLES OF ENTRIES ON THE LINE COUNTER SPECIFICATIONS FORM	10-4

**11. TELECOMMUNICATIONS SPECIFICATIONS FORM**

11.1.	GENERAL DESCRIPTION	11-1
11.2.	FORM ENTRIES	11-1
11.2.1.	File Name (Columns 7 through 13)	11-1
11.2.2.	Configuration (Column 15)	11-3
11.2.3.	Type of Station (Column 16)	11-3
11.2.4.	Transparency (Column 19)	11-3
11.2.5.	Switched (Column 20)	11-3
11.2.6.	Remote Terminal (Columns 48 through 51)	11-4
11.2.7.	Permanent Error Indicator (Columns 53 and 54)	11-4
11.2.8.	Wait Time (Columns 55 through 57)	11-5
11.2.9.	Last File (Column 60)	11-5
11.2.10.	Remote Device (Columns 65 through 70)	11-5
11.2.11.	Terminal Name (Columns 71 through 74)	11-6
11.3.	EXAMPLES OF ENTRIES ON THE TELECOMMUNICATIONS SPECIFICATIONS FORM	11-6

**PART 3. USING RPG II****12. CONTROLLING RPG II**

12.1.	GENERAL	12-1
-------	---------	------

<b>12.2. INDICATORS DEFINED ON THE RPG II SPECIFICATIONS FORMS</b>	12-1
12.2.1. Record Identifying Indicator	12-1
12.2.2. Control Level Indicator	12-3
12.2.3. Field Indicator	12-7
12.2.4. Resulting Indicator	12-7
12.2.5. Overflow Indicator	12-9
12.2.6. Function Key Indicator	12-11
<b>12.3. INDICATORS NOT DEFINED ON THE RPG II SPECIFICATIONS FORMS</b>	12-11
12.3.1. External Indicators	12-12
12.3.2. Internal Indicators	12-12
<b>12.4. USING THE INDICATORS</b>	12-14
12.4.1. File Conditioning	12-15
12.4.2. Control Level	12-15
12.4.3. Field Record Relation	12-15
12.4.4. Calculation Conditioning	12-16
12.4.5. Output Conditioning	12-17
<b>12.5. SUMMARY OF INDICATORS</b>	12-19
<b>12.6. SETTING INDICATORS ON VIA THE SYSTEM CONSOLE</b>	12-23
12.6.1. PF Key Subroutine (SUBR89)	12-23
12.6.1.1. Using a PF Key with a Positive Number	12-23
12.6.1.2. Using a PF Key with a Negative Number	12-25
12.6.2. Unsolicited Inquiry Request Subroutine (SUBR95)	12-26
12.6.3. Incorrect Type-Ins	12-28

### 13. FILE PROCESSING METHODS

<b>13.1. GENERAL</b>	13-1
<b>13.2. SEQUENTIAL PROCESSING</b>	13-2
<b>13.3. PROCESSING WITH MATCHING RECORDS</b>	13-2
13.3.1. Primary File	13-2
13.3.2. Specifying Matching Fields	13-2
13.3.3. Record Selection	13-3
13.3.4. Matching Record Indicator (MR)	13-7
<b>13.4. PROCESSING BY CHAINING</b>	13-7
13.4.1. Chaining Indexed Sequential Files	13-8
13.4.2. Chaining Direct Files	13-13
13.4.3. Creating a Direct File Using the CHAIN Operation	13-13
<b>13.5. RECORD ADDRESS FILE PROCESSING</b>	13-17
13.5.1. Processing an Indexed Sequential File Randomly	13-17
13.5.2. Processing an Indexed Sequential File between Limits	13-18
<b>13.6. PROCESSING WITH AN ADDRUT FILE</b>	13-21

---

<b>13.7. DEMAND FILES AND READ OPERATION PROCESSING</b>	13-22
13.7.1. Demand File	13-22
13.7.2. READ Operation	13-22
13.7.3. Using Demand Files and the READ Operation	13-22
13.7.3.1. Processing a Demand File	13-23
13.7.3.2. Processing a Demand File Using the SETLL Operation	13-25
<b>13.8. LOOK-AHEAD FIELDS AND FORCE OPERATION PROCESSING</b>	13-28
13.8.1. Look-Ahead Fields	13-28
13.8.2. FORCE Operation	13-28
13.8.3. Using Look-Ahead Fields and the FORCE Operation	13-29
13.8.3.1. Controlling Record Selection during Multifile Processing	13-29
13.8.3.2. Matching Records from Secondary Files	13-32
<b>13.9. PROCESSING WITH TABLES</b>	13-33
13.9.1. Table Formats	13-33
13.9.2. Table Definition	13-35
13.9.2.1. File Description and File Extension Specifications Form Entries for Table Definition	13-36
13.9.2.2. Table Definition Examples	13-39
13.9.3. Using Tables	13-41
13.9.3.1. Using the LOKUP Operation	13-41
13.9.3.2. Using Data from Table LOKUP	13-43
13.9.3.3. Modifying a Table during Program Execution	13-46
13.9.3.4. Adding Elements to an Existing Table	13-46
<b>13.10. PROCESSING WITH ARRAYS</b>	13-47
13.10.1. Array Formats	13-48
13.10.2. Array Definition	13-49
13.10.2.1. File Description and File Extension Specifications Form Entries for Array Definition	13-50
13.10.2.2. Array Definition Examples	13-53
13.10.3. Using Arrays	13-55
13.10.3.1. Loading an Array with the Input Format Specifications Form Using Fixed Indexes	13-55
13.10.3.2. Loading an Array with the Input Format Specifications Form Using Input Fields as Indexes	13-57
13.10.3.3. Loading an Array with the Calculation Specifications Form	13-58
13.10.3.4. Using the LOKUP Operation with Arrays	13-60
13.10.3.5. Using Arrays to Format Output Records	13-64
13.10.3.6. Using Arrays to Reduce the Number of Statements You Must Write	13-66
13.10.3.7. Using the EXCPT Operation to Write Array Elements	13-69
<b>13.11. PROCESSING WITH ALTERNATE COLLATING SEQUENCE</b>	13-72
13.11.1. RPG II Collating Sequence	13-72
13.11.2. Defining an Alternate Collating Sequence	13-72
13.11.3. Using an Alternate Collating Sequence	13-76
13.11.3.1. Causing Characters to Be Considered Equal	13-76
13.11.3.2. Inserting a Character between Two Existing Characters in the Collating Sequence	13-77
13.11.3.3. Changing the Position of Characters in the Collating Sequence	13-78
<b>13.12. PROCESSING WITH FILE TRANSLATION</b>	13-78
13.12.1. Defining a File Translation Table	13-78
13.12.2. Using a File Translation Table	13-81
13.12.2.1. Using File Translation to Translate Characters in a File	13-81
13.12.2.2. Using File Translation for More than One but Not All Files	13-82

13.13.	PROCESSING WITH WORKSTATIONS	13-82
13.13.1.	File Description Specifications for Workstation Files	13-84
13.13.2.	Input Format Specifications for Workstation Files	13-85
13.13.3.	Calculation Specifications for Workstation Files	13-85
13.13.4.	Output Format Specifications for Workstation Files	13-85
13.13.5.	Input/Output Errors for Workstation Files	13-86
13.13.6.	Sample Program for a Workstation File	13-86
13.14.	KEY SPECIFICATION WITH MIRAM FILES	13-90
13.14.1.	Processing a Multikey File Sequentially by Key	13-90a
13.15.	PROCESSING WITH DATA STRUCTURES	13-91
13.15.1.	Input Format Specifications for Data Structures	13-92
13.15.2.	Input Format Specifications for Subfields of a Data Structure	13-92
13.16.	PROCESSING WITH INTERACTIVE DATA ENTRY	13-94
13.16.1.	File Description Specifications for Interactive Data Entry	13-96
13.16.2.	Input Format Specifications for Interactive Data Entry	13-97

## 14. PRINTING TECHNIQUES

14.1.	GENERAL	14-1
14.2.	PRINTER FORMAT CHART	14-1
14.3.	EDITING	14-3
14.3.1.	Edit Codes	14-5
14.3.1.1.	Simple Edit Codes	14-5
14.3.1.2.	Combined Edit Codes	14-6
14.3.2.	Edit Words	14-6
14.3.2.1.	Edit Word Format	14-8
14.3.2.2.	Using Edit Words	14-8
14.4.	SPECIAL FIELD NAMES	14-10
14.4.1.	D?TE, UPDATE, UDAY, UMONTH, and UYEAR	14-10
14.4.2.	PAGE, PAGE1-7	14-12
14.4.3.	*PLACE	14-12a
14.5.	LINE COUNTER FILES	14-13
14.5.1.	Using Line Counter Files	14-13
14.5.2.	Printing Line Counter Files	14-14

## 15. INCORPORATING SUBROUTINES IN YOUR PROGRAM

15.1.	GENERAL	15-1
15.2.	INTERNAL SUBROUTINES	15-1
15.2.1.	Specifying an Internal Subroutine	15-2
15.3.	EXTERNAL SUBROUTINES	15-4
15.3.1.	Specifying an External Subroutine	15-5
15.4.	RPG II SUBROUTINES	15-6

**16. IMS – RPG II ACTION PROGRAMS**

16.1.	GENERAL	16-1
16.2.	RPG II SPECIFICATIONS FORMS	16-2
16.3.	RESTRICTIONS	16-2
16.4.	IMS – RPG II INTERFACE AREAS	16-4

**PART 4. RPG II TELECOMMUNICATIONS****17. TELECOMMUNICATIONS**

17.1.	GENERAL	17-1
17.2.	USING TELECOMMUNICATIONS	17-1
17.3.	FILE PROCESSING MODES	17-2
17.3.1.	Using Batch Remote Terminals	17-5
17.3.1.1.	Receive-Only Processing Using a DCT 2000 Data Communications Terminal	17-5
17.3.1.2.	Transmit-Only Processing Using Two DCT 1000 Data Communications Terminals	17-7
17.3.1.3.	Transmit a File, Then Receive Another File Using a BSC Remote Terminal	17-9
17.3.2.	Using Interactive Remote Terminals	17-11
17.3.2.1.	UNISCOPE 100 and UNISCOPE 200 Display Terminals	17-11
17.3.2.1.1.	Transmit with Reception of Conversational Reply Using Four UNISCOPE 100 Display Terminals	17-14
17.3.2.2.	Teletypewriter and DCT 500 Series Data Communications Terminals	17-17
17.3.2.2.1.	Transmit with Reception of Conversational Reply Using Two DCT 500 Data Communications Terminals and a Teletypewriter Terminal	17-19
17.4.	NETWORK DEFINITION REQUIREMENTS FOR RPG II TELECOMMUNICATIONS PROGRAMS	17-22
17.5.	PROGRAM EXECUTION	17-24
17.6.	ERROR PROCESSING	17-24

**PART 5. PROCESSING YOUR RPG II PROGRAM****18. COMPILING, LINK EDITING, AND EXECUTING**

18.1.	GENERAL	18-1
18.2.	JOB CONTROL STATEMENTS	18-2
18.2.1.	RPG II Job Control Procedure Call Statements – RPG, RPGL, and RPGLG	18-2
18.2.1.1.	Using the RPG Job Control Procedure Call Statement	18-8
18.2.1.2.	Using a Job Control Procedure Call Statement that Defines Keyword Parameters	18-10
18.2.1.3.	Using the RPGL Job Control Procedure Call Statement	18-12
18.2.1.4.	Using the RPGLG Job Control Procedure Call Statement	18-14

<b>18.2.2. Auto Report Job Control Procedure Call Statements – AUTO, AUTPRG, AUTRPGL, and AUTRPGLG</b>	18-18
18.2.2.1. Using the AUTO Job Control Procedure Call Statement	18-24
18.2.2.2. Using the AUTO Job Control Procedure Call that Defines Keyword Parameters	18-27
18.2.2.3. Using the AUTRPG Job Control Procedure Call Statement	18-29
<b>18.2.3. Using the EXEC and PARAM Job Control Statements</b>	18-31
<b>18.2.4. Input Deck Sequence</b>	18-40
<b>18.3. MAIN STORAGE REQUIREMENTS</b>	18-40

## PART 6. AUTO REPORT

### 19. AUTO REPORT

<b>19.1. GENERAL</b>	19-1
<b>19.2. H-*AUTO PAGE HEADINGS</b>	19-3
<b>19.2.1. Output File Identification and Control Entries (Columns 7 through 37)</b>	19-4
19.2.1.1. File Name (Columns 7 through 13)	19-4
19.2.1.2. Type (Column 15)	19-4
19.2.1.3. Space (Columns 17 and 18)	19-5
19.2.1.4. Skip (Columns 19 through 22)	19-6
19.2.1.5. Output Indicators – Records (Columns 23 through 31)	19-7
19.2.1.6. *AUTO (Columns 32 through 37)	19-9
19.2.1.7. Examples of Entries on H-*AUTO Output File Identification	19-10
<b>19.2.2. Field Description and Control Entries (Columns 32 through 70)</b>	19-11
19.2.2.1. Field Description that Prints a Title on the Page Heading (Blanks in Columns 32 through 37 and Title in Columns 45 through 70)	19-12
19.2.2.1.1. Title (Columns 45 through 70)	19-12
19.2.2.1.2. Examples of Entries on H-*AUTO Field Description (Blanks in Columns 32 through 37 and Title in Columns 45 through 70)	19-14
19.2.2.2. Field Description that Prints a Field on the Page Heading (Field Name in Columns 32 through 37)	19-16
19.2.2.2.1. Field Name (Columns 32 through 37)	19-16
19.2.2.2.2. Edit Codes (Column 38)	19-17
19.2.2.2.3. Blank After (Column 39)	19-18
19.2.2.2.4. Edit Word (Columns 45 through 70)	19-18
19.2.2.2.5. Examples of Entries on H-*AUTO Field Description (Field Name in Columns 32 through 37)	19-19
19.2.2.3. Examples of Entries on H-*AUTO Specifications	19-20
<b>19.3. D-*AUTO DETAIL REPORTS</b>	19-21
<b>19.3.1. Output File Identification and Control Entries (Columns 7 through 37)</b>	19-21
19.3.1.1. File Name (Columns 7 through 13)	19-22
19.3.1.2. Type (Column 15)	19-22
19.3.1.3. Fetch Overflow (Column 16)	19-22
19.3.1.4. Space (Columns 17 and 18)	19-23
19.3.1.5. Skip (Columns 19 through 22)	19-23
19.3.1.6. Output Indicators – Records (Columns 23 through 31)	19-24
19.3.1.7. *AUTO (Columns 32 through 37)	19-24
19.3.1.8. Examples of Entries on D-*AUTO Output File Identification	19-24

<b>19.3.2. Field Description and Control Entries (Columns 23 through 70)</b>	<b>19-25</b>
19.3.2.1. Field Description that Prints a Field and Column Heading (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-28
19.3.2.1.1. Output Indicators - Fields (Columns 23 through 31)	19-28
19.3.2.1.2. Field Name (Columns 32 through 37)	19-28
19.3.2.1.3. Edit Codes (Column 38)	19-30
19.3.2.1.4. Blank After (Column 39)	19-30
19.3.2.1.5. End Position in Output Record (Columns 40 through 43)	19-30
19.3.2.1.6. First Column Heading Line (Columns 45 through 70)	19-30
19.3.2.1.7. Examples of Entries on D-*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-33
19.3.2.2. Field Description that Prints a Heading (Blank in Column 39 and Heading in Columns 45 through 70)	19-34
19.3.2.2.1. Output Indicators - Fields (Columns 23 through 31)	19-34
19.3.2.2.2. End Position in Output Record (Columns 40 through 43)	19-34
19.3.2.2.3. Heading (Columns 45 through 70)	19-35
19.3.2.2.4. Examples of Entries on D-*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)	19-35
19.3.2.3. Field Description that Prints a Numeric Field and Column Heading and Accumulates Totals (A in Column 39)	19-36
19.3.2.3.1. Output Indicators - Fields (Columns 23 through 31)	19-37
19.3.2.3.2. Field Name (Columns 32 through 37)	19-37
19.3.2.3.3. Edit Codes (Column 38)	19-40
19.3.2.3.4. Accumulate Totals (Column 39)	19-41
19.3.2.3.5. End Position in Output Record (Columns 40 through 43)	19-42
19.3.2.3.6. First Column Heading Line (Columns 45 through 70)	19-42
19.3.2.3.7. Examples of Entries on D-*AUTO Field Description (A in Column 39)	19-44
19.3.2.4. Field Description that Prints a Second and Third Column Heading Line (C in Column 39)	19-45
19.3.2.4.1. Column Heading Continuation Lines (Column 39)	19-45
19.3.2.4.2. Second and Third Column Heading Line (Columns 45 through 70)	19-45
19.3.2.4.3. Examples of Entries on D-*AUTO Field Description (C in Column 39)	19-46
19.3.2.5. Field Description that Prints a Heading next to a Total (1 through 9 or R in Column 39)	19-47
19.3.2.5.1. Specific Total Line (Column 39)	19-47
19.3.2.5.2. Heading (Columns 45 through 70)	19-47
19.3.2.5.3. Examples of Entries on D-*AUTO Field Description (1 through 9 or R in Column 39)	19-48
19.3.2.6. Field Description that Prints a Field next to a Total (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-49
19.3.2.6.1. Field Name (Columns 32 through 37)	19-49
19.3.2.6.2. Edit Codes (Column 38)	19-50
19.3.2.6.3. Specific Total Line (Column 39)	19-50
19.3.2.6.4. Edit Word (Columns 45 through 70)	19-50
19.3.2.6.5. Examples of Entries on D-*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-50
19.3.2.7. Examples of Entries on D-*AUTO Specifications	19-52
<b>19.4. T-*AUTO TOTAL REPORTS</b>	<b>19-54</b>
<b>19.4.1. Output File Identification and Control Entries (Columns 7 through 37)</b>	<b>19-54</b>
19.4.1.1. File Name (Columns 7 through 13)	19-55
19.4.1.2. Type (Column 15)	19-55

19.4.1.3.	Fetch Overflow (Column 16)	19-55
19.4.1.4.	Space (Columns 17 and 18)	19-55
19.4.1.5.	Skip (Columns 19 through 22)	19-57
19.4.1.6.	Output Indicators - Records (Columns 23 through 31)	19-58
19.4.1.7.	*AUTO (Columns 32 through 37)	19-58
19.4.1.8.	Examples of Entries on T-*AUTO Output File Identification	19-58
<b>19.4.2.</b>	<b>Field Description and Control Entries (Columns 23 through 70)</b>	<b>19-59</b>
19.4.2.1.	Field Description that Prints a Field and Column Heading (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-62
19.4.2.1.1.	Output Indicators - Fields (Columns 23 through 31)	19-62
19.4.2.1.2.	Field Name (Columns 32 through 37)	19-62
19.4.2.1.3.	Edit Codes (Column 38)	19-63
19.4.2.1.4.	Blank After (Column 39)	19-63
19.4.2.1.5.	End Position in Output Record (Columns 40 through 43)	19-63
19.4.2.1.6.	First Column Heading Line (Columns 45 through 70)	19-63
19.4.2.1.7.	Examples of Entries on T-*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-65
19.4.2.2.	Field Description that Prints a Heading (Blank in Column 39 and Heading in Columns 45 through 70)	19-66
19.4.2.2.1.	Output Indicators - Fields (Columns 23 through 31)	19-66
19.4.2.2.2.	End Position in Output Record (Columns 40 through 43)	19-67
19.4.2.2.3.	Heading (Columns 45 through 70)	19-67
19.4.2.2.4.	Examples of Entries on T-*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)	19-67
19.4.2.3.	Field Description that Prints a Numeric Field and Column Heading and Accumulates Totals (A in Column 39)	19-68
19.4.2.3.1.	Output Indicators - Fields (Columns 23 through 31)	19-69
19.4.2.3.2.	Field Name (Columns 32 through 37)	19-69
19.4.2.3.3.	Edit Codes (Column 38)	19-70
19.4.2.3.4.	Accumulate Totals (Column 39)	19-71
19.4.2.3.5.	End Position in Output Record (Columns 40 through 43)	19-72
19.4.2.3.6.	First Column Heading Line (Columns 45 through 70)	19-72
19.4.2.3.7.	Examples of Entries on T-*AUTO Field Description (A in Column 39)	19-73
19.4.2.4.	Field Description that Prints a Second and Third Column Heading Line (C in Column 39)	19-74
19.4.2.4.1.	Column Heading Continuation Lines (Column 39)	19-75
19.4.2.4.2.	Second and Third Column Heading Line (Columns 45 through 70)	19-75
19.4.2.4.3.	Examples of Entries on T-*AUTO Field Description (C in Column 39)	19-76
19.4.2.5.	Field Description that Prints a Heading next to a Total (1 through 9 or R in Column 39)	19-76
19.4.2.5.1.	Specific Total Line (Column 39)	19-77
19.4.2.5.2.	Heading (Columns 45 through 70)	19-77
19.4.2.5.3.	Examples of Entries on T-*AUTO Field Description (1 through 9 or R in Column 39)	19-78
19.4.2.6.	Field Description that Prints a Field next to a Total (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-79
19.4.2.6.1.	Field Name (Columns 32 through 37)	19-79
19.4.2.6.2.	Edit Codes (Column 38)	19-80
19.4.2.6.3.	Specific Total Line (Column 39)	19-80
19.4.2.6.4.	Edit Word (Columns 45 through 70)	19-80
19.4.2.6.5.	Examples of Entries on T-*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-81
19.4.2.7.	Examples of Entries on T-*AUTO Specifications	19-82



---

<b>19.5. AUTO REPORT OPTIONS SPECIFICATIONS FORM</b>	19-84
<b>19.5.1. Form Entries</b>	19-84
19.5.1.1. Catalog Source Program in Library File (Column 7)	19-84
19.5.1.2. Library File and Module Name (Columns 8 through 24)	19-86
19.5.1.3. Date Suppress (Column 27)	19-86
19.5.1.4. Asterisks Suppress (Column 28)	19-87
19.5.1.5. List Options (Column 30)	19-87
<b>19.5.2. Examples of Entries on the Auto Report Options Specifications Form</b>	19-89
<b>19.6. /COPY STATEMENT</b>	19-89
<b>19.6.1. Form Entries</b>	19-90
19.6.1.1. Form Type (Column 6)	19-90
19.6.1.2. /COPY (Columns 7 through 11)	19-90
19.6.1.3. Library File and Module Name (Columns 13 through 29)	19-90
19.6.1.4. Comments (Columns 50 through 80)	19-91
<b>19.6.2. Examples of Entries on the /COPY Statement</b>	19-91
<b>19.6.3. /COPY Modifier Statements</b>	19-91
19.6.3.1. /COPY Modifier Statements for File Description Specifications	19-92
19.6.3.1.1. Form Type (Column 6)	19-92
19.6.3.1.2. File Name (Columns 7 through 13)	19-92
19.6.3.1.3. Modifying File Description Entries (Columns 15 through 80)	19-92
19.6.3.1.4. Setting File Description Entries to Blanks (Columns 15 through 80)	19-94
19.6.3.2. /COPY Modifier Statements for Input Format Specification	19-95
19.6.3.2.1. Form Type (Column 6)	19-95
19.6.3.2.2. Field Name (Columns 53 through 58)	19-95
19.6.3.2.3. Modifying Input Format Entries (Columns 43 through 70)	19-96
19.6.3.2.4. Setting Input Format Entries to Blanks (Columns 43 through 70)	19-98
<b>19.7. SUMMARY OF HOW AUTO REPORT FORMATS YOUR REPORTS</b>	19-100
<b>19.7.1. Spacing for H-*AUTO Page Heading Lines</b>	19-100
<b>19.7.2. Spacing for D-*AUTO Lines</b>	19-100
<b>19.7.3. Spacing for T-*AUTO Lines</b>	19-101
<b>19.7.4. Spacing for D-*AUTO or T-*AUTO Column Heading Lines</b>	19-102
<b>19.7.5. Placement of H-*AUTO Page Heading Lines</b>	19-102
<b>19.7.6. Placement of D-*AUTO or T-*AUTO Column Heading Lines over Fields</b>	19-103
<b>19.7.7. Placement of D-*AUTO or T-*AUTO Fields</b>	19-105
<b>19.7.8. Placement of D-*AUTO or T-*AUTO Headings or Fields next to Totals</b>	19-106
<b>19.7.9. Skipping</b>	19-106
<b>19.8. JOB CONTROL FOR AUTO REPORT</b>	19-107
<b>19.9. NON-ENGLISH LANGUAGE FEATURE</b>	19-107

**PART 7. APPENDIXES****A. RPG II DETAIL LOGIC CYCLE**

<b>A.1.</b>	<b>GENERAL</b>	A-1
<b>A.2.</b>	<b>OVERALL RPG II DETAIL LOGIC CYCLE</b>	A-1
<b>A.3.</b>	<b>RPG II DETAIL LOGIC SUBROUTINES</b>	A-6
<b>A.3.1.</b>	<b>Matching Fields Subroutine</b>	A-6
<b>A.3.2.</b>	<b>Chaining Subroutine</b>	A-8
<b>A.3.3.</b>	<b>Overflow Subroutine</b>	A-8
<b>A.3.4.</b>	<b>Look-Ahead Subroutine</b>	A-8

**B. PROGRAMMING EXAMPLES**

<b>B.1.</b>	<b>GENERAL</b>	B-1
<b>B.2.</b>	<b>CREATING A SEQUENTIAL FILE</b>	B-1
<b>B.2.1.</b>	<b>Form Entries</b>	B-3
B.2.1.1.	Control Card Specifications	B-3
B.2.1.2.	File Description Specifications	B-3
B.2.1.3.	Input Format Specifications	B-3
B.2.1.4.	Calculation Specifications	B-3
B.2.1.5.	Output Format Specifications	B-3
<b>B.2.2.</b>	<b>Report Format</b>	B-3
<b>B.3.</b>	<b>SEQUENTIAL FILE PROCESSING – MONTHLY SALES EARNINGS</b>	B-7
<b>B.3.1.</b>	<b>Form Entries</b>	B-7
B.3.1.1.	Control Card Specifications	B-7
B.3.1.2.	File Description Specifications	B-7
B.3.1.3.	Line Counter Specifications	B-7
B.3.1.4.	Input Format Specifications	B-7
B.3.1.5.	Calculation Specifications	B-8
B.3.1.6.	Output Format Specifications	B-14
<b>B.3.2.</b>	<b>Report Format</b>	B-14
<b>B.4.</b>	<b>CREATING AN INDEXED SEQUENTIAL FILE – PAYROLL MASTER FILE</b>	B-14
<b>B.4.1.</b>	<b>Form Entries</b>	B-14
<b>B.4.2.</b>	<b>Report Format</b>	B-19
<b>B.5.</b>	<b>PROCESSING AN INDEXED SEQUENTIAL FILE SEQUENTIALLY BY KEY – WAGE INCREASE</b>	B-19
<b>B.5.1.</b>	<b>Form Entries</b>	B-19
B.5.1.1.	Control Card Specifications	B-19
B.5.1.2.	File Description Specifications	B-19
B.5.1.3.	Input Format Specifications	B-25
B.5.1.4.	Calculation Specifications	B-25
B.5.1.5.	Output Format Specifications	B-25
<b>B.5.2.</b>	<b>Report Format</b>	B-25

<b>B.6.</b>	<b>PROCESSING AN INDEXED SEQUENTIAL FILE SEQUENTIALLY BETWEEN LIMITS – DEPARTMENT PAY RATES</b>	B-25
<b>B.6.1.</b>	<b>Form Entries</b>	B-28
B.6.1.1.	Control Card Specifications	B-28
B.6.1.2.	File Description Specifications	B-28
B.6.1.3.	File Extension Specifications	B-28
B.6.1.4.	Input Format Specifications	B-28
B.6.1.5.	Output Format Specifications	B-28
<b>B.6.2.</b>	<b>Report Format</b>	B-28
<b>B.7.</b>	<b>UPDATING AN INDEXED SEQUENTIAL FILE BY CHAINING – MASTER FILE CHANGES</b>	B-33
<b>B.7.1.</b>	<b>Form Entries</b>	B-33
B.7.1.1.	Control Card Specifications	B-33
B.7.1.2.	File Description Specifications	B-33
B.7.1.3.	Input Format Specifications	B-33
B.7.1.4.	Calculation Specifications	B-38
B.7.1.5.	Output Format Specifications	B-38
<b>B.7.2.</b>	<b>Report Format</b>	B-38
<b>B.8.</b>	<b>CREATING AN INDEXED SEQUENTIAL FILE – INVENTORY MASTER FILE</b>	B-38
<b>B.8.1.</b>	<b>Form Entries</b>	B-38
<b>B.8.2.</b>	<b>Report Format</b>	B-43
<b>B.9.</b>	<b>PROCESSING WITH MATCHING RECORDS – INVENTORY CONTROL</b>	B-43
<b>B.9.1.</b>	<b>Form Entries</b>	B-43
B.9.1.1.	Control Card Specifications	B-43
B.9.1.2.	File Description Specifications	B-54
B.9.1.3.	Input Format Specifications	B-54
B.9.1.4.	Calculation Specifications	B-54
B.9.1.5.	Output Format Specifications	B-55
<b>B.9.2.</b>	<b>Report Format</b>	B-55
<b>B.9.3.</b>	<b>Entering a Source Program from a Workstation</b>	B-57
<b>B.10.</b>	<b>CREATING, UPDATING, AND ADDING RECORDS TO A KEYED MIRAM FILE</b>	B-58
<b>B.10.1.</b>	<b>Creating a 5-Key MIRAM File Interactively</b>	B-58
<b>B.10.2.</b>	<b>Interactively Updating a MIRAM File Using a Console File and Automatic Screen Prompts</b>	B-61
<b>B.10.3.</b>	<b>Adding Records to a Keyed MIRAM File</b>	B-65

## C. PROGRAM TESTING AIDS

<b>C.1.</b>	<b>GENERAL</b>	C-1
<b>C.2.</b>	<b>DEBUG OPERATION</b>	C-2
<b>C.3.</b>	<b>DSPLY OPERATION</b>	C-3
<b>C.4.</b>	<b>EXCPT OPERATION</b>	C-3
<b>C.5.</b>	<b>OPERATOR CONTROL</b>	C-5
C.5.1.	Compilation Time Specification	C-5
C.5.2.	Execution Time Specification	C-5
C.5.3.	Displayed Messages	C-6

<b>C.6.</b>	<b>*ERROR FIELD</b>	C-7
<b>C.7.</b>	<b>ERROR ANALYSIS DUMP</b>	C-7
<b>C.7.1.</b>	<b>Using the Error Analysis Dump</b>	C-8
C.7.1.1.	REG SAVE AREA	C-8
C.7.1.2.	*ERROR	C-8
C.7.1.3.	RECORD	C-9
C.7.1.4.	LINKAGE VECTOR	C-9
C.7.1.5.	FLDS/CONSTS/TABLES/ARRAYS	C-10
C.7.1.6.	Error Analysis Dump Example	C-10
<b>C.8.</b>	<b>UNFORMATTED DUMP</b>	C-10
<b>C.8.1.</b>	<b>Using the Unformatted Dump</b>	C-10
C.8.1.1.	Locating Information in an Unformatted Dump	C-10
C.8.1.1.1.	Locating an Indicator	C-10
C.8.1.1.2.	Locating *ERROR Field	C-11
C.8.1.1.3.	Locating a Data Field	C-11
C.8.1.1.4.	Locating the Table Linkage Field (TLF) for a Table or Array	C-11
C.8.1.2.	Unformatted Dump Example	C-11
 <b>D. RPG II COMPILATION TIME MESSAGES</b>		
<b>D.1.</b>	<b>GENERAL</b>	D-1
<b>D.2.</b>	<b>MESSAGES</b>	D-1
 <b>E. MAIN STORAGE CONSERVATION TECHNIQUES</b>		
<b>E.1.</b>	<b>GENERAL</b>	E-1
<b>E.2.</b>	<b>DIVIDING A PROGRAM INTO SEPARATE TASKS</b>	E-1
<b>E.3.</b>	<b>USING IRAM TO PROCESS ALL DISK FILES</b>	E-1
<b>E.4.</b>	<b>CONTROL CARD SPECIFICATIONS FORM</b>	E-1
E.4.1.	Error Analysis Dump	E-1
E.4.2.	Sign Handling	E-2
<b>E.5.</b>	<b>FILE DESCRIPTION SPECIFICATIONS FORM</b>	E-2
E.5.1.	Device	E-2
<b>E.6.</b>	<b>INPUT FORMAT SPECIFICATIONS FORM</b>	E-2
E.6.1.	Record Identification Codes	E-2
E.6.2.	Identical Input Fields in Two or More Record Types	E-2
E.6.3.	Fields Not Used in Your Program	E-2
E.6.4.	Using Same Field Name for Different Fields in Different Record Types	E-2
E.6.5.	Specifying a Numeric Field as a Nonnumeric Field	E-3
<b>E.7.</b>	<b>CALCULATION SPECIFICATIONS FORM</b>	E-3
E.7.1.	Using the Same Result Field as a Common Work Field	E-3
E.7.2.	Using the GOTO Operation in Place of Conditioning Indicators	E-3
E.7.3.	Grouping Operations by Indicator	E-3
E.7.4.	Using Subroutines Rather Than Repeating Identical Operations	E-3

E.7.5.	Using Odd Length Numeric Fields	E-3
E.7.6.	Moving a Common Array Element to a Work Area	E-3
E.7.7.	Restricting the Use of Half Adjust	E-3
E.7.8.	Using Factor 1 or Factor 2 as the Result Field	E-4
E.7.9.	Using Actual Bit Patterns	E-4

## F. PROGRAM CONVERSION REQUIREMENTS

F.1.	GENERAL	F-1
------	---------	-----

F.2.	IBM SYSTEM/3 AND SYSTEM/34 RPG II SOURCE PROGRAMS	F-1
------	---	-----

F.2.1.	Compiling an IBM System/3 or System/34 RPG II Source Program	F-1
--------	--	-----

F.2.1.1.	Control Card Specifications	F-1
----------	-----------------------------	-----

F.2.1.1.1.	Core Size to Compile (Columns 7 through 9)	F-1
------------	--	-----

F.2.1.1.2.	Object Output (Column 10)	F-2
------------	---------------------------	-----

F.2.1.1.3.	Listing Option (Column 11)	F-2
------------	----------------------------	-----

F.2.1.1.4.	Core Size to Execute (Columns 12 through 14)	F-2
------------	--	-----

F.2.1.1.5.	Inquiry (Column 37)	F-2
------------	---------------------	-----

F.2.1.1.6.	Punch MFCU Zeros (Column 44)	F-2
------------	------------------------------	-----

F.2.1.2.	File Description Specifications	F-2
----------	---------------------------------	-----

F.2.1.2.1.	Device (Columns 40 through 46)	F-2
------------	--------------------------------	-----

F.2.1.2.2.	Continuation Lines (Columns 53 through 65)	F-2
------------	--	-----

F.2.1.2.3.	Number of Extents (Columns 68 and 69)	F-3
------------	---------------------------------------	-----

F.2.1.3.	File Extension Specifications	F-3
----------	-------------------------------	-----

F.2.1.4.	Line Counter Specifications	F-3
----------	-----------------------------	-----

F.2.1.5.	Input Format Specifications	F-3
----------	-----------------------------	-----

F.2.1.6.	Calculation Specifications	F-3
----------	----------------------------	-----

F.2.1.6.1.	Divide Operation (DIV)	F-3
------------	------------------------	-----

F.2.1.6.2.	Time Operation (TIME)	F-3
------------	-----------------------	-----

F.2.1.7.	Output Format Specifications	F-3
----------	------------------------------	-----

F.2.1.7.1.	*PRINT	F-4
------------	--------	-----

F.2.1.7.2.	* in Column 40	F-4
------------	----------------	-----

F.2.1.7.3.	LR Output	F-4
------------	-----------	-----

F.2.1.8.	Telecommunications Specifications	F-4
----------	-----------------------------------	-----

F.2.1.8.1.	Configuration (Column 15)	F-4
------------	---------------------------	-----

F.2.1.8.2.	Type of Control (Column 17)	F-4
------------	-----------------------------	-----

F.2.1.8.3.	Type of Code (Column 18)	F-4
------------	--------------------------	-----

F.2.1.8.4.	Dial Number (Columns 21 through 31)	F-4
------------	-------------------------------------	-----

F.2.1.8.5.	Identification - This Station (Columns 32 through 39)	F-4
------------	---	-----

F.2.1.8.6.	Identification - Remote Station (Columns 40 through 47)	F-5
------------	---	-----

F.2.1.8.7.	ITB (Column 52)	F-5
------------	-----------------	-----

F.2.1.8.8.	Polling Characters (Columns 61 and 62)	F-5
------------	--	-----

F.2.1.8.9.	Addressing Characters (Columns 63 and 64)	F-5
------------	---	-----

F.2.1.8.10.	Interspersed Mode	F-5
-------------	-------------------	-----

F.2.2.	Executing an IBM System/3 RPG II Program	F-5
--------	--	-----

F.3.	SPERRY UNIVAC 9200/9300 SYSTEM RPG SOURCE PROGRAMS	F-5
------	--	-----

F.3.1.	Compiling a SPERRY UNIVAC 9200/9300 RPG Source Program	F-6
--------	--	-----

F.3.2.	Control Card Specifications	F-6
--------	-----------------------------	-----

F.3.2.1.	Listing Option (Column 11)	F-6
----------	----------------------------	-----

F.3.2.2.	Rewind (Column 12)	F-6
----------	--------------------	-----

F.3.2.3.	Program Identification (Columns 30 through 33 or 46 through 49)	F-6
----------	---	-----

<b>F.3.3. File Description Specifications</b>	F-6
F.3.3.1. File Name (Columns 7 through 13)	F-6
F.3.3.2. Logical Unit Number (Columns 47 through 52)	F-6
F.3.3.3. Alternate Logical Unit Numbers (Columns 54 through 56)	F-6
<b>F.3.4. File Extension Specifications</b>	F-7
F.3.4.1. Compilation Time Table Indicator (Column 6)	F-7
<b>F.3.5. Input Format Specifications</b>	F-7
F.3.5.1. Matching Fields	F-7
F.3.5.2. Invalid Numeric Data	F-7
<b>F.3.6. Calculation Specifications</b>	F-7
F.3.6.1. UPSI Special Name	F-7
<b>F.3.7. Output Format Specifications</b>	F-7

## G. AUTO REPORT ERROR MESSAGES

<b>G.1. GENERAL</b>	G-1
<b>G.2. COMPILE-TIME MESSAGES</b>	G-1

## INDEX

## FIGURES

→ 1-1. Using RPG II via Workstation	1-23
1-2. Using RPG II via Cards	1-24
2-1. Indexed Sequential File Structure	2-4
2-2. Direct File Structure	2-6
2-3. Fixed-Length File Formats	2-8
2-4. Variable-Length File Formats	2-10
2-5. Alphanumeric Data Format	2-11
2-6. Binary Data Format	2-11
2-7. Packed Numeric Data Format	2-12
2-8. Unpacked Numeric Data Format	2-12
3-1. Line Insertion Example	3-3
4-1. Control Card Specifications Form	4-2
4-2. Examples of Entries on Control Card Specifications Form	4-8
5-1. File Description Specifications Form	5-2
5-2. Examples of Entries on File Description Specifications Form	5-31
6-1. Input Format Specifications Form	6-2
6-2. Spread Card Example	6-6
6-3. Specifying Spread Cards	6-8
6-4. Providing for Undefined Record Types	6-9
6-5. Example of Entries on Input Format Specifications Form	6-21

7-1.	Calculation Specifications Form	7-2
7-2.	Using * Calculation Lines	7-4
7-3.	Using AN/OR Lines	7-5
7-4.	Examples of Entries on Calculation Specifications Form	7-45
8-1.	Output Format Specifications Form	8-2
8-2.	Examples of Entries on Output Format Specifications Form	8-12
9-1.	File Extension Specifications Form	9-2
9-2.	Examples of Entries on File Extension Specifications Form	9-9
10-1.	Line Counter Specifications Form	10-2
10-2.	Examples of Entries on Line Counter Specifications Form	10-4
11-1.	Telecommunications Specifications Form	11-2
11-2.	Examples of Entries on Telecommunications Specifications Form	11-7
12-1.	Examples of Specifying Record Identifying Indicators	12-2
12-2.	Unwanted Control Break Example - Input Records	12-3
12-3.	Unwanted Control Break Example - Entries to Avoid Incorrect Output	12-4
12-4.	Unwanted Control Break Example - Outputs	12-6
12-5.	Using Fetch Overflow	12-10
12-6.	Specifying Calculation Conditioning	12-17
12-7.	Specifying Output Conditioning for Records and Fields	12-19
12-8.	Allocating a PF Key with a Positive Number	12-23
12-9.	Testing PF Key Activation Status	12-24
12-10.	Allocating a PF Key with a Negative Number and Testing Its Activation Status	12-25
12-11.	Allocating the Unsolicited Inquiry Request Subroutine	12-26
12-12.	Testing the Unsolicited Inquiry Request Status	12-26
13-1.	Example of Specifying Matching Fields	13-3
13-2.	Matching Record Selection	13-4
13-3.	Example of Specifying Chaining Using CHAIN Operation	13-9
13-4.	Example of Specifying Chaining Using C1 through C9	13-10
13-5.	Example of Chaining a Direct File	13-14
13-6.	Creating a Direct File	13-15
13-7.	Example of Processing an Indexed Sequential File Randomly, Using a Record Address File	13-18
13-8.	Example of Processing an Indexed Sequential File between Limits	13-19
13-9.	Example of Processing with an ADDRROUT File	13-21
13-10.	Example of Processing a Demand File with the READ Operation	13-23
13-11.	Example of Processing a Demand File with the SETLL Operation	13-26
13-12.	Example of Controlling Record Selection during Multifile Processing	13-30
13-13.	Example of Required Entries for Matching Records from a Secondary File	13-32
13-14.	Table File Formats	13-34
13-15.	Table Definition	13-39
13-16.	LOKUP Operation Examples	13-41
13-17.	Computing a Payroll with Tables	13-44
13-18.	Modifying Table Data during Program Execution	13-46
13-19.	Adding Elements to Existing Tables	13-47
13-20.	Array File Formats	13-48
13-21.	Array Definition	13-53
13-22.	Loading an Array with the Input Format Specifications Form Using Fixed Indexes	13-56
13-23.	Loading an Array with the Input Format Specifications Form Using Input Fields as Indexes	13-57

13-24. Loading an Array with the Calculation Specifications Form	13-59
13-25. Using the LOKUP Operation with Arrays	13-62
13-26. Using Arrays to Format Output Records	13-65
13-27. Calculating Totals without Arrays	13-67
13-28. Calculating Totals with Arrays	13-68
13-29. Using EXCPT Operation to Write Array Elements	13-70
13-30. Causing Characters to Be Considered Equal	13-76
13-31. Inserting a Character between Two Existing Characters in the Collating Sequence	13-77
13-32. Changing the Position of Characters in the Collating Sequence	13-78
13-33. Using File Translation to Translate Characters in a File	13-82
13-34. Using File Translation to Translate Characters for More than One but Not All Files	13-82
13-35. Screen Format ENTRNAME for Sample Workstation Program	13-87
13-36. Screen Format ADDRFRMT for Sample Workstation Program	13-87
13-37. Screen Format NOFIND for Sample Workstation Program	13-87
13-38. Sample Workstation Program	13-88
↓ 13-38A. Indexed MIRAM File Program Example	13-90b
13-39. Function Key 5 Prompt Screen	13-94
13-40. One-Column Prompt Screen (23 Fields)	13-94b
13-41. Two-Column Prompt Screen (24 Fields)	13-94b
13-42. Two-Column Prompt Screen, Self-Adjusting Fields	13-94c
13-43. Three-Column Prompt Screen (47 Fields)	13-94c
↑ 13-44. Four-Column Prompt Screen (70 Fields)	13-94d
14-1. Printer Format Chart	14-2
14-2. Example of Entries on Printer Format Chart	14-4
14-3. Edit Word Format	14-9
14-4. Examples of Edit Words and Results Printed	14-11
14-5. Example of Using *PLACE	14-13
14-6. Example of Using *PLACE to Repeat Constants	14-13
14-7. Specifying a Line Counter File	14-14
14-8. Specifying Lines Associated with Channel Numbers for a Line Counter File	14-14
15-1. Example of Specifying an Internal Subroutine	15-3
15-2. Example of Specifying an External Subroutine	15-5
17-1. Specifying and Describing Communications Files	17-3
17-2. Receive-Only Processing Using a DCT 2000 Data Communications Terminal	17-5
17-3. Transmit-Only Processing Using Two DCT 1000 Data Communications Terminals	17-7
17-4. Transmit a File, Then Receive Another File Using a BSC Remote Terminal	17-9
17-5. Formatting a Screen and Entering Data	17-13
17-6. Transmit with Reception of Conversational Reply Using Four UNISCOPE 100 Display Terminals	17-14
17-7. Formatting a Printer and Entering Data	17-18
17-8. Transmit with Reception of Conversational Reply Using Two DCT 500 Data Communications Terminals and a Teletypewriter Terminal	17-19
17-9. ICAM Load Module Generation	17-22
18-1. Using the RPG Job Control Procedure Call Statement with Standard Default Options	18-8a
18-2. Job Control Statements Generated by the RPG Procedure Call Statement with Standard Default Options	18-9
18-3. Using the RPG Job Control Procedure Call Statement with Nonstandard Options	18-10
18-4. Job Control Statements Generated by RPG Procedure Call Statement with Nonstandard Options	18-11



18-5.	Using the RPGL Job Control Procedure Call Statement	18-12
18-6.	Job Control Statements Generated by RPGL Procedure Call Statement	18-13
18-7.	Using the RPGLG Job Control Procedure Call Statement	18-14
18-8.	Job Control Statements Generated by RPGLG Procedure Call Statement	18-16
18-9.	Using the AUTO Job Control Procedure Call Statement with Standard Default Options	18-25
18-10.	Job Control Statements Generated by the AUTO Procedure Call Statement with Standard Default Options	18-26
18-11.	Using the AUTO Job Control Procedure Call Statement with Nonstandard Options	18-27
18-12.	Job Control Statements Generated by the AUTO Procedure Call Statement with Nonstandard Options	18-28
18-13.	Using the AURPG Job Control Procedure Call Statement	18-29
18-14.	Job Control Statements Generated by the AURPG Procedure Call Statement	18-30
18-15.	Using the EXEC and PARAM Job Control Statements	18-35
18-16.	Using the EXEC and PARAM Job Control Statements with AUTO Report (with Input on Cards)	18-37
18-17.	Using the EXEC and PARAM Job Control Statements with AUTO Report (with Input on Diskette)	18-39
18-18.	Deck Arrangement for Compiling, Link Editing, and Execution	18-41
19-1.	Auto Report Function	19-3
19-2.	H-*AUTO Output File Identification	19-4
19-3.	Examples of Entries on H-*AUTO Output File Identification	19-10
19-4.	H-*AUTO Field Descriptions	19-11
19-5.	H-*AUTO Field Description (Blanks in Columns 32 through 37 and Title in Columns 45 through 70)	19-12
19-6.	Examples of Entries on H-*AUTO Field Description (Blanks in Columns 32 through 37 and Title in Columns 45 through 70)	19-14
19-7.	H-*AUTO Field Description (Field Name in Columns 32 through 37)	19-16
19-8.	Examples of Entries on H-*AUTO Field Description (Field Name in Columns 32 through 37)	19-19
19-9.	Examples of Entries on H-*AUTO Specifications	19-20
19-10.	D-*AUTO Output File Identification	19-21
19-11.	Examples of Entries on D-*AUTO Output File Identification	19-24
19-12.	D-*AUTO Field Descriptions	19-26
19-13.	D-*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-28
19-14.	Examples of Entries on D-*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-33
19-15.	D-*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)	19-34
19-16.	Examples of Entries on D-*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)	19-35
19-17.	D-*AUTO Field Description (A in Column 39)	19-36
19-18.	Examples of Entries on D-*AUTO Field Description (A in Column 39)	19-44
19-19.	D-*AUTO Field Description (C in Column 39)	19-45
19-20.	Examples of Entries on D-*AUTO Field Description (C in Column 39)	19-46
19-21.	D-*AUTO Field Description (1 through 9 or R in Column 39)	19-47
19-22.	Examples of Entries on D-*AUTO Field Description (1 through 9 or R in Column 39)	19-48
19-23.	D-*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-49
19-24.	Examples of Entries on D-*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-51
19-25.	Examples of Entries on D-*AUTO Specifications	19-52

19-26.	T-*AUTO Output File Identification	19-54
19-27.	Examples of Entries on T-*AUTO Output File Identification	19-59
19-28.	T-*AUTO Field Descriptions	19-60
19-29.	T-*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-62
19-30.	Examples of Entries on T-*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)	19-66
19-31.	T-*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)	19-66
19-32.	Examples of Entries on T-*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)	19-68
19-33.	T-*AUTO Field Description (A in Column 39)	19-68
19-34.	Examples of Entries on T-*AUTO Field Description (A in Column 39)	19-74
19-35.	T-*AUTO Field Description (C in Column 39)	19-75
19-36.	Examples of Entries on T-*AUTO Field Description (C in Column 39)	19-76
19-37.	T-*AUTO Field Description (1 through 9 or R in Column 39)	19-77
19-38.	Examples of Entries on T-*AUTO Field Description (1 through 9 or R in Column 39)	19-78
19-39.	T-*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-79
19-40.	Examples of Entries on T-*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)	19-81
19-41.	Examples of Entries on T-*AUTO Specifications	19-82
19-42.	Auto Report Options Specifications Form	19-85
19-43.	Examples of Entries on Auto Report Options Specifications Form	19-89
19-44.	Examples of Entries on /COPY Statement	19-91
19-45.	Modifying File Description Entries	19-93
19-46.	Setting File Description Entries to Blanks	19-94
19-47.	Modifying Input Field Entries	19-97
19-48.	Setting Input Field Entries to Blanks	19-98
19-49.	Overflow of D-*AUTO or T-*AUTO Lines	19-105
A-1.	Overall RPG II Detail Logic Cycle	A-2
A-2.	RPG II Detail Logic Subroutines	A-7
B-1.	Creating a Sequential File	B-2
B-2.	Entries for Creating a Sequential File	B-4
B-3.	Staff File Listing Format	B-6
B-4.	Sequential File Processing - Monthly Sales Earnings	B-8
B-5.	Entries for Sequential File Processing - Monthly Sales Earnings	B-9
B-6.	Monthly Sales Report Format	B-15
B-7.	Creating an Indexed Sequential File - Payroll Master File	B-16
B-8.	Entries for Creating an Indexed Sequential File - Payroll Master File	B-17
B-9.	Master File Listing Format	B-20
B-10.	Processing an Indexed Sequential File by Key - Wage Increase	B-21
B-11.	Entries for Processing an Indexed Sequential File by Key - Wage Increase	B-22
B-12.	Hourly Pay Rate Increase Report Format	B-26
B-13.	Processing an Indexed Sequential File between Limits - Department Pay Rates	B-27
B-14.	Entries for Processing an Indexed Sequential File Sequentially between Limits - Department Pay Rates	B-29
B-15.	Department Pay Rates Report Format	B-32
B-16.	Updating an Indexed Sequential File by Chaining - Master File Changes	B-34
B-17.	Entries for Updating an Indexed Sequential File by Chaining - Master File Changes	B-35

B-18.	Master File Changes Report Format	B-39
B-19.	Creating an Indexed Sequential File - Inventory Master File	B-40
B-20.	Entries for Creating an Indexed Sequential File - Inventory Master File	B-41
B-21.	Master Record Listing Format	B-44
B-22.	Processing with Matching Records - Inventory Control	B-45
B-23.	Entries for Processing with Matching Records - Inventory Control	B-47
B-24.	Daily Activity Report Format	B-56
B-25.	Creating a 5-Key MIRAM File Interactively	B-58
B-26.	Interactively Updating a MIRAM File Using a Console File and Automatic Screen Prompts	B-61
B-27.	Adding Records to a Keyed MIRAM File	B-65
C-1.	Using the DEBUG Operation	C-4
C-2.	Using *ERROR Field to Allow Processing to Continue when Error Occurs	C-7
C-3.	Error Analysis Dump	C-12
C-4.	Unformatted Dump	C-25

## TABLES

2-1.	Allowable File Formats	2-7
3-1.	RPG II Sequence Checking during Compilation	3-4
5-1.	File Use	5-4
5-2.	Summary of File Format, Block Length, and Record Length Entries	5-12
5-3.	Summary of File Processing Mode, Record Address Type, and File Organization Entries	5-15
5-4.	Summary of Interaction between the Overflow and Output Indicators	5-17
5-5.	Summary of Device Field Entries	5-18
5-6.	Summary of Continuation Line Entries	5-21
5-7.	Summary of File Sharing Environments	5-25
5-8.	Summary of File Addition Entries	5-28
5-9.	Summary of Tape Rewind Options	5-29
6-1.	Summary of Sequence, Number, and Optional Field Entries	6-4
6-2.	Interpretation of &, -, and Blank Characters	6-10
7-1.	Summary of Operations	7-39
8-1.	Skip Channel Number Translation	8-6
9-1.	Summary of From File Name and To File Name Entries	9-4
11-1.	Summary of Remote Terminal and Remote Device Entries	11-6
12-1.	Indicator Definition	12-20
12-2.	Indicator Use	12-21
12-3.	Setting Indicators On and Off	12-22
13-1.	File Organization, Record Retrieval, and Processing Methods	13-1
13-2.	Required File Description Entries for Table Definition	13-37
13-3.	Required File Extension Entries for Table Definition	13-38

---

13-4.	Required File Description Entries for Array Definition	13-51
13-5.	Required File Extension Entries for Array Definition	13-52
13-6.	RPG II Collating Sequence	13-73
14-1.	Combined Edit Codes	14-6
14-2.	Using Combined Edit Codes	14-7
16-1.	RPG II Language Features with Restricted Use	16-2
16-2.	Summary of Required Entries – File Description Specifications Form	16-5
16-3.	Summary of File Organization, Access Methods, and File Types	16-6
17-1.	Summary of Record Position Ranges and Screen Line Numbers for Entry or Display	17-12
19-1.	Message Text Format	19-108

**PART 1. RPG II CONCEPTS AND DATA  
HANDLING CAPABILITIES**

1944

# 1. Concepts

## 1.1. WHAT IS RPG II?

RPG II is a programming language designed to improve the efficiency of a business operation. RPG II helps provide up-to-date, accurate information on such vital operations as payroll, accounts receivable, inventory control, billing, sales, and marketing analyses. Unlike many programming languages, RPG II has a fixed logic, which means it is easy to use. You need not labor over the program, writing lengthy, detailed program statements.

The RPG II compiler produces a printed listing of input, a listing of error diagnostics, and a map of main storage. These listings promote quick discovery and correction of coding or keypunching errors. As you can see, using RPG II can solve your business data processing problems with minimum effort.



## 1.2. HOW TO USE RPG II ON A WORKSTATION

The workstation is the principal means by which you and the OS/3 system communicate. With the workstation, you can perform almost every function available with OS/3. You can create a program, correct any errors in it, build a job control stream to run it, and then execute it.

This section describes how to:

- log on (connect to) the system;
- activate the general editor (EDT);
- activate the RPG II editor to create or update an RPG II program;
- store the program in a file;
- use the general editor to build a job control stream;
- activate the job control dialog to build a job control stream;
- activate the error file processor to correct the program after compilation;
- activate screen mode processing to correct the program;





- store data in a file;
- execute the program; and
- log off (disconnect from) the system.

To turn on the workstation, press the POWER ON/OFF switch on the right side of the front panel below the screen. Allow the workstation a few seconds to warm up.

Your system must be active before you can use the workstation. For information about this procedure, see the current version of the handbook for operators, UP-8859.

The following screen appears when the system is active and the workstation is sitting idle:

```

000000      SSSS      /      333
000000000  SSSSSS    //     33333
00  00  SS  SS      ///    33  33
00  00  SS          ///     33
00  00  SS          ///     33
00  00  SS          ///     333
00  00  SS          ///     33
00  00  SS  SS     ///    33  33
000000000  SSSSSS    //     3333333
000000      SSSS      /      3333

```

SPERRY UNIVAC INTERACTIVE OPERATING SYSTEM  
DEPRESS TRANSMIT FOR LOGON

This screen tells you to log on. You must log on every time you use the workstation. Logging on identifies you as a legitimate user, connects you to the system, and keeps an accounting of the time you use.

After pressing the XMIT key to log on, this screen appears:

```

OS/3 INTERACTIVE SERVICES
LOGON IDENTIFICATION:  USER-ID          >MAB<
                      ACCOUNT NUMBER   >.....<
                      PASSWORD         >.....<
OPTIONS:              EXECUTION PROFILE >.....<
                      BULLETIN        >YES<
                      LOG              >YES<

```





Only the user-id is required, but your system may also need the other information. Check with your system administrator. Enter the applicable information in the underline areas and press the XMIT key. (You must press the XMIT key every time you want to send information to the system.)

The system returns a message accepting the logon:

```
OS/3 INTERACTIVE SERVICES
LOGON ACCEPTED AT 12:45:39 on 82/01/28. REV. X.X
```

Next, the system bulletin appears. It provides information about the system and the procedure for entering a command:

```
IS27 TODAYS BULLETIN IS:
-- TO TYPE IN COMMANDS.  DEPRESS 'FUNCTION' AND --
-- 'SYSTEM MODE' KEYS SIMULTANEOUSLY. THEN TYPE --
-- THE COMMAND AND DEPRESS TRANSMIT.           --
-- ON UNISCOPE'S DEPRESS 'MESSAGE WAITING' KEY. --
```

Press the FUNCTION and SYSTEM MODE keys. The cursor blinks in the upper-left corner of the screen and you can enter a command.

Enter

```
EDT
```

to activate the general editor (EDT).

The general editor allows you to create, edit, and update programs as well as edit data and library files. It also provides an RPG II subeditor that creates and updates RPG II programs. For more information about the general editor, see the general editor (EDT) user guide/programmer reference, UP-8828 (current version).

Enter @RPG alongside the EDT command to activate the RPG II editor:

```
EDT @RPG
```

The system returns a message indicating the general editor is activated:

```
ED0000 EDITOR VERSION XX.X READY
1.0000 >
```

If you are already in an EDT session, just enter @RPG to activate the RPG II editor. Since the RPG II editor is a subeditor of the general editor, you can only use it while the general editor is activated.

The first screen displayed by the RPG II editor is:

```
          RPGEDT VERSION #
SELECT MODE ( )
C = CREATE   U = UPDATE
SELECT FORMAT TYPE ( )
  1 = POSITIONAL   2 = FORMATTED   3 = FREEFORM
SPECIFICATION TYPE DISPLAY? ( )   Y = YES   N = NO
```

Use this screen to specify:

- Whether the RPG II editor will create or update an RPG II program.
- Whether the display formats for the RPG II specifications forms will be formatted, positional, or free form. The format types are geared to how well you know RPG II.
- Whether the list of specification types will be displayed next so that you can select the RPG II specification forms needed.

Enter the requested information. If you requested the list of specification types, the following screen appears:

```
ENTER SPECIFICATION TYPE: (--)
          **** SPECIFICATION TYPES ****
H - HEADER           IF - INPUT FIELD           A - ALTSEQ
F - FILE             C - CALCULATION           FT - *FILES
E - EXTENSION        OR - OUTPUT FILE         EQ - *EQUATE
L - LINE COUNTER     OF - OUTPUT FIELD         FF - FREE FORM/
IR - INPUT RECORD    T - TELECOMMUNICATION           COMMENTS
AU - AUTO REPORT     AC - AUTO REPORT (/COPY)  ** - TABLE/ARRAY
          OPTIONS                               DELIMITER
```

Enter the code for first specification form you want to use. The form is displayed in the format type you chose:

■ Formatted display

The formatted display provides the most prompting and is the easiest to use. Each field is identified by its name and starting column number and the length of each field is depicted by underlines. Enter data in the underlines and tab to the next field you want. The following example shows sample entries on a control specifications form with a formatted display:

```

LINE - 1.0000
1 SEQUENCE NUMBER: B0010 6 FORM TYPE H
7 COMPILATION MODE: _ 8 ERROR DUMP: D 9. OPERATOR CONTROL: _
15 DEBUG: _ 18 CURRENCY SYMBOL: _
21 INVERTED PRINT: _ 26 ALTSEQ: _
31 BINARY SEARCH: _ 40 SIGN HANDLING: S
41 FORMS ALIGNMENT: I 42 INDICATOR INIT.: _
43 FILE TRANSLATION: _ 70 CCA NAME: _____
74 SUBROUTINE: _ 75 PROGRAM ID: CRSEQ
NEXT SPECIFICATION TYPE, ST, OR CMD: (F_)
-----
-----

```

■ Positional display

The positional display is for experienced programmers. It offers less prompting and requires a more complete knowledge of RPG II. Each field is identified by a starting column number and the length of each field is indicated by underlines. You enter data in the underlines and tab to the next field you want. The following screen shows sample entries on a control specifications form with a positional display:

```

LINE - 1.0000
1 1 2 2 3 4 4 4 4 7 7 7
1 6 7 8 9 5 8 1 6 1 0 1 2 3 0 4 5
B0010 H D I _____ S I _____ CRSEQ
NEXT SPECIFICATION TYPE, ST, OR CMD: (F_)
-----
-----

```

▼ ■ Free-form display

The free-form display offers no prompting and is for highly experienced programmers who wish to create programs quickly. Column numbers are the only prompts provided. You fill in the characters exactly where you want and space to the following column positions. The following example shows sample entries on a free-form display:

```

LINE - 1.0000
      1      2      3      4      5      6
123456789012345678901234567890123456789012345678901234
 000010H D      I      SI
-----
      1      1      1
6 7      8      9      0      1      2
567890123456789012345678901234567890123456789012345678
  CRSEQ
-----
NEXT SPECIFICATION TYPE, ST, OR CMD: (F_)
-----
-----

```

After completing a specifications form, press the XMIT key. If you made any syntactical errors, an error message appears on the last two lines of the screen. The error message contains an error number, description, and recovery action. A blinking character depicts the location of the error. Correct the error by repositioning the cursor to the error, entering the correct entry, tabbing to the end of the form to enter the next type of screen you want, and then pressing the XMIT key.

The source statements are entered into an EDT temporary workspace one at a time where they are assigned temporary line numbers. You use these line numbers to manipulate statements in your program during an EDT session.

When your program is complete, you can continue to use the editor to correct or update it. When the program is exactly as you want it, use the @WRITE command to store the program on a SAT disk or diskette file since the EDT workspace is a temporary file:

```
@WRITE MO=SFIL,FIL=FILE1,VSN=PUBDSK,SIZE=2,SAT=Y
```

To terminate the RPG II editor, enter

```
@RPG END
```

This command terminates the RPG II editor but not the general editor. For more information about the RPG II editor, see the current version of the RPG II editor programmer reference, UP-8803.

▲

We will now use the general editor to build a job control stream to compile the RPG II program we just built with the RPG II editor. First, use an @DELETE command to delete all lines in the EDT workspace so that you can start coding the job control stream without carrying over any statements from the RPG II editor session.

You can enter either individual job control statements or the RPG II jprocs (job control procedure call statements). The jprocs save considerable time because you don't need to write individual job control statements; the jprocs generate them for you.

Include the ERRFIL parameter in the job control stream if you want to use the error file processor to correct compilation errors.

This job control stream uses a jproc to compile the program and create an error file:

```
// JOB JLLFIL
// SFIL RPG IN=(PUBDSK,FIL1),ERRFIL=(PUBDSK,PROG1,MERRFIL)
/ &
```

**NOTE:**

*An alternate way to write a job control stream is to activate the job control dialog, which is a question-and-answer session that prompts you to enter either individual job control statements or jprocs. To use this dialog, enter @HALT to terminate the general editor, and then enter:*

```
RV JC$BLD
```

For more information about interactive job control, see the OS/3 job control user guide, UP-8065 (current version).

Store the job control stream with the @WRITE command:

```
@WRITE MO=JCLFIL,FIL=JCL2,VSN=PUBDSK,SIZE=2,SAT=Y
```

To terminate the general editor, enter

```
@HALT
```

You must allocate the error file (that you specified in the jproc) as a MIRAM file

```
AL MI,FIL=PROG1,VSN=PUBDSK,SIZE=2
```

before you compile the program.

Now, enter

```
RV JCLFIL:(FIL2,PUBDSK)
```

to compile the program.

↓  
Immediately after compilation, if there are any compilation errors, you can correct them using the general editor or you can also activate the error file processor:

```
EDT @EFP
```

The error file processor is a subeditor of the general editor, so you can only use it while the general editor is activated. If you are in an EDT session, just enter @EFP.

Remember that you can only use the error file processor if you specified the ERRFIL parameter in the job control stream. The advantage of the error file processor is that you don't need to wait for the printed compilation listing to correct the program.

**NOTE:**

*If you are in an EDT session when you activate the error file processor, first use an @DELETE command to delete all lines in the EDT workspace so that no statements carry over from the previous session. Use an @ command to set the line number and increment back to 1 if you used an @ command earlier in the session to change line numbers.*

Once you activate the error file processor, you see the following screen:



```
EFP001 VERSION n.n  
EFP002 ENTER ERROR-FILE MODULE-NAME,FILE-NAME,VSN  
▷ MERRFIL,PROG1,PUBDSK
```

Use this screen to specify:

- the error file module name you specified on the ERRFIL parameter;
- the error file name (lblname) you specified on the ERRFIL parameter; or
- the error file volume serial number you specified on the ERRFIL parameter.

Enter the requested information. The error file processor reads the error file, locates the program to which it applies, reads a copy of the program into the EDT workspace, and displays the number of errors.

Using error file processor commands (such as @EFP), you direct the error file processor to display the compilation errors along with the program statements that contain the errors. The following example shows that the first error applies to line 5 of the program. The EDT command corrects the error.

↑



```

ERR-0001 WARNING: FIELD NAME IS UNREFERENCED.
INRED NOTE 205
0005.0000 02020I 1 80 INRED
15.0000>@on 5 change 'inred' to 'inrec'

```

After correcting the errors, we store the corrected program back to its original file (overwriting it):

```
@WRITE MO=SFIL,FIL=FIL1,VSN=PUBDSK
```

To terminate this error file processor, enter

```
@EFP END
```

The @EFP END command terminates the error file processor but not the general editor. For more information about the error file processor, see the general editor (EDT) user guide/programmer reference, UP-8828 (current version).

**NOTE:**

*An alternate way to make minor corrections to an existing program is to use the screen mode processing capability provided by the general editor. An RPG II template screen is displayed for your input, but unlike the RPG II editor, there is no syntax checking. It is particularly helpful, however, because the column numbers are displayed which makes it easy to correct a line.*

Enter

```
@SET,MODE=SCREEN,LANGUAGE=RPG,SCRDSKY=FOLD,RECENTRY=SINGLE
```

to use screen mode processing:

The following screen appears and you correct the line in error:

```

OS/3 EDT (V7.52)          EDT   RPG   SINGLE   FOLD
*****
...+...1...+...2...+...3...+...4...+...5...+...6...+...7

PG-LN FORM CONTROL-FIELDS/PROGRAM-ID
LINE #  ...+ .  ...1...+...2...+...3...+...4...+...5...+...6
15.0000 02020 I  ----- I 80 INREC -----
.+...7...+...8
-----

EDT COMMAND: @MOVE 15 TO 5
*****
ERROR MESSAGE AREA (2 lines)

```



↓  
In this example, the EDT command replaces the line that was in error with the line you just created using screen mode processing.

Enter

```
@SET M=L
```

to terminate screen mode processing. For more information about screen mode processing, see the general editor (EDT) user guide/programmer reference, UP-8828.

We will continue to use the general editor to create a data file that will provide input to the RPG II program. First, use an @DELETE command to delete all lines in the EDT workspace so that no statements carry over from the previous session. Then, enter the data your program requires.

Use the @WRITE command to store the data in a MIRAM file:

```
@WRITE FIL=MFIL,VSN=PUBDSK,SIZE=2,RCSZ=80
```

We will now use the general editor to build a job control stream to execute the RPG II program. First, use an @DELETE command to delete all lines in the EDT workspace so that no statements carry over from the previous session.

The following job control stream uses a jproc to execute the program:

```
// JOB BAB  
// DVC 20  
// LFD LIST  
// DVC 50  
// VOL PUBDSK  
// LBL MFIL  
// LFD INDATA  
//SFIL RPGLG IN=(PUBDSK,FIL1)  
/ &
```

Use the @WRITE command to store the job control stream:

```
@WRITE MO=BAB,FIL=JL3,VSN=PUBDSK,SIZE=2,SAT=Y
```

Enter

```
@HALT
```

to terminate the general editor.

Enter

```
RV BAB:(JL3,PUBDSK)
```

to execute the program.

↑



If there are any execution errors, use the general editor, the RPG II editor, or screen mode processing to correct them. ↓

After the program successfully executes, enter

LOGOFF

to log off the system.

The following key-in session creates an RPG II program, builds job control streams, corrects the program, and then compiles, links, and executes the program. An explanation of this coding example follows the coding.

```
1 LOGON MAB
2 EDT @RPG
```

**NOTE:**

*The coding is continued following the examples of screen entries.*

**INITIAL DISPLAY**

```
                RPGEDT VERSION XX.XX/XX
SELECT MODE (C)
C = CREATE    U = UPDATE
SELECT FORMAT TYPE (2)
1 = POSITIONAL  2 = FORMATTED  3 = FREEFORM
SPECIFICATION TYPE DISPLAY? (N)  Y = YES  N = NO
```

In this example, we select C, 2, and N; then we transmit these selections to the PRG II editor by pressing the XMIT key. A control (header) specification screen in formatted format is automatically displayed. ↑

↓

## HEADER SPECIFICATION

LINE - 1.0000

1 SEQUENCE NUMBER: 00010      6 FORM TYPE H

7 COMPILATION MODE: \_      8 ERROR DUMP: \_      9. OPERATOR CONTROL: \_

15 DEBUG: \_      18 CURRENCY SYMBOL: \_

21 INVERTED PRINT: \_      26 ALTSEQ: \_

31 BINARY SEARCH: \_      40 SIGN HANDLING: \_

41 FORMS ALIGNMENT: \_      42 INDICATOR INIT.: \_

43 FILE TRANSLATION: \_      70 CCA NAME: \_\_\_\_

74 SUBROUTINE: \_      75 PROGRAM ID: CRSEQ\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (F\_)

-----

-----

↑

Here, we key in our source statement on the appropriate underlines. On the next screen identification line, we specify what we want displayed next. Because we keyed in F, after we transmit the screen, a file specification (in formatted format) is displayed.

# FILE SPECIFICATION



```

LINE - 2.0000
1 SEQUENCE NUMBER: 01010 6 FORM TYPE F 7 FILENAME: INDATA__ 15 FILE TYPE: I
16 FILE;DESIGNATION: P 17 EOF: E 18 SEQ: A 19 FILE FORMAT: F 20 BLK LEN: 80__
24 RECORD LEN: 80__ 28 FILE PROCESSING MODE: _ 29 KEY OR FIELD LENGTH: __
31 RECORD ADDRESS TYPE: _ 32 FILE ORGANIZATION: _ 33 OVERFLOW INDICATOR: __
35 KEY FLD STARTING LOC: ____ 39 EXTENSION/LINE CTR CODE: _ 40 DEVICE: DISK __
53 CONT LINES: 5 54 OPTION: _____ 60 ENTRY/STORAGE/KEY LOC: _____
66 FILE ADDITION AND CYL OVF % /KEY-LENGTH: __ 68 # OF EXTENTS/KEY OPTIONS: __
70 REWIND: _ 71 FILE CONDITIONERS: __

NEXT SPECIFICATION TYPE, ST, OR CMD: (___)

-----
-----

```

Again, we key in our source statement, specify the next screen display, and transmit. Because we transmitted the screen without keying in an entry on the next screen identification line, another file specification screen in formatted format is displayed.



↓  
**FILE SPECIFICATION**

```

LINE - 3.0000
1 SEQUENCE NUMBER: 01020 6 FORM TYPE F 7 FILENAME: LIST ___ 15 FILE TYPE: 0
16 FILE DESIGNATION: _ 17 EOF: _ 18 SEQ: _ 19 FILE FORMAT: F 20 BLK LEN: 132
24 RECORD LEN: 132 28 FILE PROCESSING MODE: _ 29 KEY OR FIELD LENGTH: __
31 RECORD ADDRESS TYPE: _ 32 FILE ORGANIZATION: _ 33 OVERFLOW INDICATOR: 0F
35 KEY FLD STARTING LOC: ____ 39 EXTENSION/LINE CTR CODE: _ 40 DEVICE: PRINTER
53 CONT LINES: _ 54 OPTION: _____ 60 ENTRY/STORAGE/KEY LOC: _____
66 FILE ADDITION AND:CYL OVF %/KEY-LENGTH: __ 68 # OF EXTENTS/KEY OPTIONS: __
70 REWIND: _ 71 FILE CONDITIONERS: __

NEXT SPECIFICATION TYPE, ST, OR CMD: (ST_)
-----
-----
    
```

Like the previous file specification screen, again we key in our source statement, specify the next screen display, and then press the XMIT key. The next screen is the specification type display.

**SPECIFICATION TYPE DISPLAY**

```

ENTER SPECIFICATION TYPE: (IR)

**** SPECIFICATION TYPES ****

H - HEADER          IF - INPUT FIELD          A - ALTSEQ
F - FILE            C - CALCULATION        FT - *FILES
E - EXTENSION       OR - OUTPUT FILE          EQ - *EQUATE
L - LINE COUNTER    OF - OUTPUT FIELD          FF - FREE FORM/
IR - INPUT RECORD    T - TELECOMMUNICATION        COMMENTS
AU - AUTO REPORT    AC - AUTO REPORT (/COPY)     ** - TABLE/ARRAY
OPTIONS              DELIMITER
    
```

Select the specification screen we want next. Key in the appropriate abbreviation and press the XMIT key. The input specification screen (record identification) appears next.





### INPUT SPECIFICATION RECORD IDENTIFICATION

LINE - 4.0000

1 SEQUENCE NUMBER: 02010      6 FORM TYPE IR (RECORD ID)

7 FILENAME: INDATA\_ 14 \_      15 SEQUENCE: AA      17 NUMBER: \_      18 OPTION: \_

19 RECORD IDENTIFYING INDICATOR, DS, OR \*\*: 01

RECORD	POSITION	N = NOT	C/Z/D	CHARACTER
IDENTIFICATION	21 __	25 _	26 _	27 1
CODES	28 ____	32 _	33 _	34 _
	35 ____	39 _	40 _	41 _

42 STACKER SELECT: \_

NEXT SPECIFICATION TYPE, ST, OR, CMD: (IF\_)

-----

-----

**NOTE:**

*The remainder of the specification screens in this example (except for the final screen) operate in this manner. We key in the source statement, specify the next screen display (on the next screen identification line), and then transmit.*

### INPUT SPECIFICATION FIELD DESCRIPTION

LINE - 5.0000

1 SEQUENCE NUMBER: 02020      6 FORM TYPE IF

43 DATA FORMAT: \_      44 FROM: \_\_1      48 TO: \_\_80

52 DECIMAL POSITIONS: \_      53 FIELD NAME: INREC

59 CONTROL LEVEL: \_\_      61 MATCHING/CHAINING FIELDS: \_\_

63 FIELD RECORD RELATION: \_\_

65 PLUS: \_\_      67 MINUS: \_\_      69 ZERO/BLANK: \_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (0\_)

-----

-----



↓

## CALCULATION SPECIFICATION

LINE - 6.0000

1 SEQUENCE NUMBER: 03010      6 FORM TYPE C      7 CONTROL LEVEL: \_\_

INDICATORS: 9: 01    12: \_\_\_    15: \_\_\_                      18 FACTOR 1: COUNT

28 OPERATION: ADD      33 FACTOR 2: 1                      43 RESULT FIELD: COUNT

49 RESULT FIELD LENGTH: 33      52 DECIMAL POSITIONS: 0      53 HALF ADJUST: \_

ARITHMETIC:            PLUS                      MINUS      ZERO

COMPARE                1>2                      1<2        1=2

LOOKUP (FACTOR 2) IS:    HIGH                      LOW        EQUAL

RESULTING INDICATORS:                      54 \_\_\_                      56 \_\_\_    58 \_\_\_

60 COMMENTS: \_\_\_\_\_      NEXT SPECIFICATION TYPE, ST, OR CMD: (OR\_)

-----

-----

## OUTPUT SPECIFICATION (FILE IDENTIFICATION AND CONTROL)

LINE - 7.0000

1 SEQUENCE NUMBER: 04010      6 FORM TYPE OR

7 FILENAME: LIST                      14 \_ (AND/OR COL 14-16)

15 TYPE (H/D/T/E): H                      16 STACKER/FETCH: \_ (ADD/DEL COL 16-18)

17 SPACE BEFORE/AFTER: 2                      19 SKIP BEFORE: 07

21 SKIP AFTER: \_\_\_

OUTPUT INDICATORS: 23 IP    26 \_\_\_    29 \_\_\_

32 AUTO REPORT (\*AUTO): \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (OR\_)

-----

-----

↑



### OUTPUT SPECIFICATION (FILE IDENTIFICATION AND CONTROL)

LINE - 8.0000

1 SEQUENCE NUMBER: 04020      6 FORM TYPE OR

7 FILENAME: \_\_\_\_\_      14 OR (AND/OR COL 14-16)

15 TYPE (H/D/T/E): \_      16 STACKER/FETCH: \_ (ADD/DEL COL 16-18)

17 SPACE BEFORE/AFTER: \_      19 SKIP BEFORE: \_\_

21 SKIP AFTER: \_\_

OUTPUT INDICATORS: 23 OF 26 \_\_\_ 29 \_\_\_

32 AUTO REPORT (\*AUTO): \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (OF\_)

-----

-----

### OUTPUT SPECIFICATION (FIELD DESCRIPTION AND CONTROL)

LINE - 9.0000

1 SEQUENCE NUMBER: 04030      6 FORM TYPE OF

OUTPUT INDICATORS - 23 \_\_\_ 26 \_\_\_ 29 \_\_\_

32 FIELD NAME: \_\_\_\_\_

38 EDIT CODES: \_

39 BLANK AFTER: \_

40 END POSITION: \_\_28

44 DATA FORMAT: \_

45 CONSTANT OR EDIT WORD: 'STAFF FILE LISTING' \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (OR)

-----

-----



↓  
**OUTPUT SPECIFICATION (FILE IDENTIFICATION AND CONTROL)**

LINE - 10.0000

1 SEQUENCE NUMBER: 04040      6 FORM TYPE OR

7 FILENAME: \_\_\_\_\_      14 \_ (AND/OR COL 14-16)

15 TYPE (H/D/T/E): 0      16 STACKER/FETCH: \_ (ADD/DEL COL 16-18)

17 SPACE BEFORE/AFTER: 2      19 SKIP BEFORE: \_\_

21 SKIP AFTER: \_\_

OUTPUT INDICATORS: 23\_01 26 \_\_\_ 29 \_\_\_

32 AUTO REPORT (\*AUTO): \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (OF\_)

-----

-----

**OUTPUT SPECIFICATION (FIELD DESCRIPTION AND CONTROL)**

LINE - 11.0000

1 SEQUENCE NUMBER: 04050      6 FORM TYPE OF

OUTPUT INDICATORS - 23 \_\_\_ 26 \_\_\_ 29 \_\_\_

32 FIELD NAME: INREC

38 EDIT CODES: \_

39 BLANK AFTER: \_

40 END POSITION: \_\_80

44 DATA FORMAT: \_

45 CONSTANT OR EDIT WORD: \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (OR)

-----

-----

↑



**OUTPUT SPECIFICATION (FILE IDENTIFICATION AND CONTROL)**



LINE - 12.0000

1 SEQUENCE NUMBER: 04060      6 FORM TYPE OR

7 FILENAME: \_\_\_\_\_      14 \_ (AND/OR COL 14-16)

15 TYPE (H/D/T/E): I      16 STACKER/FETCH: \_ (ADD/DEL COL 16-18)

17 SPACE BEFORE/AFTER: 3      19 SKIP BEFORE: \_\_

21 SKIP AFTER: \_\_

OUTPUT INDICATORS: 23 LR 26 \_\_\_ 29 \_\_\_

32 AUTO REPORT (\*AUTO): \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (OF\_)

-----

-----

**OUTPUT SPECIFICATION (FIELD DESCRIPTION AND CONTROL)**

LINE - 13.0000

1 SEQUENCE NUMBER: 04070      6 FORM TYPE OF

OUTPUT INDICATORS - 23 \_\_\_ 26 \_\_\_ 29 \_\_\_

32 FIELD NAME: \_\_\_\_\_

38 EDIT CODES: \_

39 BLANK AFTER: \_

40 END POSITION: \_\_25

44 DATA FORMAT: \_

45 CONSTANT OR EDIT WORD: 'TOTAL RECORDS LOADED' \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (OF)

-----

-----



↓  
**OUTPUT SPECIFICATION (FIELD DESCRIPTION AND CONTROL)**

LINE - 14.0000

1 SEQUENCE NUMBER: 04080 6 FORM TYPE OF

OUTPUT INDICATORS - 23 \_\_\_ 26 \_\_\_ 29 \_\_\_

32 FIELD NAME: COUNT

38 EDIT CODES: \_

39 BLANK AFTER: \_

40 END POSITION: 35

44 DATA FORMAT: \_

45 CONSTANT OR EDIT WORD: \_\_\_\_\_

NEXT SPECIFICATION TYPE, ST, OR CMD: (CMD)

On the last output screen, we key in the final statement of the program. However, instead of selecting a particular specification screen or the specification type display on the next screen identification line, we key in CMD. This puts the RPG II editor into the update mode (EDT command mode) so we can store our program. Then we press the XMIT key.

③ @WRITE MO=SFIL,FIL=JCLFIL,VSN=PUBDSK,SIZE=2,SAT=Y  
 ④ @RPG END  
 ⑤ @DELETE  
 ⑥ // JOB JCLFIL  
 //SFIL RPG IN=(PUBDSK,FIL1)  
 ,ERRFIL=(PUBDSK,PROG1,MERRFIL)  
 /&  
 ⑦ @WRITE MO=JCLFIL,FIL=JCLFIL,VSN=PUBDSK,SIZE=2,SAT=Y  
 ⑧ @HALT  
 ⑨ AL MI,FIL=PROG1,VSN=PUBDSK,SIZE=2  
 ⑩ RV JCLFIL:(FIL2,PUBDSK)  
 ⑪ EDT @EFP  
 MERRFIL,PROG1,PUBDSK  
 ⑫ @EFP  
 - } Correct RPG II source  
 - } program using EDT  
 - } commands

Enter module name, file name (Iblname), and volume serial number of the error file. The screen displays the number of errors in your program.

↑



**Explanation:**

- 1 The LOGON command, plus the identification, connects the workstation to the operating system.
- 2 Activates the general editor (EDT) and the RPG II editor.
- 3 Stores the RPG II program in a file.
- 4 Terminates the RPG II editor.
- 5 Deletes all lines of the EDT workspace so no statements carry over into the next session.
- 6 Job control stream that compiles the program and creates an error file.
- 7 Stores the job control stream in a file.
- 8 Terminates the general editor.
- 9 Allocates the error file used by the error file processor.
- 10 Compiles the program.
- 11 Activates the general editor and the error file processor so you can correct compilation errors. A screen prompts you to enter the module name, file name, and volume serial number of the error file.
- 12 Displays the error and the statement so you can correct the errors using EDT.
- 13 Stores the corrected program back to its original file.
- 14 Terminates the error file processor.
- 15 Deletes all lines in the EDT workspace.
- 16 Data for the RPG II program.
- 17 Stores the data in a MIRAM file.
- 18 Deletes all lines in the EDT workspace.
- 19 Job control stream that executes the program.
- 20 Stores the job control stream in a file.
- 21 Terminates the general editor.
- 22 Executes the program.
- 23 Disconnects the workstation from the operating system.



- 24 Compilation listing of the source program.
  - 25 Printed report resulting from the execution of the program.
- Figure 1-1 illustrates the process you have just completed.

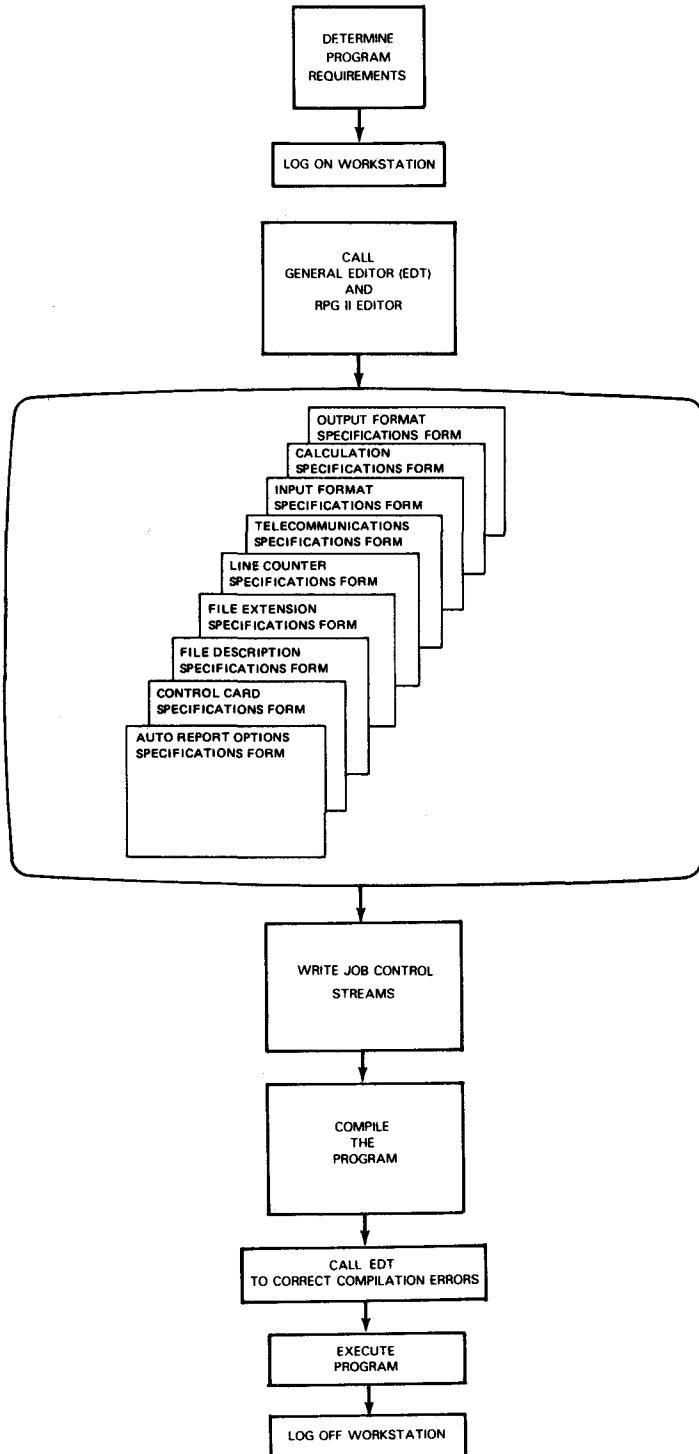


Figure 1-1. Using RPG II via Workstation

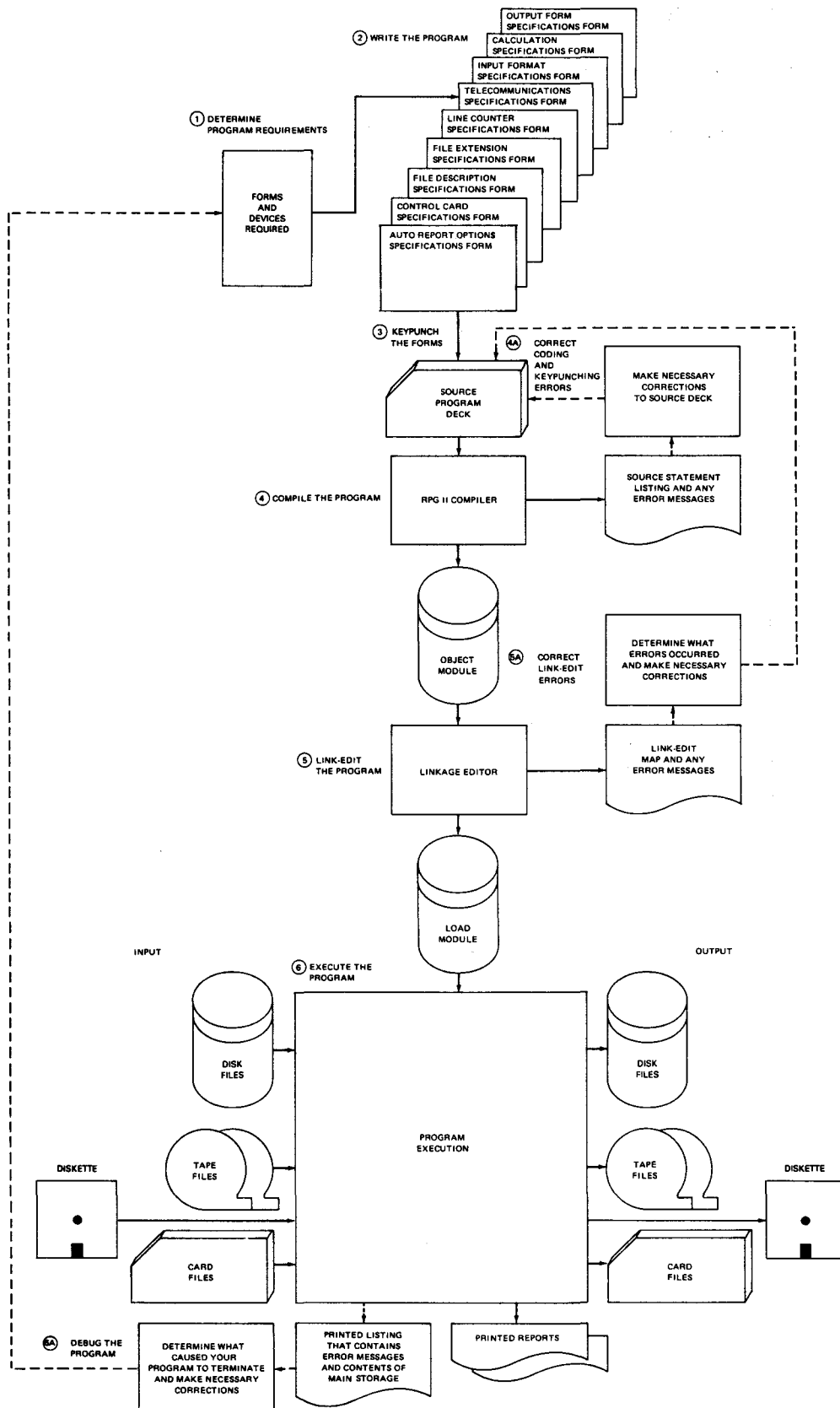


Figure 1-2. Using RPG II via Cards

### 1.3. HOW TO USE RPG II WITH CARDS

Figure 1-2 shows the logical steps you follow each time you write an RPG II program. You use the same procedure when you write a payroll program, inventory control program, or any other type of program. The circled numbers in the figure refer to the numbered explanations.

Explanation of steps in Figure 1-2:

#### 1. Determine Program Requirements

Before you start writing your program, determine what information your program must process and the results you want. For example, ask yourself:

- a. What is the format of my input files and how many input files are there?
- b. What fields in the input records will I use?
- c. What calculations must be performed?
- d. What format do I need for my output records?
- e. What totals do I have to accumulate?
- f. What format do I want for my printed reports?

#### 2. Write the Program

After you establish your program goals and decide on file and report formats, describe them by filling out the various RPG II specifications forms. RPG II uses the entries on these forms to generate the program.

- a. You describe the options you want to use with the output from auto report on the auto report options specifications form.
- b. You furnish special information about your program and your system on the control card specifications form.
- c. You describe all the files used or created by your program on the file description specifications form.
- d. If you use tables, arrays, input chaining, or record address files, you provide information about them on the file extension specifications form.
- e. If you need intermediate printer files (you are going to temporarily store your reports on tape or disk), you provide information about these files on the line counter specifications form.
- f. If you intend to send data to or receive data from a remote terminal, you provide information about this on the telecommunications form.

(continued)

- g. You describe your input files on the input format specifications form.
- h. You describe the calculations you want performed on the calculation specifications form.
- i. You describe how your data is to appear on your output files and the layout of your printed reports on the output format specifications form.

### 3. Keypunch the Forms

Once you fill out the specifications forms, keypunch them. One card is punched for each line on each specifications form. The resulting deck is called the source program deck.

### 4. Compile the Program

The source program deck is processed by the RPG II compiler at this point. The compilation generates an object program (the executable machine language for your program) and a listing that contains the source statements and a main storage assignment map. If errors occur during compilation, one or more compilation time messages, as shown in Appendix D, will be printed on the listing.

If the compilation is error free, perform step 5. If not, perform step 4A.

#### 4A. Correct Coding and Keypunching Errors

Correct the coding and keypunching errors detected during compilation and repeat step 4.

### 5. Link-Edit the Program

When you have an error-free object module, it must be link-edited. The linkage editor transforms the object module into a relocatable load module. The linkage editor also produces a link-edit map. This map lists: the link edit control stream and any error messages, unresolved references, and all reference definitions, including type, address, and phase number. It also contains a phase structure diagram, a main storage allocation map, and an ending message. If the link-edit is error-free, your program is now in a form that can be loaded and executed, as described in step 6. If not, perform step 5A.

#### 5A. Correct Link-Edit Errors

Examine the link-edit map to determine what errors occurred. Make the necessary corrections and repeat steps 4 and 5.

### 6. Execute the Program

The load module is loaded into the system at this point and processing begins. The input files are read, the calculations are performed on the input data, and the desired results are produced if there are no program logic errors. If so, your work on this program is finished. You now have a program you can execute any time you need to.



If errors occur during execution, your program terminates and a listing is printed that contains one or more execution time messages, as shown in the system messages programmer reference manual and the contents of main storage. If this happens, perform step 6A.

#### 6A. Debug the Program

Examine the printed listing to determine what caused your program to terminate. Make the necessary corrections and repeat the preceding steps.



### 1.4. RPG II GENERAL PROGRAM LOGIC

All programs generated by the RPG II compiler are handled in the same way. When the program is executed, each record processed goes through the same general program cycle. This cycle consists of three basic logical steps:

1. Read a record from an input file.
2. Perform calculations with the information contained in the record.
3. Produce the desired output.

In each program, there are two different points in time when the calculation and output operations are performed. These points in time are called *detail time* and *total time*.





## 2. RPG II File and Data Handling Capabilities

### 2.1. FILE CHARACTERISTICS

RPG II allows you to use files in a variety of ways. Consequently, when you write a program you must describe how each file is used in the program. You do this by specifying the *file characteristics* for each file. You specify what the file type is and what its function is in relation to the other files in your program.

#### 2.1.1. File Type

The types of files you can use in your program are input, output, combined, update, and display.

- Input File

An input file supplies input to your program. It can be a data file (data records to be processed), a table or array file, a record address file, or a tag file (a file created by using the ADDROUT option of sort/merge).

- Output File

An output file is any data file that is written, printed, or punched during the execution of your program.

- Combined File

A combined file is used for both input and output. It can be a communications file, a file that is on a card punch with the read/punch feature, a diskette, or a workstation file.

- Update File

An update file is a disk file in which the data records may be read, updated, and written back onto the file in the locations from which they were read.

- Display File

A display file contains data that is to be displayed on the system console. The system operator may modify the display data or he may make a reply.

### 2.1.2. File Designation

Except for output and display files, all other files in your program must have their function described; that is, a file is a primary data file, secondary data file, table or array file, record address or tag file, chained file, or demand file.

- Primary Data File

A primary data file is the main file from which data records are read during execution of your program. Only one primary file is permitted in a program.

- Secondary Data File

A secondary data file is any input file other than the primary file when more than one file is used for input.

- Table or Array File

A table or array file is a sequential input file, loaded at execution time, that contains table or array records.

- Record Address or Tag File

A record address or tag file is an input file that supplies parameters that indicate how a disk file is to be processed.

- Chained File

A chained file is a disk file that is processed randomly. It can be an input or update file chained by chaining indicators or by the CHAIN operation code.

- Demand File

A demand file is processed sequentially by the READ operation.

## 2.2. FILE ORGANIZATION

The arrangement of records in a file is called the *file organization*. RPG II is capable of processing files that are organized as sequential access method files (SAM), indexed sequential access method files (ISAM), direct access method files (DAM), multiple indexed random access method files (MIRAM), or indexed random access method files (IRAM). Before you write your program, decide which file organization is best suited to your application. If your files are not in the desired format, you must create these files before you write your program.

### 2.2.1. Sequential Files

In a sequential file, the arrangement of data records is determined by the order in which they are placed on the file. The first record placed on the file is always in the first record position, the second is always in the second record position, and so on. Card and magnetic tape files are always sequential files; however, disk files may also be indexed sequential or direct files. A sequential file is used when your application calls for processing an entire file starting with the first record, then the second, and so on until the file is exhausted.

### 2.2.2. Indexed Sequential Files

An indexed sequential file records data in ascending sequence by record key in prime data and overflow areas on a disk. These data areas consist of successive tracks on one or more cylinders in the disk file. The overflow area is used to store records that are added to a file. Figure 2-1 illustrates how data appears on an indexed or extended indexed sequential file.

In Figure 2-1, each block in the prime data area contains five records. Each number shown in the prime data area and the overflow area represents the highest key in each block. As each block is written in the prime data area, the highest key is entered in a block index. Entries are not made in the indexes for overflow records. After the last block is written, the last entry in each track on the block index is placed in the top index. The advantage of this type of file organization is that these indexes allow you to quickly locate the approximate position of data records in the file; thus, it is possible for you to retrieve records randomly or within specified ranges.

### 2.2.3. Direct Files

A direct file stores data records on disk in any sequence. Some of these records may be blank (unoccupied). Figure 2-2 provides an example of how data appears on a typical direct file.

Direct file organization is useful in those applications where your processing requirements are as follows:

- Anticipated file activity is low.
- File size is stable.
- Unordered (random) transactions are processed.
- Immediate inquiry capability is required.

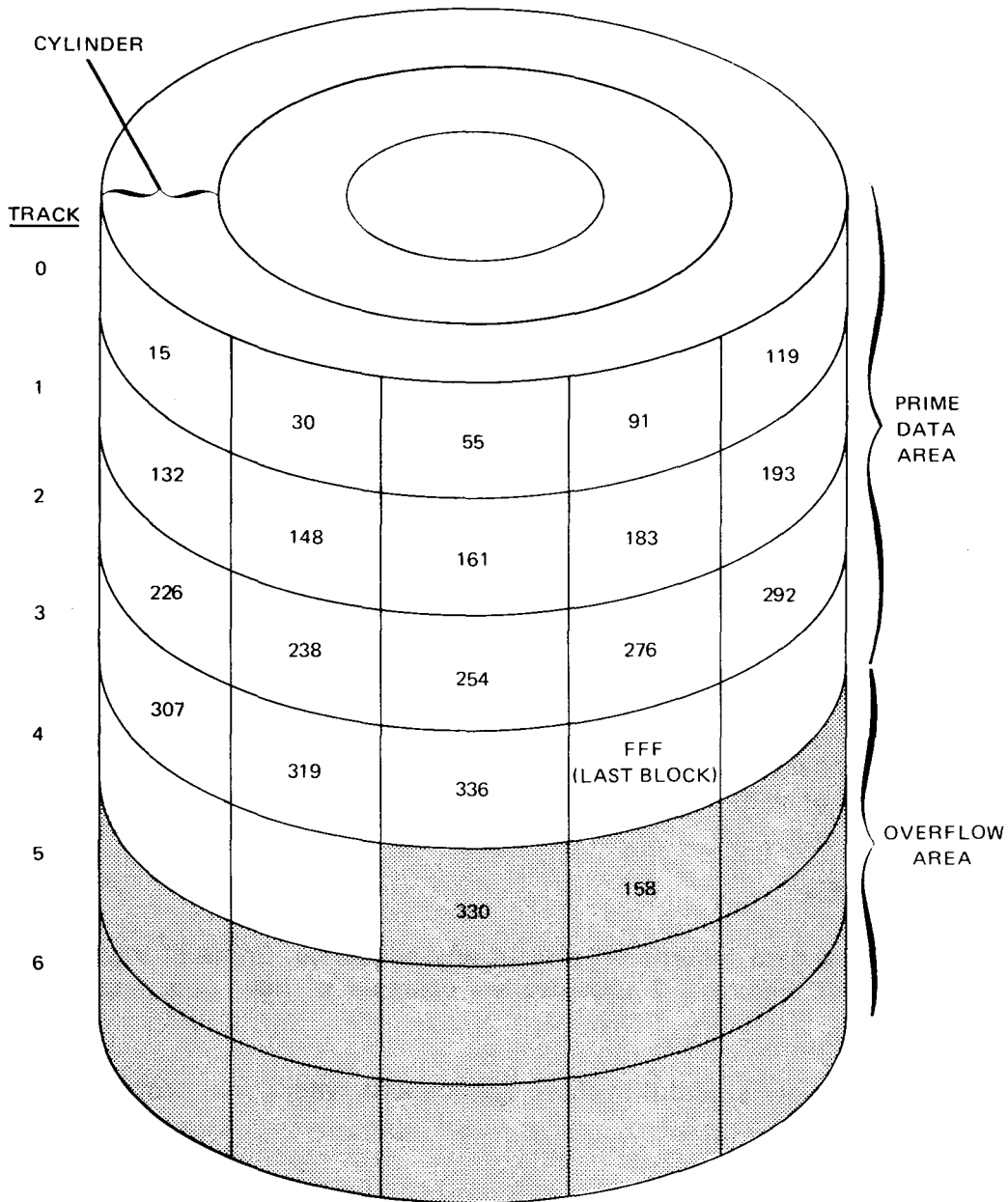
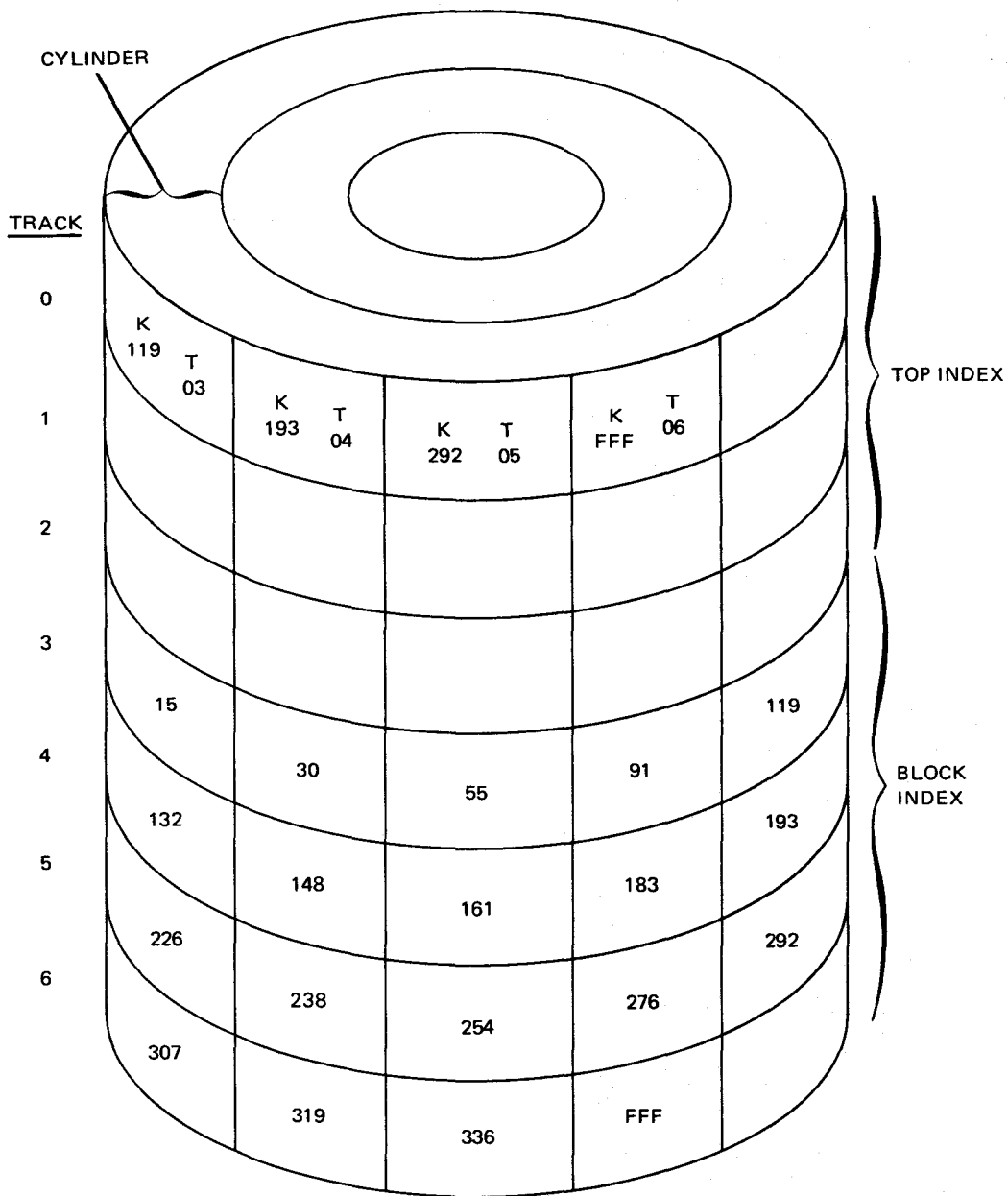


Figure 2-1. Indexed Sequential File Structure (Part 1 of 2)



LEGEND:

K = highest key in block index  
T = track in block index

Figure 2-1. Indexed Sequential File Structure (Part 2 of 2)

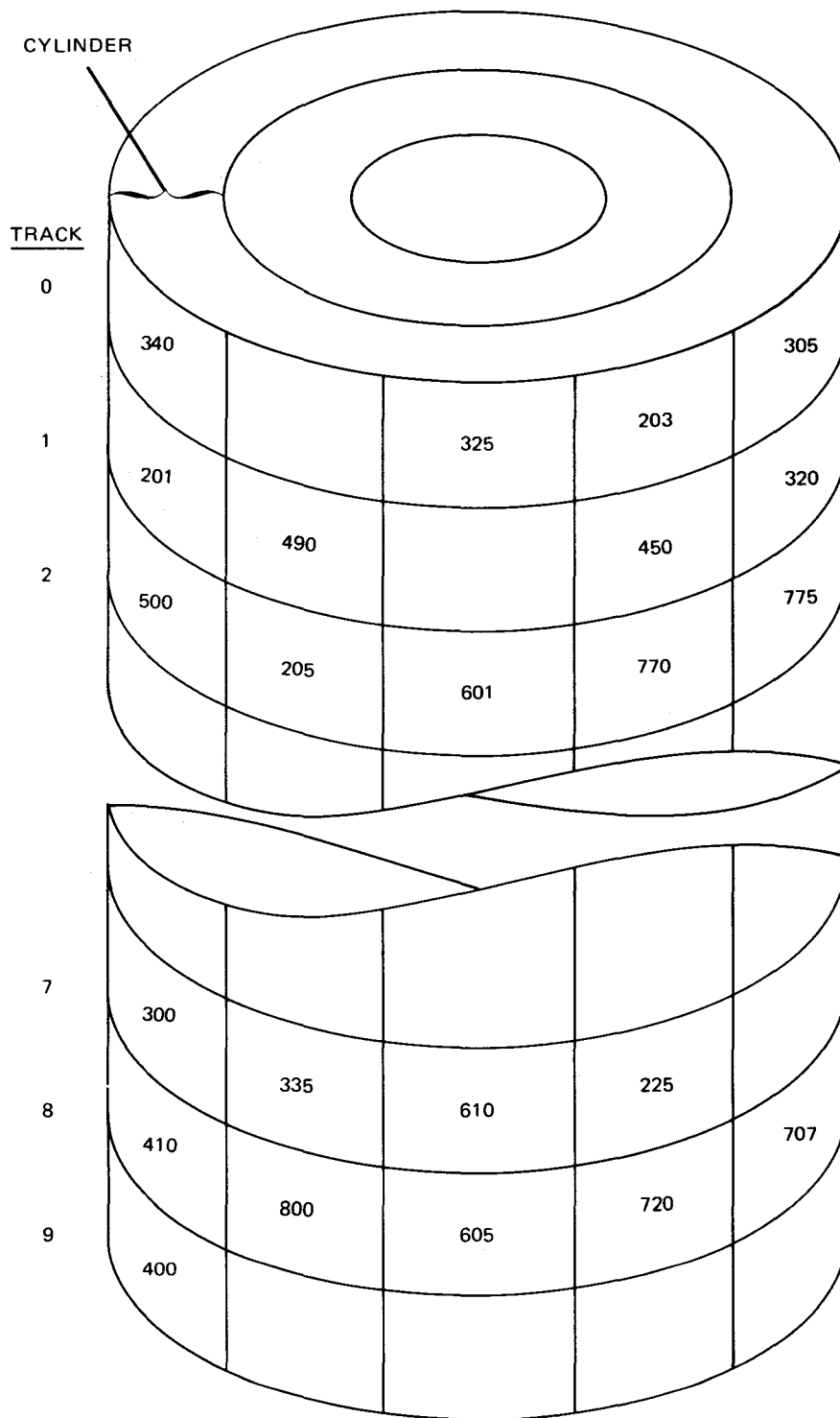


Figure 2-2. Direct File Structure



### 2.2.4. MIRAM Files

A multiple indexed random access method (MIRAM) file stores records and consequently allows you to retrieve records, either sequentially or randomly. MIRAM files are permitted when operating either under the System 80 environment or in a mixed-mode environment, that is, both the System 80 and Series 90 environment. To specify MIRAM files, code as if for IRAM (indexed random access method) files. See 18.2.3 for more details.

### 2.2.5. IRAM Files

An indexed random access method (IRAM) file stores records and consequently allows you to retrieve records, either sequentially or randomly. IRAM files are permitted when operating in a Series 90 environment.

## 2.3. FILE FORMAT

RPG II allows you to process or produce files containing fixed-length or variable-length records. When you write a program, you must describe how the records are recorded on each file used by the program.

The allowable file format for a file in a given application is governed by the type of medium the file is stored or written on. The allowable file formats are shown in Table 2-1.

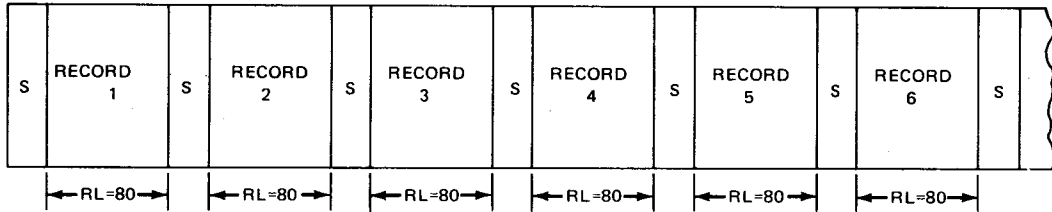
Table 2-1. Allowable File Formats

Storage or Recording Medium	File Organization	Allowable File Format
Cards	Sequential	Fixed, unblocked
Magnetic tape	Sequential	Fixed, unblocked Fixed, blocked Variable, unblocked Variable, blocked
Disk	Sequential	Fixed, unblocked Fixed, blocked Variable, unblocked Variable, blocked
	Indexed sequential	Fixed, unblocked Fixed, blocked Variable, unblocked Variable, blocked
	Direct	Fixed, unblocked
	MIRAM	Fixed Variable*

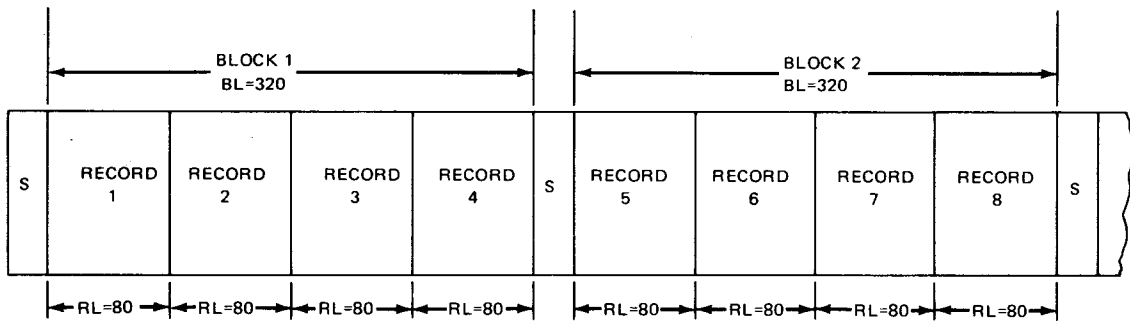
\* There is no provision for blocking in MIRAM files.

### 2.3.1. Fixed-Length Records

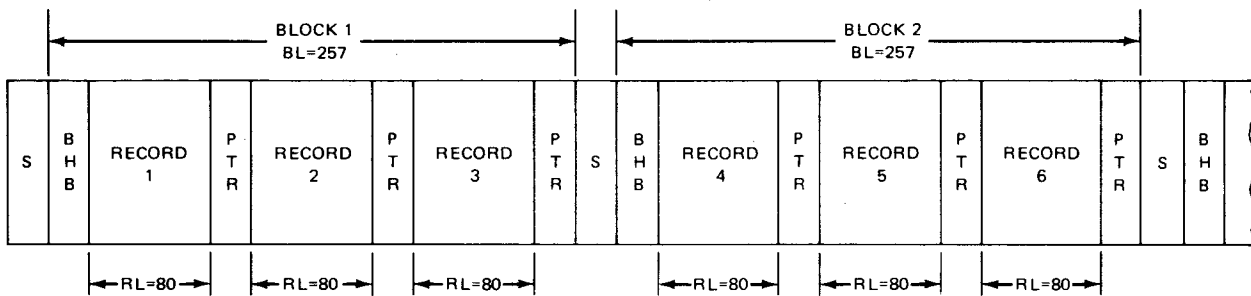
A file that contains fixed-length records is one in which each data record in the file is the same size (occupies the same number of character positions). The individual data records may appear on the file as individual data units or in groups of two or more units. In the former case, the file format is described as fixed-length, unblocked, and in the latter case as fixed-length, blocked. These formats are shown in Figure 2-3.



a. Fixed-length, unblocked format (sequential and direct files)



b. Fixed-length, blocked format (sequential files)



c. Fixed-length, blocked format (indexed sequential files)

**LEGEND:**

- BHB Block header bytes - two bytes that specify the count of the data in the block, including the BHB and PTR
- BL Block length
- PTR Pointer - five bytes in the form rrrbb, where rrr is the relative block number and bb is the byte displacement
- RL Record length
- S Space between records or blocks

Figure 2-3. Fixed-Length File Formats

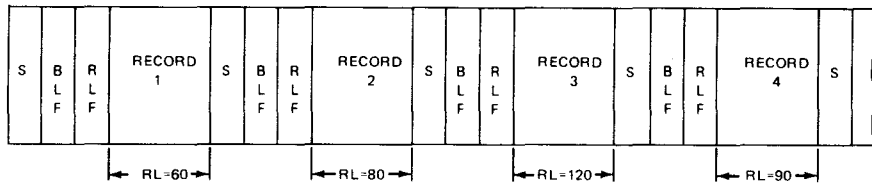
As shown in Figure 2-3, records in unblocked format are preceded and followed by a space or blank area. This space is used to distinguish between the beginning and end of a record. Similarly, there is a space between each block in the blocked format. The difference between the two formats is that there are fewer spaces between the data records in the blocked format. Input/output is more efficient with the blocked format because only one data access is required for more than one record as opposed to one data access for each record with the unblocked format. When unblocked records are specified for an indexed sequential file on the file description specifications form, these records are processed as blocked records in the format shown in Figure 2-3.

The block header bytes (BHB) and the pointer (PTR) are required control fields. These fields are automatically included when data records are written by RPG II on an indexed sequential file in fixed-length file format.

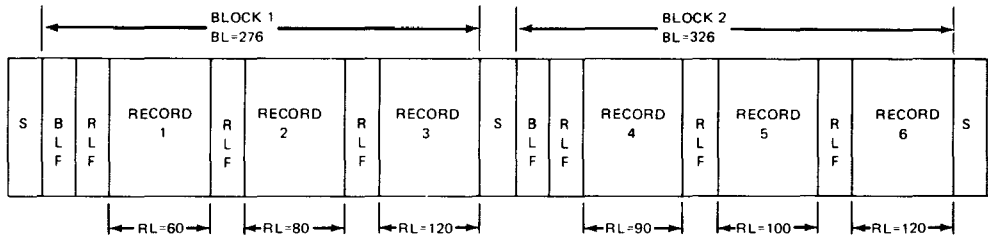
### 2.3.2. Variable-Length Records

A file that contains variable-length records is one in which the data records vary in size. (Each record does not occupy the same number of character positions.) Variable-length records may appear on the file in unblocked or blocked format. These formats are shown in Figure 2-4.

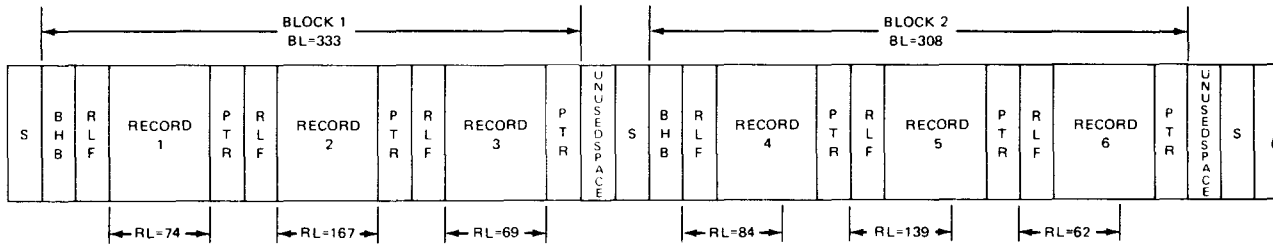
As shown in Figure 2-4, the formats for variable-length records vary from those for fixed-length records in that a field that specifies the record length is present in each record. The beginning and end of records in unblocked format and of blocks in blocked format are indicated in the same manner as for fixed-length records. Also, as with fixed-length records, input/output is more efficient if the records are blocked. Note that when variable-length records are specified for an indexed sequential file on the file description specifications form, these records will be processed as blocked records in the format shown in Figure 2-4. The block length field (BLF), the block header bytes (BHB), the pointer (PTR), and the record length field (RLF) are required control fields. The BLF and RLF are automatically included by RPG II when data records are written on a sequential file in variable-length file format. The BHB, RLF, and PTR are automatically included by RPG II when data records are written on an indexed sequential file in variable-length file format.



a. Variable-length, unblocked format (sequential files)



b. Variable-length, blocked format (sequential files)



c. Variable-length, blocked format (indexed sequential files)

LEGEND:

- BHB Block header bytes – two bytes that specify the count of the data in the block, including the BHB, PTR, and RLF
- BL Block length
- BLF Block length field – four bytes that specify the block length and include the data bytes RLF and BLF
- PTR Pointer – five bytes in the form rrrbb, where rrr is the relative block number and bb is the byte displacement
- RL Record length
- RLF Record length field. For indexed sequential files, two bytes that specify the length of the record, including the data bytes and the RLF. For sequential files, four bytes that specify the length of the record, including the data bytes and the RLF.
- S Space between records or blocks

Figure 2—4. Variable-Length File Formats

## 2.4. DATA FORMATS

The data processed by your program can be in alphanumeric, binary, packed numeric, or unpacked numeric format. When you write your program, you must describe how the data is formatted in each input and output field.

### 2.4.1. Alphanumeric Format

The format of an alphanumeric field is shown in Figure 2-5.

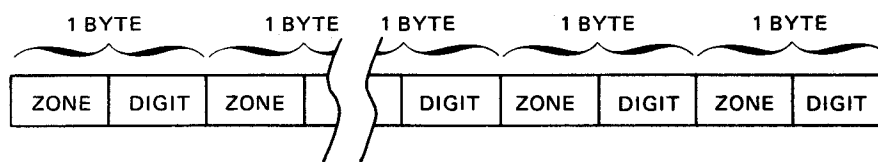


Figure 2-5. Alphanumeric Data Format

In this format, one alphanumeric character is placed in each byte (character position) of the field. Each byte consists of a zone portion and a digit portion.

### 2.4.2. Binary Format

The format of a binary field is shown in Figure 2-6.

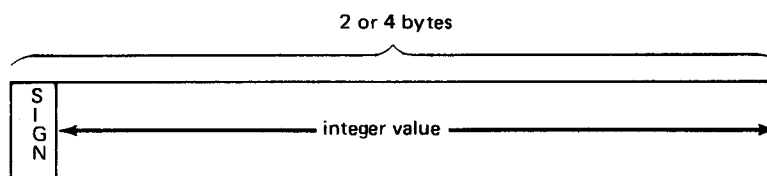


Figure 2-6. Binary Data Format

In this format, a field contains the sign in the leftmost bit position and an integer value in the remaining bit positions. Positive numbers have a 0 in the sign position. Negative numbers have a 1 in the sign position, and the integer value is the binary complement of the positive number.

### 2.4.3. Packed Numeric Format

The format of a packed numeric field is shown in Figure 2-7.

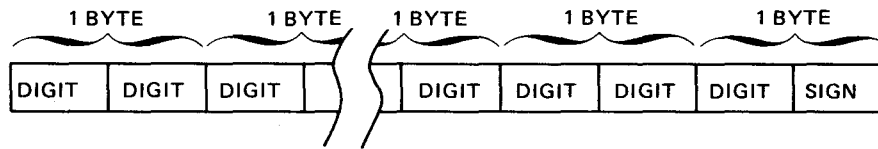


Figure 2-7. Packed Numeric Data Format

In this format, two decimal digits are placed in each byte (character position) except the rightmost byte of the field, which contains one digit and the sign.

### 2.4.4. Unpacked Numeric Format

The format of an unpacked numeric field is shown in Figure 2-8.

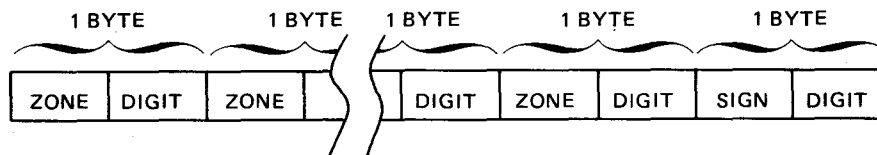


Figure 2-8. Unpacked Numeric Data Format

In this format, one decimal digit is placed in each byte (character position) of the field. Each byte except the rightmost byte consists of a zone portion and a digit portion. The rightmost byte contains the sign in the zone portion.

**PART 2. RPG II SPECIFICATIONS FORMS**

1954-1955



## 3. General Specification Information

### 3.1. TYPE AND NUMBER OF SPECIFICATIONS FORMS REQUIRED FOR A PROGRAM

As you know, you write RPG II programs by making entries on preprinted forms. The number and type of forms you use depends on the program's complexity. But at the very least, you must use four of the nine available specifications forms:

- Control Card Specifications Form

To specify the compilation mode and provide control information.

- File Description Specifications Form

To describe all files used or created by your program.

- Input Format Specifications Form

To identify the types of records and describe record fields within the input files.

- Output Format Specifications Form

To describe how data is to appear on your output reports.

If your program involves more than reading an input file, editing the information, and writing an output file or producing a printed report, you will have to use additional types of specifications forms. For example:

- Calculation Specifications Form

If your program requires operations.

- File Extension Specifications Form

If you are going to use tables, arrays, or use one file to retrieve records from another file.

- Telecommunications Specifications Form

If you intend to send data to or receive data from a remote terminal.

- Auto Report Options Specifications Form

If you want to specify the options you want to use with the output from auto report.

- Line Counter Specifications Form

If you want to store a report on an intermediate tape or disk file for printing at a later date.

As you can see, the *control card specifications form*, the *file description specifications form*, the *input format specifications form*, and the *output format specifications form* are required for all programs.

The *calculation specifications form*, the *file extension specifications form*, the *telecommunications specifications form*, the *auto report options specifications form*, and the *line counter specifications form* are used only when they are needed.

## 3.2. COMMON FIELDS

If you examine the RPG II specifications forms, you'll see that each form has a page number field, a line number field, a form type field, and a program identification field in the same position. These fields are provided to help you when you write your program. With the exception of the form type field, you do not have to use these fields. There are, however, certain advantages in using these fields that we'll cover in our discussions of the individual fields.

### 3.2.1. Page Number (Columns 1 and 2)

You use this field to number the specifications forms and indicate the order in which they should be processed. If you number the forms, they should be in ascending order from 01 through 99. Another advantage in using this field is that RPG II automatically checks the sequence of the statements during compilation and prints an error message on the source program listing if any are out of sequence. This sequence checking takes place whether the line number field is used or not. Table 3-1 gives you a summary of when RPG II performs sequence checking during compilation.

### 3.2.2. Line Number (Columns 3 through 5)

You use this field to number the individual lines on a specifications form. As you can see, columns 3 and 4 are preprinted, and column 5 is blank so that you can insert your own number. If you number your lines 010, 020, 030, and so on, you can insert up to nine lines between any two previously written lines. To insert a line, number the line in ascending sequence within the preceding and succeeding lines. Figure 3-1 is an example of inserting a line between line 010 and 020.



As you can see in Figure 3-1, you write the line being inserted after the last line with a preprinted line number. The line being inserted is numbered 011. Since line 010 and 020 already are written on the form, the presence of a number other than zero (1 through 9) in the third position (column 5) indicates that you want this line inserted between line 010 and 020. If you use insertion lines in your program, the cards that contain these lines must be inserted in their proper sequence in your source program deck before you compile your program.

As with the page number field, RPG II automatically checks the sequence of the statements during compilation and prints an error message on the source program listing if any are out of sequence. This sequence checking takes place whether the page number field is used or not. Table 3-1 gives you a summary of conditions when RPG II performs sequence checking during compilation.

From Table 3-1, we see that RPG II performs sequence checking during compilation when the page number or the line number field is used or when both fields are used.

*Table 3-1. RPG II Sequence Checking during Compilation*

Page Number Field Used? (Columns 1 and 2)	Line Number Field Used? (Columns 3 - 5)	Does RPG II Perform Sequence Checking?
Yes	No	Yes
No	Yes	Yes
Yes	Yes	Yes
No	No	No

### 3.2.3. Form Type (Column 6)

Each RPG II specifications form has a unique letter preprinted in this field. This letter identifies the specification type for each statement you write on a form. It must be punched on all source program cards when these forms are transcribed into a source program deck.

- C Identifies a calculations specifications statement
- E Identifies a file extension specifications statement
- F Identifies a file description specifications statement
- H Identifies a control card (header) statement
- I Identifies an input format specifications statement
- L Identifies a line counter specifications statement

- O Identifies an output format specifications statement
- T Identifies a telecommunications specifications statement
- U Identifies an auto report options specifications statement

One obvious advantage of using these identifying letters is that if your source program deck is disarranged, you can readily place the cards back in order by using the statement identifying letter along with the page and the line number.

### 3.2.4. Compiler Directives (Columns 7 through 74)

Three directives control the source listing produced by the compiler: /EJECT, /SPACE n, and /TITLE. You can place them anywhere in the RPG II program. ←

- /EJECT (columns 7 through 12)

This directive prints all specifications after this entry on a new page. The /EJECT directive itself is not printed in the compilation listing.

- /SPACE n (columns 7 through 14) ←

The /SPACE n directive causes blank lines immediately following the directive on the compilation listing. The spacing occurs only at the point where the /SPACE n directive appears with n specifying the number of lines (1 to 3) to be spaced. A blank is required in column 13 and n in column 14. If the number you specify is greater than the number of lines remaining on the current page, the next RPG II specification is printed on the next page. If you don't specify a number from 1 to 3, the default is a single blank line. The /SPACE n directive itself is not printed on the compilation listing. ←

- /TITLE (columns 7 through 74)

The /TITLE directive specifies the heading information that is printed at the top of each page of the compilation listing. Compiler heading information is printed before the /TITLE heading information on the first page. You enter the /TITLE heading information in columns 14 through 74. A blank is required in column 13. /TITLE ejects to the top of the next page before printing the title information. If a title is specified on the first page, /TITLE must be the first program statement. You can use more than one /TITLE directive in a program. The heading information stays in effect until a new /TITLE directive is used. The /TITLE directive itself is not printed on the compilation listing. ↓ ↑

### 3.2.5. Comments (Columns 7 through 74)

You can insert a comment at any point among your specification statements by placing an asterisk (\*) in column 7 and the comment text in columns 8 through 74. The asterisk in column 7 indicates to RPG II that the associated text is a comment rather than a specification statement. When your program is compiled, RPG II prints any comment lines that are present on the source program listing. These lines are not included in your program.

The main use for comment lines is to write notes or reminders that will help you during program testing. For example, you can use a comment line to identify each section of your program or to indicate what a series of statements is used for. Lines 130 and 140 in Figure 3-1 show an example of the way in which you can use comment lines.

### **3.2.6. Program Identification (Columns 75 through 80)**

You use this field to identify individual specification statements. This is true for all RPG II specifications forms, except the control specifications form. If you use this field, RPG II treats the entry as a comment and prints it on the source program listing opposite the associated specification statement. Any combination of alphanumeric characters can be used in this field.

On the control card specifications form, you use this field to name your program. This is a specialized use of this field so we'll cover it later in our discussion of the control card specifications form.

## 4. Control Card Specifications Form

### 4.1. GENERAL DESCRIPTION

You must have a control card specifications form (Figure 4-1) in all programs. This form specifies the compilation mode you want used when your program is compiled, assigns a unique name to your program, and supplies other control information.

### 4.2. FORM ENTRIES

In the following subsections you'll see what the fields are used for and how your entries in these fields affect your program.

*NOTE:*

*Columns labeled "not used" in Figure 4-1 are not checked for blanks. Any characters you enter are ignored.*

#### 4.2.1. Compilation Mode (Column 7)

You use this field to specify the compilation mode for your program. There are four compilation modes: SPERRY UNIVAC OS/3 mode, SPERRY UNIVAC 9200/9300 mode, IBM 360/20 mode, and IBM System/3 or System/34 mode. The OS/3 mode is the normal mode and should be specified for all new programs you write. The other modes are provided so that programs you have written for these systems do not have to be rewritten. Instead, you specify the required compilation in this field, and the RPG II compiler will compile the program.

If you are writing a new program, leave this field blank.

If your program is written for the IBM 360/20 systems, enter 2 in this field. For other IBM 360 or 370 systems, leave this field blank.

If your program is written for the SPERRY UNIVAC 9200/9300 System, enter 3 in this field.

If your program is written for the IBM System/3 or System/34, enter 4 in this field.





#### 4.2.2. Error Analysis Dump (Column 8)

You use this field to specify if you want a formatted error analysis dump of main storage printed when an error occurs during the execution of your program. When you test your program, you should specify an error analysis dump because, if an error occurs, the dump is formatted to help you find the cause of the error. After your program is error free, you should change the entry in this field and recompile your program. We recommend this because the error analysis dump option requires an additional 5642 (hexadecimal 160A) bytes of main storage. ←

If you do not want an error analysis dump or you have specified operator control (1 in column 9), leave this field blank.

If you want an error analysis dump, enter a D in this field. The error analysis dump uses the PRNTR file, so you must include a // LFD PRNTR statement in the job stream used to execute your programs.

If you do not want an error analysis dump but you want an unformatted cancel dump, leave this field blank, but include a //△OPTION△DUMP statement in the job control stream used to execute your program.

For IMS action programs, this field is ignored.

#### 4.2.3. Operator Control (Column 9)

You use this field to control program termination when a halt indicator is set on during program execution. A halt indicator can be one that you specified in your program (H1 through H9), or it can be the H0 indicator, which is set on as a result of an error condition.

If you do not want operator control or you have specified an error analysis dump (D in column 8), leave this field blank. This field is ignored for IMS action programs.

If you want operator control, enter a 1 in this field. This causes the message

```
RPG031 RPGII OPERATOR CONTROL, TYPE IN AVAILABLE OPTION (0, 1, 2, 3)
```

to be displayed on the system console when a halt indicator is set on. If a halt indicator specified in your program (H1 through H9) is set on, the message

```
RPG028 USER SET HALT INDICATORS ARE:
```

```
Hn, Hn, . . . Hn
```

is displayed on the system console preceding the operator control message. If the H0 indicator is set on as the result of an error, the appropriate execution time error message in the form

```
RPGnnn explanatory text
```

is displayed on the system console preceding the operator control message. In either case, type in the appropriate option; that is:

- 0 Continues. Control is returned to the program and processing continues at the instruction immediately following the error condition.
- 1 Bypass. The remainder of the program cycle is bypassed, and the next record is read.
- 2 Controlled termination. LR processing is performed.
- 3 Immediate termination. Program terminates immediately.

Only option 0, 2, or 3 should be used in reply to the RPG028 message. The operator control options that can be used with the individual execution time error messages are shown in the system messages programmer/operator reference.

#### 4.2.4. Generate Debug Code (Column 15)

You use this field to include one or more DEBUG operations in your program on the calculation specifications form. It determines if the output produced by the DEBUG operation is printed during the execution of your program. The DEBUG operation is a program testing aid that prints the indicators that are on and any specified field or literal when it is executed. If you use DEBUG operations in your program, you should specify that you want the DEBUG operation output printed when you test your program. After your program is error free, you should change the entry in this field and recompile your program.

If you do not want the DEBUG operation output printed or you did not include any DEBUG operations, leave this field blank.

If you want the DEBUG operation output printed, enter a 1 in this field.

#### 4.2.5. Currency Sign (Column 18)

Use this field to specify a currency sign other than the dollar sign (\$) in edit words.

If you want to use the dollar sign (\$) as the currency symbol, leave this field blank.

If you want to use a different symbol as the currency sign, enter any character except:

0 \* , & . - C R

These characters have a special meaning when used in edit codes and words. The character you select can be used as a fixed or floating currency symbol when output data is edited.

#### 4.2.6. Inverted Print (Column 21)

You use this field to specify the system date format and how commas and decimal points are used to punctuate numeric fields. If you leave this field blank, the system date format is mm/dd/yy (mm=month, dd=day, yy=year), and commas and decimal points are used to punctuate numeric fields in the normal manner.

If you enter a D in this field, the system date format is dd/mm/yy, and commas and decimal points are used to punctuate numeric fields in the normal manner.

If you enter an I in this field, the system date format is dd.mm.yy, and the use of commas and decimal points is reversed; the comma is used as a decimal point and the decimal point as a comma. For example, \$3,600.42 would appear as \$3.600,42 in this format.

If you enter a J in this field, it is the same as if you entered an I except that all zero balances or balances with a zero to the left of the decimal comma are printed with one leading zero. For example, 0,00 or 0,79.

#### 4.2.7. Alternate Collating Sequence (Column 26)

You use this field to specify a collating sequence other than the normal EBCDIC collating sequence. The EBCDIC collating sequence (arrangement order in ascending value for comparison purposes) is blanks followed by special characters, alphabetics (A through Z), and numerics. If the data you are processing requires a different collating sequence or you want certain characters to be considered equal, you must supply this information via ALTSEQ (alternate sequence) records in your input deck at compilation time.

If you intend to use the normal EBCDIC collating sequence, leave this field blank. If you are going to use an alternate collating sequence, enter an S in this field.

#### 4.2.8. Binary Search (Column 31)

You use this field to specify that a binary search is to be used on sequenced tables and arrays for LOKUP operations. If you leave this field blank, a sequential search is used for LOKUP operations. If you enter a 1 in this field, a binary search is used on all sequenced tables and arrays in the program. All rules for LOKUP operations apply. In addition:

- Tables and arrays specified as sequenced must remain sequenced during program execution.
- At least two elements must be searched. For a subscripted array, the subscript must be at least one less than the array size.

Tables and arrays should contain at least 50 elements for the binary search to be efficient. Search time improves as the number of table or array elements is increased. For example, the binary search is approximately eight times faster than a sequential search on a table of 500 elements and 60 times faster on a table of 2500 elements.

#### 4.2.9. Sign Handling (Column 40)

You use this field to specify whether you want the signs of numeric fields checked when data is moved to or from them. A hexadecimal D is the negative sign for both input and output fields. A hexadecimal C is the positive sign for input fields and a hexadecimal F is the positive sign for output fields.

If you want the signs of both the input and output fields checked to see if they are positive or negative, either enter an S or leave this field blank. If a field does not contain a sign, RPG II will insert one. It is your responsibility to ensure that signs of packed numeric input fields (P in column 43 of input format specifications form) are valid. If a hexadecimal 6 is used to indicate a negative sign in an input field, it is automatically changed to a hexadecimal D when the field is read. If you do not want the signs of the input or output fields checked, enter a B in this field.

If you do not want the signs of the input fields checked, enter an I in this field.

If you do not want the signs of the output fields checked, enter an O in this field.

If input sign control is specified, program check island code is generated. If you do not want this feature, enter an I or a B in this field.

#### 4.2.10. Forms Alignment (Column 41)

You use this field to specify whether or not you want first page forms alignment for your printed reports when your program is executed. We suggest that you use this because it allows you to make sure that your forms are properly aligned in the printer before you start. If you specify first page forms alignment, the first detail line conditioned by the 1P (first page) indicator for the first printer file is printed and your program waits. A message that asks the operator if the form is aligned is then displayed on the system console. If the form is aligned, the operator types in Y on the system console and program processing resumes. If the form is not aligned, the operator can adjust the form and type in N to have the line printed again.

If you do not want to use first page forms alignment, leave this field blank. If you want to use first page forms alignment, enter a 1 in this field.

For IMS action programs, this field is ignored.

#### 4.2.11. Indicator Initialization (Column 42)

You use this field to specify whether the zero or blank indicators specified on the input or calculation specifications form are to be set on when your program begins execution.

If you want the zero or blank indicators set on, enter an S in this field. When you use this entry and a field on the output format specifications form is to be blanked after it is placed in the output record, the first indicator that tests this field for zero or blank is set on after the field is placed in the output record.

#### 4.2.12. File Translation (Column 43)

You use this field to specify that all data in the input or output files you select is to be translated in accordance with a translation table that defines the characters that are to replace existing characters in the input and output data. If you want file translation, you must supply the file translation table via file translation records in your input deck at compilation time.

If you are not using file translation, leave this field blank.

If file translation is required, enter an F in this field.

#### 4.2.13. CCA Name (Columns 70 through 73)

You use this field to specify the name of the ICAM communications control area (CCA) when telecommunications are used in your program.

If you do not use telecommunications in your program, leave this field blank. If you do use telecommunications, enter the name of the CCA in this field.

#### 4.2.14. Subroutine or Action Program (Column 74)

You use this field to specify whether the program is to be compiled as a main program, a subroutine, or an IMS action program.

If the program is to be compiled as a main program, leave this field blank.

If the program is to be compiled as a subroutine, enter an S in this field.

If the program is to be compiled as an IMS action program, enter an A in this field.

#### 4.2.15. Program Identification (Columns 75 through 80)

You use this field to give a unique name to your program. This name is assigned to your program during compilation. If you don't specify a name in this field, RPG II automatically assigns RPGOBJ as your program name.

If you do not want to name your program, leave this field blank.

If you want to name your program, enter a 1- to 6-character name in this field. The first character must be an alphabetic character, A through Z. The remaining characters may be any combination of alphanumeric characters.

When you make an entry in this field, it must be left-justified.



## 5. File Description Specifications Form

### 5.1. GENERAL DESCRIPTION

You must have a file description specifications form (Figure 5-1) in all your programs. It describes the files you are going to use. You describe each file by naming it. Then, you tell how the file is used, the size of the data records and how they are arranged, where the key field is located in the records, how the file is processed, the type of input/output device the file is stored on or to be written on, how file labels are handled, the amount of main storage you want reserved for the top index if you have an indexed sequential (ISAM) file, the percentage of each cylinder that is reserved for cylinder overflow if you have an indexed sequential file that you are going to add records to at a later time, how the tape reel is to be rewound if you have a magnetic tape file, and what external indicator controls the file.

### 5.2. FORM ENTRIES

In the following subsections you'll see what each field is used for and how your entries in these fields affect your program.

#### 5.2.1. File Name (Columns 7 through 13)

You identify the file you want to describe by putting the name in this field. The file name consists of from one to seven alphanumeric characters. The first character must be an alphabetic character A through Z or a special character \$, #, @, /, %, ?, +, or &. The remaining characters may be alphabetic, A through Z; numeric, 0 through 9; or the special characters \$, #, @, /, %, ?, +, and &. You cannot use any other special characters or the space character. When you make an entry in this field, it must be left-justified. The \$ must not be the first character in a console output file.

If you are using a SPERRY UNIVAC 9200/9300, IBM System/3, or IBM 360/20 source program, RPG II accepts an 8-character file name.

When you use table or array files, the point in time when these files are to be loaded determines if they are described on this form. If you are going to load a table or array file at compilation time, you must not enter the name of this file in this field. You identify this file on the file extension specifications form. If you are going to load a table or array file at execution time, you must enter the name of this file in this field and also on the file extension specifications form.





### 5.2.2. File Type (Column 15)

You use this field to specify the file type for each file in your program. The entry you make in this field in conjunction with the entry in the file designation field (column 16) tells how you want this file used in your program. Entries and instructions on file use are summarized in Table 5-1, which follows the description of the file designation field (5.2.3). A file can be a combined file, a display file, an input file, an output file, or an update file. You specify the file type by entering a letter in this field:

- C Specifies a combined file
- D Specifies a display file
- I Specifies an input file
- O Specifies an output file
- U Specifies an update file

A combined file is used for both input and output. It can only be a communications file, a diskette, a workstation terminal, or a card file that resides on a card punch device that contains the prepunch read feature. This specification must not be used for IMS action programs.

A display file contains data for display on the system console. The system operator may modify the displayed data by making a reply. This specification must not be used for IMS action programs.

An input file contains data records, a record address file, an array file, a table file, or a tag file (a file created by using the ADDRROUT option of the sort/merge).

An output file is any data, array, or table file that is written, punched, or printed during the execution of the program.

An update file is a disk file containing records that may be retrieved, updated, and written back on the file in locations from which they were read. The updating process must not alter the record characteristics in any way. The field length, record length, and field location must remain the same. When an update file is used, data can only be updated at detail time, except in the case of data from a chained update file. This file can be updated at either detail or total time. The file name of the update file must be specified on both the input and output format specifications forms. Only those fields that are to be updated should be specified on the output format specifications form.

When you write your program, you can list files of different types in any order. You must have at least one output file (output, combined, or update). You can use a maximum of 20 files in your program; however, there are specific restrictions on how many files can be used to perform certain functions. These restrictions are shown in Table 5-1.

### 5.2.3. File Designation (Column 16)

You use this field to specify the function of each input, combined, and update file in your program. If you use a display or output file, this field must be left blank. The entry you make in this field in conjunction with the entry in the file type field (column 15) tells how you want this file used in your program.

These entries and how files may be used are summarized in Table 5-1.

Table 5-1. File Use

File Type (Column 15)	File Designation (Column 16)	File Use	Maximum Number of Files Allowed
I (input) or C (combined) or U (update)	P	Primary file	1
	S	Secondary file with matching fields	8
	S	Secondary file without matching fields	18
	D	Demand file	18
I (input)	R	Record address or tag file	1
I (input) or U (update)	C	Chained file using input chaining (C1—C9)	9
I (input) or U (update) or O (output)	C	Chained file using CHAIN operation	18
I (input) or C (combined)	T	Table or array file	18
D (display)	Blank	Console input/output file	1
O (output)	Blank	Report files	8
	Blank	Other output files	18

A file can be a chained file, a demand file, a primary file, a record address or tag file, a secondary file, or a table or array file. You specify the file designation by entering a letter in this field.

- C Specifies a chained file
- D Specifies a demand file
- P Specifies a primary file
- R Specifies a record address or tag file
- S Specifies a secondary file
- T Specifies a table or array file

A chained file is an indexed sequential file that is processed randomly. It may be an input or update file processed by input chaining indicators (C1 through C9) specified on the input format specifications form or by a CHAIN operation on the calculation specifications form, or a direct file that is being created using the CHAIN operation.

A demand file is a file processed by a READ operation on the calculation specifications form. It can be an input, combined, or update file.

A primary file is the main file that records are read from during program execution. When you have more than one input file in your program (multifile processing), the primary file controls the order in which the records are selected for processing. If you don't specify a primary file, the first secondary file is assumed primary. If there are no primary or secondary files, you are responsible for terminating your program by setting on the LR indicator.

**NOTE:**

*If you choose a workstation, you can designate it as a primary file or a demand file. If you designate a workstation file as a primary file, secondary files are prohibited. If you designate a workstation file as a demand file, a primary file is not required.*

A record address file supplies the parameters that control how an indexed sequential or direct file is processed. This file contains upper and lower address limits when only a certain portion of an indexed sequential file is to be processed (sequential processing between limits), or it contains record keys when the file is to be processed randomly. For a direct file, the record address file contains relative record numbers.

A tag file supplies the required processing sequence for a sequential disk file that is to be processed randomly. You use the ADDRROUT option of sort/merge to create a tag file. The tag file is made up of the addresses of the records on the sequential file. The order that the addresses are written on the tag file determines how the records on the sequential file are selected for processing.

A secondary file is any file except the primary file when multifile processing is involved. Secondary files are processed in the order that they appear on this form.

A table or array file is a file that contains table or array records. Make an entry in this field only for a table or array that is loaded at execution time. You do not make an entry if a table or array is to be loaded at compilation time or written on an output file. For an IMS action program, a table or array file may not be a SAM file; however, it may be a sequential, nonindexed IRAM file.

When you write your program, you can only have one record address or tag file in your program and this file must also be defined on the file extension specifications form. If you use an input chaining file (C1 through C9 chaining) or table or array files at execution time, these files must also be defined on the file extension specifications form.

#### **5.2.4. End of File (Column 17)**

You use this field to specify which input, combined, or update file is to be completely processed before the last record indicator, LR, is turned on. The file must be used as a primary, secondary, or record address file. If the file is a display file, output file, table file, or array file, this field must be left blank.

If you leave this field blank for all files, all the records in each file are processed before the LR indicator is turned on. If it is not blank for all files, all the records from this file may or may not be processed before the LR indicator is turned on.

If you enter an E in this field, all the records in this file are processed before the LR indicator is turned on.

If you specify an E for only one file, the LR indicator is turned on when all the records from this file have been processed, regardless of whether all records in the other files in the program have been processed.

If you specify E for more than one file, the last record indicator is turned on when all records from these files have been processed.

If you specify E for all files, this is the same as leaving this field blank for all files.

#### **5.2.5. Sequence (Column 18)**

You use this field to specify the sequence of the data records within an input, combined, or update file when you want RPG II to check the sequence of the data records in a file or when you want to process more than one file using the matching records technique. If you make an entry in this field, you must make a matching field entry in columns 61 and 62 of the input format specifications form for each input field that contains sequencing information.

If you specify matching field entries for only one file, RPG II will check the sequence of the records in that file.

If you specify matching field entries for more than one file, RPG II will check the sequence of the records in each file and process the files, using the matching records technique. In this case, you do not have to make a sequence entry for each file because RPG II assumes that the sequence entry for the first file that has matching fields applies to all files. If you make a sequence entry for the other files, these entries must be the same as for the first file. If they are not, the sequence for the first file applies.

If the matching fields are in ascending order, leave this field blank or enter an A in it. If the matching fields are in descending order, enter a D in this field.

Sequence checking is not performed in the IBM System/3 mode if this field is left blank.

This field must be blank for demand, output, record address and display files, and for any file processed randomly.

#### **5.2.6. File Format (Column 19)**

You use this field to specify format of the data records on a file. All records on a file can be the same or variable in length. If they are the same length, the file contains fixed-length records. If the records vary in length, the file contains variable-length records. This field is used in conjunction with the entries in the block length field (columns 20 through 23) and the record length field (columns 24 through 27).

If all the records in the file are the same length or the file is a table, array, workstation terminal, or special file, enter an F in this field. F must be specified for all disk files in the IBM System/3 mode.

If the records vary in length or the file is a line counter file, enter a V in this field.

If the file contains variable-length ASCII (American Standard Code for Information Interchange) records, enter a D in this field.

An entry is not required for card, printer, display, or direct files. ←

#### **5.2.7. Block Length (Columns 20 through 23)**

You use this field to specify the number of data characters contained in each block in a blocked file (a file where the data records are grouped into blocks containing two or more records). The entries in this field are used in conjunction with the entries in the file format field (column 19) and the record length field (columns 24 through 27). These entries are summarized in Table 5-2, which follows the description of record length (5.2.8). If the file is an unblocked file, leave this field blank.

If the file is a blocked file, enter the logical number of data characters (bytes) in the block in this field. The entry must be right-justified; leading zeros may be omitted. The allowable block length ranges for the devices that blocked files can be stored on or written on are:

<u>Device</u>	<u>Minimum Block Length</u>	<u>Maximum Block Length</u>
UNISERVO VI-C or 10 Magnetic Tape Subsystem	18	8191
UNISERVO 12/16 or 20 Magnetic Tape Subsystem	18	9999
8411 disk subsystem	6	3618
8413 diskette subsystem	1	1023
8414 disk subsystem	6	7287
8415 disk subsystem	6	9999
8416 disk subsystem	6	9999
→ 8417 disk subsystem	6	9999
8418 disk subsystem	6	9999
→ 8419 disk subsystem	6	9999
8424 disk subsystem	6	9999
8425 disk subsystem	6	9999
8430 disk subsystem	6	9999
8433 disk subsystem	6	9999
0768 printer subsystem	3	132
0770 printer subsystem	3	132 or 160*
0773 printer subsystem	3	120, 132* or 144*
0776 printer subsystem	3	120 or 136*
0778 printer subsystem	3	120 or 136*

\*Requires a printer subsystem equipped with special features

If your file is a fixed-format file on tape or disk, the block length must be a multiple of the logical record length. If it is not, the block length will be automatically increased to the nearest multiple. If it is a variable format file, the block length must be equal to or greater than the logical record length.

When using a console, workstation terminal, or special file, this field must be left blank.

At compilation time, RPG II adjusts the block length specified in columns 20 through 23 to allow space for data management block and record descriptors. The adjustments are based on the following formulas.

■ Tape and Nonindexed Sequential Disk Files

- Fixed format files             $b = nr$
- Line counter files             $b = r + 9$  if  $l = r$  or unblocked  
                                       $b = l + 4$  if  $l \neq r$
- Other variable format files  $b = r + 8$  if  $l = r$  or unblocked  
                                       $b = l + 4$  if  $l \neq r$

■ Indexed Disk Files

- Fixed format files             $b = 2 + n (r+5)$
- Variable format files         $b = r + 9$  if  $l = r$  or unblocked  
                                       $b = l + 2$  if  $l \neq r$

where:

b  
Is total number of physical bytes required for data and control fields per block.

n  
Is number of records per block.

r  
Is logical record length specified in columns 24 through 27.

l  
Is logical block length specified in columns 20 through 23.

In the IBM System/3 mode, the block length does not affect the way records are written on a disk file. Its function is to specify the amount of main storage to use for the input/output area.

For IMS action programs, a block length specification will be ignored. However, a block length must be specified during IMS configuration.

### 5.2.8. Record Length (Columns 24 through 27)

You use this field to specify the number of data characters in each record in a file. The entries in this field are used in conjunction with the entries in the file format field (column 19) and the block length field (columns 20 through 23). These entries are summarized in Table 5-2.

If the file is other than a tape file, disk file, workstation file, or a file stored on or to be written on a user-supported device, you can leave this field blank, and the default record length for the device is assumed.

If the file is a tape file, disk file, workstation file, or a file stored on or to be written on a user-supported device, or if you do not want the default record length assumed, you must enter the logical number of characters (bytes) in a record in this field. The entry must be right-justified; leading zeros may be omitted. The allowable record length ranges for the various devices are:

<u>Device</u>	<u>Maximum Record Length</u>	<u>Default Record Length</u>
0716, 0717, or 0719 card reader subsystem	80	80
0716, 0717, or 0719 card reader subsystem with 51-column read feature	51	51
0716, 0717, or 0719 card reader subsystem with 66-column read feature	66	66
0716 card reader subsystem with 96-column read feature	96	80
0768 printer subsystem	132	120
0770 printer subsystem	132 or 160*	120
0773 printer subsystem	120, 132*, or 144*	120
0776 printer subsystem	120 or 136*	120
0778 printer subsystem	120 or 136*	120
UNISCOPE 100 Display Terminal (system console)	60	60

\*Requires a printer subsystem equipped with special features



<u>Device</u>	<u>Maximum Record Length</u>	<u>Default Record Length</u>
UNISERVO VI-C or 10 Magnetic Tape Subsystem	8191	246
UNISERVO 12/16 or 20 Magnetic Tape Subsystem	9999	246
8411 disk subsystem	3618	-
8413 diskette subsystem	128	80
8414 disk subsystem	7287	-
8415 disk subsystem	9999	-
8416 disk subsystem	9999	-
8418 disk subsystem	9999	-
8424 disk subsystem	9999	-
8425 disk subsystem	9999	-
8430 disk subsystem	9999	-
8433 disk subsystem	9999	-
User-supported device (SPECIAL)	9999	-
Workstation terminal	1920	-

If the lines on a printed report are not longer than 120 characters, you should specify that you want your report file written in device-independent format. By doing this, your program can use any type of printer subsystem that is available at execution time. If you want a report file written in this format, you must specify a record length of 120 characters or less in this field and specify that any printer subsystem can be used in the job control stream device assignment statements for your program.

The entry you make in this field is used by RPG II to calculate the actual record length according to the formulas that follow. The actual record length is the sum of the data characters in the record and any required control fields. These control fields are present in all files that are created using OS/3 data management; that is, the control fields are present in your input files, and they will be included in your output files.

The RPG II actual record length calculation formulas are:

■ **Tape and Nonindexed Sequential Disk Files**

- Fixed format files  $r = b$
- Line counter files  $r = b + 5$
- Other variable format files  $r = b + 4$

▶ ■ **Indexed Disk Files**

- Fixed format files  $r = b + 5$
- Variable format files  $r = b + 7$

where:

$r$   
Is actual record length.

$b$   
Is logical record length specified in columns 24 through 27.

*Table 5—2. Summary of File Format, Block Length, and Record Length Entries*

Record Format	Entries		
	Column 19 File Format	Columns 20 through 23 Block Length	Columns 24 through 27 Record Length
Fixed, unblocked	F	Blank or same as record length	Required
Fixed, blocked	F	Required	Required
Variable, unblocked	V	Blank or same as record length	Required
Variable, blocked	V	Required	Required
Variable, ASCII	D	Required	Required

### 5.2.9. File Processing Mode (Column 28)

You use this field to specify the file processing mode for a disk file. If the file is a card or tape file, this field must be left blank. The entries in this field are used in conjunction with the entries in the record address type field (column 31) and the file organization field (column 32). These entries are summarized in Table 5-3, which follows the description of file organization (5.2.12). If the file is a sequential file, indexed file, or direct file that is processed sequentially or a record address or tag file, leave this field blank. This field is left blank for record address or tag files because these are sequential files that are processed sequentially. This field should also be left blank if the file is a sequential, indexed, or direct file that is processed consecutively in the IBM System/3 mode.

Enter an L in this field if the file is an indexed file that is processed sequentially between limits by using a record address file to supply the upper and lower limits or by using a SETLL operation to supply the lower limit.

Enter an R in this field if the file is an indexed file that is processed randomly by using a record address file, a chaining file, a tag file, or the CHAIN or REFER operation. You also use R if the file is a sequential file that is processed randomly using a tag file, or if it is a direct file that is processed randomly by using the CHAIN operation to supply relative record numbers, or if you are creating a direct file with relative record numbers supplied with the CHAIN operation. ←

### 5.2.10. Key or Record Address Field Length (Columns 29 and 30)

You use this field to specify the length of the entries in a record address or tag file, or the length of the key field when the record address type (column 31) is specified as alphanumeric (A) or packed decimal (P).

The minimum key length for an indexed file is 3.

The minimum key length for a MIRAM file is 1.

The maximum key length for IRAM, MIRAM, and IBM System/3 mode files is 80.

If the file is a tag file, 10 must be entered in this field.

The entry in this field must be right-justified; leading zeros may be omitted.

### 5.2.11. Record Address Type (Column 31)

You use this field to specify how data records are retrieved from a file. The entries in this field are used in conjunction with the entries in the file processing mode field (column 28) and the file organization field (column 32). These entries are summarized in Table 5-3.

Data records can be retrieved sequentially, by record key, by record address, or by relative record number. If data records are retrieved sequentially, they are retrieved in the order they appear on the file; that is, the first record on the file is the first one retrieved and so on. If the data records are retrieved by record key, the record keys are supplied by a record address file, chaining file, or the CHAIN or REFER operation, and the records are retrieved randomly from the data file based on the record keys that are supplied. If data records are retrieved by record address, the record addresses are supplied by a tag file, and the records are retrieved from the data file based on the record addresses that are supplied. If the data records are retrieved by relative record number (the position of the record relative to the beginning of the file), the relative record numbers are supplied by the CHAIN operation and the data records are retrieved based on the relative record numbers that are supplied.

If the data records are retrieved sequentially, leave this field blank. This entry is not required for a sequential file. If any other entry is made for a sequential file, it is treated as if it were a blank. When your program is executed in the IBM System/3 mode, indexed files are processed consecutively.

If the data records are retrieved by alphanumeric keys, enter an A in this field. This entry can be used only for an indexed file.

If the data records are retrieved by a tag file, enter an I in this field. This entry is also interpreted in this way when your program is executed in the SPERRY UNIVAC 9200/9300 mode, IBM 360/20 mode, or IBM System/3 mode.

If your program is executed in the SPERRY UNIVAC 9200/9300 mode, a K in this field is interpreted the same as if it were an A. If your program is executed in the OS/3 mode, this entry is interpreted as if it were a P.

If the data records are retrieved by packed decimal key, enter a P in this field. This entry can be used only for an indexed file.

If the data records are retrieved by relative record number, enter an R in this field.

#### **5.2.12. File Organization (Column 32)**

You use this field to specify how a file is organized and how many input/output areas you want assigned to that file. A file can be a sequential file with one or two input/output areas, an indexed file with one or two input/output areas, a direct file, or a tag file. The entries in this field are used in conjunction with the file processing mode field (column 28) and the record address type field (column 31). These entries are summarized in Table 5-3.

If the file is a sequential file or record address file, leave this field blank. This field must also be left blank if the file is a combined table, array, or special file or if the file is a card input or output file and you want to use stacker selection.

Table 5-3. Summary of File Processing Mode, Record Address Type, and File Organization Column Entries

File Retrieval	Column Entries		
	28 File Processing Mode	31 Record Address Type	32 File Organization
Tag (ADDROUT) file	Blank	Blank	T
Sequential file processing Sequentially or record address file Sequentially with additional I/O areas requested Randomly by using tag (ADDROUT) file to supply addresses	Blank Blank R	Blank Blank I	Blank 2 D
Indexed file processing Sequentially Sequentially with additional I/O areas requested (ISAM files only) Sequentially between limits by using record address (RA) file or SETL operation to supply upper and lower limits Sequentially between limits by using record address (RA) file or SETL operation to supply upper and lower limits with additional I/O areas requested (ISAM files only) Randomly by using a record address (RA) file, C1 to C9 chaining, or CHAIN or REFER operation code	Blank Blank L L R	A or P A or P A or P A or P A or P	I X I X I
Direct file processing Randomly by relative record numbers supplied with CHAIN operation	R	R	D
IBM System/3 mode Tag (ADDROUT) file Sequential file processed randomly by using a tag (ADDROUT) file to supply addresses Indexed file processed randomly by a tag (ADDROUT) file Direct file processed randomly by relative record number File processed consecutively	Blank R R R Blank	I I I Blank Blank	T Blank I Blank Blank
SPERRY UNIVAC 9200/9300 mode Indexed file processed sequentially between limits by using a record address (RA) file to supply upper and lower limits	L	K	I



If the file is a sequential file and you want two input/output areas assigned to the file, enter a 2 in this field. If 1 or 3 through 9 is specified, it is treated as if 2 were specified. If the file is a card input or output file and you enter 2 in this field, you cannot use the stacker selection. If the file is a nonsequential file and you enter a 2 in this field, the entry is ignored.

If the file is an indexed file, enter an I in this field.

If the file is a direct file or a sequential file processed randomly by using a tag (ADDROUT) file to supply addresses, enter a D in this field.

If the file is a tag file, enter a T in this field.

If the file is an indexed file and you want two input/output areas assigned to the file, enter an X in this field. This is allowed only for ISAM files. It is not allowed in IBM System/3 or System/34 mode, or for IRAM or MIRAM files.

For IMS action programs, a tag (ADDROUT) or a record address file may not be a SAM file. However, either file may be a nonindexed, sequential IRAM file. Tape or disk input files may not be sequential access (SAM) files; they may be nonindexed, sequential IRAM files.

Additional I/O areas may not be specified for an IMS action program. They must be specified, however, during IMS configuration.

### **5.2.13. Overflow Indicator (Columns 33 and 34)**

You use this field to specify a unique overflow indicator for a printer or line counter file. The indicator that you specify will be set on when page overflow occurs during the printing of the file. This field must be left blank for all files other than printer or line counter files. The entries in this field are used in conjunction with the entries in the output indicators fields (columns 23 through 31) on the output format specifications form. These entries control the printing of heading information and any associated processing when page overflow occurs. The interaction of these entries is summarized in Table 5-4. You specify the overflow indicator by specifying two letters in this field. You can use OA through OG, or OV. If you have more than one printer or line counter file in your program, you must specify a different overflow indicator for each file. If you do not specify an overflow indicator for a printer or line counter file, an automatic skip to the top of the next page occurs when overflow is detected. OF is assumed as the overflow indicator if your program is a SPERRY UNIVAC 9200/9300 program.

Table 5-4. Summary of Interaction between the Overflow and Output Indicators

Entries		Action
File Description Specifications Form, Columns 33 and 34 (Overflow indicator)	Output Format Specifications Form, Columns 23 through 31 (Output Indicator)	
Blank	Blank	Automatic skip to the top of the next page when the overflow condition occurs
Blank	OA—OG, or OV	An error will be noted at compilation time. An automatic skip will be made to the top of the next page when the overflow condition occurs.
OA—OG, OV	Same entry as on file description specifications form	Sets overflow indicator on and performs overflow processing
OA—OG, OV	Blank	Sets overflow indicator on and continues printing (overflow processing is ignored) without skipping to the top of the next page

#### 5.2.14. Key Field Starting Location (Columns 35 through 38)

You use this field to specify the starting location of the key field in the data records on an indexed file. This entry can range from 0001 through 9999. The entry must be right-justified; leading zeros may be omitted. If the file is an indexed file, you must make an entry in this field.

#### 5.2.15. Extension or Line Counter Code (Column 39)

You use this field to indicate to the RPG II compiler that additional information concerning this file is provided on the file extension specifications form or the line counter specifications form.

If the file is a table or array file loaded at execution time, a chaining file, or a record address file, you must enter an E in this field. The E indicates that additional information concerning the file will be found on the file extension specifications form.

If the file is a line counter file (a print image file that is to be stored on tape or disk), you must enter an L in this field.

#### 5.2.16. Device (Columns 40 through 46)

You use this field to specify the type of input/output device that each file in your program resides on or is to be written on. The names that you use to specify the various types of input/output devices are shown in Table 5-5.

Table 5-5. Summary of Device Field Entries

Entry				Input/Output Devices
SPERRY UNIVAC OS/3 Mode	SPERRY UNIVAC 9200/300 Mode	IBM 360/20 Mode	IBM System/3 Mode	
CTLRDR*				Job control stream reader
READER*	READER	READ01, 2501	READ01 MFCU1	Series 90: 0716 or 0717 card reader System 80: 0719 card reader
STUB51				0716, 0717, or 0719 card reader with 51-column read feature
STUB66				0716, 0717, or 0719 card reader with 66-column read feature
PUNCH*	PUNCH ROWPNCH	PUNCH20, PUNCH42, READ40, READ42, MFCM1	READ42 MFCU1 MFCU2 MFCM1 MFCM2	Series 90: 0604 or 0605 card punch System 80: 0608 card punch
CRP*	CRP,RRP	CRP20 READ20 READ40, READ42, MFCM1, MFCM2	READ42 MFCU2 MFCU1 MFCM1 MFCM2	Series 90: 0604 or 0605 card punch with prepunch read feature, or 8413 diskette subsystem System 80: 0608 card punch
PRINTER*	PRINT16, PRINT48, PRINT63, PRINTDR	PRINTER, PRINTLF, PRINTUF,	PRINTER PRINTR2 PRINT84	Series 90: 0768, 0770, 0773, 0776, or 0778 printer System 80: 0776 or 0789 printer
TAPE	TAPE, TAPE7	TAPE	TAPE	Series 90: UNISERVO V1-C, UNISERVO 12/16, UNISERVO 10, UNISERVO 14, or UNISERVO 20 magnetic tape subsystem
DISC DISK	DISC	DISK, DISK11, DISK11F, DISK14	DISK DISK40 DISK45	Series 90: 8411, 8414, 8415, 8416, 8418, 8424, 8425, 8430, or 8433 disk System 80: 8417 or 8419 disk
DISCVS				Series 90: 8411, 8414, or 8430 disk
DISKET	DISKET	DISKET	DISKET	Series 90: 8413 diskette subsystem System 80: 8420 or 8422 diskette subsystem
CONSOLE*			CONSOLE CRT77	UNISCOPE 100 display terminal or workstation
REMOTE			BSCA	Remote communications terminal
SPECIAL			SPECIAL	Any user-supported input/output device
WORKSTN	WORKSTN	WORKSTN	WORKSTN	SPERRY UNIVAC 3560 or 3561 workstation

\*These devices may not be specified for an IMS action program.



The entry in this field must be left-justified. Devices that are valid for one mode are valid for every mode. For example, READ01 is valid for the OS/3 mode.

If you have a disk file and you use a variable sector disk whose sector sizes are not 256-byte multiples, you may specify DISCVS in this field. If you do not, the block size will be increased to the next highest 256-byte multiple.

To access a data set label diskette with RPG, specify DISKET in the device field. To access a format label diskette, specify DISK in this field. For instructions on how to prepare diskettes for information on the job control to be used with diskettes, refer to consolidated data management concepts and facilities, UP-9978 (current version).

If CONSOLE is specified and the file is specified as a display file (D in column 15), the DSPLY operation must be used in conjunction with this file. Note that only those characters in the basic UNISCOPE 100 display terminal character set will be displayed and that nondisplayable characters in an output record will be displayed as blanks. The internal representation of the nondisplayable characters remains unchanged.

If CONSOLE is specified and the file is specified as an input file (I in column 15), this will allow you to enter input records via the system console. If this is done, the following message is displayed with the appropriate job message identification number.

#### RPGII REQUESTS INPUT FOR file name

Input records can then be entered via the system console beginning with the appropriate job identification number. If the first typed character is a space, the system assumes it is a separator character and it is not passed to the program.

The records that are entered will be treated as any other input records. Each character must be typed in. The characters must be typed in as you would punch a card; that is, the fields of the record must be left- or right-justified as required and you must space where blanks are required in the record. When all of the characters in the record have been typed in, press the TRANSMIT key. If more characters have been typed in than are specified in columns 24 through 27, the excess characters are truncated.

If you specify CONSOLE and also enter the CONSOLE=filename parameter in the // PARAM job control statement, the operator can enter input to an executing RPG II program from a workstation in a manner that is compatible with the IBM System/32 and IBM System/34. Workstation prompts generated from the field name on the input format specifications form prompt the operator to enter the data. See 13.16 for more information on interactive data entry.

Except for table or array files, CTRLDR can only be specified once in a program.

If a 0605 card punch subsystem is used, the record length for a punch output file must be an even number.

When you choose a workstation terminal, the nature of data is determined by screen formats that you create. For example, if you want to add a new employee to a payroll program, you call up the payroll format, which might display NAME:, followed by 20 dots, and EMPLOYEE NUMBER:, followed by six dots. You write the employee's name and number over the dots. For details on creating formats, see the screen format services concepts and facilities, UP-8802 (current version).

### 5.2.17. Labels (Column 53)

You can use this field to specify the type of labels on tape or disk files and how you want them processed. A tape or disk file can be unlabeled, can contain standard labels followed by user labels, can contain nonstandard labels, or can contain standard labels. If the file is unlabeled or SPECIAL is specified in columns 40 through 46, leave this field blank.

If the file contains standard labels followed by user labels, enter an E in this field. RPG II processes the standard labels and then transfers control to the external label processing routine you have supplied. The name of your label processing routine must be entered in columns 54 through 59, and you must include this routine in your program at link-edit time. You must supply a label processing routine if the file is an output file. If it is an input file and you do not want to process the user labels, you do not have to supply a label processing routine.

If the file contains nonstandard labels, enter an N in this field. RPG II immediately transfers control to the external label processing routine you have supplied. The name of your label processing routine must be entered in columns 54 through 57, and you must include this routine in your program at link-edit time. You must supply a label processing routine if the file is an output file. If it is an input file and the labels are separated from the data by a tape mark, you do not have to supply a label processing routine.

If the file contains standard labels, enter an S in this field.

This field is ignored for IMS action programs.

### 5.2.18. Continuation Lines (Columns 53 through 69)

You use these fields to indicate that the tape file defined on the preceding line is an ASCII file and to provide additional information about the file.

→ For IBM System/3 mode or MIRAM files, these fields are also used to provide an additional amount of main storage for the index buffer of the indexed disk file defined on the preceding line.

For workstation files, the continuation line is used to name the field that contains the workstation identification or to handle multiple workstations and workstation error processing.

For multikey MIRAM files (both background and IMS action programs), the continuation line is used to specify the complete key structure of an existing or new file.

For MIRAM file sharing, the continuation line specifies a file sharing environment for the file.

For MIRAM files, the continuation line supports record control bytes.

The entries in these fields are summarized in Table 5-6.

Table 5—6. Summary of Continuation Line Entries

Type of File	Entries					
	Continuation Line Indicator (Column 53)	Option (Columns 54 through 59)	Entry (Columns 60 through 65)	Entry (Columns 66 and 67)	Entry (Column 68)	Entry (Column 69)
ASCII input or output tape file	K	ASCII	Blank	Blank	Blank	Blank
ASCII input file with block prefix	K	ASCII	Blank (Line 1)	Blank	Blank	Blank
	K	BUOFF	0-99 (Line 2)	Blank	Blank	Blank
IBM System/3 mode or MIRAM disk file with index buffer	K	INDEX	1-9	Blank	Blank	Blank
Workstation	K	ID	fieldname	Blank	Blank	Blank
	K	NUM	1-99*	Blank	Blank	Blank
	K	SAVDS	data structure name	Blank	Blank	Blank
	K	INFDS	data structure name	Blank	Blank	Blank
	K	IND	1-99	Blank	Blank	Blank
	K	INFSR	subroutine name	Blank	Blank	Blank
Multikey MIRAM file	K	KEY1-KEY5	starting location	1-80	Blank or D	Blank or C
Multikey MIRAM file action program	K	KEY1-KEY5	starting location	1-80	Blank or D	Blank, P, or D
MIRAM file sharing	K	ACCESS	EXC,EXCR, SRD,SRDO, SADD	Blank	Blank	Blank
MIRAM file	K	RCB	Blank	Blank	Blank	Blank

\* If the ID field is not used, the number may range from 1 to 255.

### 5.2.18.1. Continuation Line Indicator (Column 53)

You use this field to indicate whether or not this line is a continuation line. If this is not a continuation line, leave this field blank.

If this is a continuation line, enter a K in this field. When you specify a continuation line, columns 7 through 52 must be blank, the option field (columns 54 through 59) must contain an entry, and the entry fields (columns 60 through 69) may require an entry.

If you are dealing with a tape file, one or two continuation lines may be specified.

### 5.2.18.2. Option (Columns 54 through 59)

→ You use this field to specify ASCII tape files. For IBM System/3 mode or MIRAM files, this field is used to specify that an additional amount of main storage is required for the index buffer of an indexed file. If you are dealing with an ASCII tape input or output tape file, enter ASCII in this field. If an input file contains a block prefix, you must enter BUFOFF in this field on a second continuation line and this line must immediately follow the first continuation line. The second continuation line must also specify the length of the block prefix in the entry field (columns 60 through 75).

If you are dealing with an indexed disk file (IBM System/3 mode only) that requires additional main storage for the index buffer, enter INDEX in this field and specify the required amount of additional main storage in the entry field (columns 60 through 65).

If you are dealing with workstation files, you can use any or all of the following six options.

- ID

For the identification continuation line.

- NUM

For the maximum number of workstations that can be attached to the file at the same time. It is required for the SAVDS and IND options.

- SAVDS

For the data structure saved and restored for each workstation attached to the file. It requires the NUM option.

- INFDS

For the data structure that contains the identification of the error that occurred and the workstation operation that caused the error. It provides status and record information to the program.

- IND

For the number of indicators saved and restored for each workstation attached to the file. It requires the NUM option.

- INFSR

For the user-written calculation subroutine that receives control when a workstation error occurs during a NEXT, READ, primary input, EXCPT, or normal cycle output.

If you are dealing with multikey MIRAM files, enter KEYn, where n is a number from 1 through 5. These entries must be in sequence, and complete, on successive source statements. For example, if the file has three keys, you must specify KEY1, KEY2, and KEY3 in that order.

If you are dealing with MIRAM file sharing, enter ACCESS.

If you are dealing with MIRAM files containing a record control byte, enter RCB. Once a file is created with a record control byte, the RCB continuation statement must always be specified for the file.

### 5.2.18.3. Entry (Columns 60 through 65)

You use this field to specify the length of the block prefix for ASCII tape input files. In the IBM System/3 mode, this field is used to specify the amount of additional main storage required for the index buffer of an indexed file.

If you are dealing with an ASCII tape input file that has a block prefix (BUFOFF specified in columns 54 through 59), enter the length of the block prefix in this field. This entry must be right-justified and may range from 0 to 99. If the ASCII tape input file is a variable-blocked or variable-unblocked file, enter 0 or 4 in this field.

If you are dealing with an indexed file (IBM System/3 mode or MIRAM) that requires additional main storage for the index buffer (INDEX specified in columns 54 through 59), enter the amount of main storage in this field. This entry must be right-justified and may range from 1 to 9. The entry represents the number of 256-byte increments of main storage that are required. If INDEX is used when creating the file, it must be used in each job that accesses it. ←

When dealing with workstation files:

- If you entered ID in columns 54 through 59, then enter the name of a 2-character alphanumeric field for the workstation identification continuation line. The ID field is updated whenever a record is read from the WORKSTN file. The ID field contains a number (starting from 01) that identifies the workstation.
- If you entered NUM in columns 54 through 59, then enter the maximum number of workstations (right-justified) that can be attached to the file at the same time. The number may range from 1 to 99. If the ID field is not used, the number may range from 1 to 255. If you don't specify a number, 1 is the default. If this number is exceeded at execution time, an error message is issued and the program terminates.

- If you entered SAVDS in columns 54 through 59, then enter the name of the data structure to be saved and restored for each workstation attached to the file. If you didn't specify the SAVDS option or if you specified 1 for the NUM option, no saving and restoring is done.
- If you entered INFDS in columns 54 through 59, then enter the name of the data structure that contains the identification of the error that occurred and the workstation operation that caused the error. The information in the data structure is updated for each workstation operation. If you didn't specify the INFDS option, the information is not available to the program.

You define the INFDS data structure on the input format specification form. Make sure the name on the input form matches the name you specify here. Also, on the input specifications form, you must enter a special reserved word (left-justified in columns 44 through 50) that identifies the location of self-defining subfields containing the status information. You must also assign field names (columns 53 through 58) on the input form to each reserved word to reference the subfields. (See 6.3.2.)

- If you entered IND in columns 54 through 59, then enter the number of indicators to be saved and restored for each workstation attached to the file. All indicators from 01 up to and including the indicator you specify are swapped. The number may range from 1 to 99. Indicator swapping isn't done if you didn't specify the IND option or if you specified 1 for NUM.

For the SAVDS and IND options, only the copy of the indicators and data structure for the workstation from which the last input was read is available. The copy that is available changes every time there is an input operation. The current copy in the program is written to the save area for the workstation from which the last input was read. The copy in the save area for the workstation being read is then written to the program area. The workstation save areas are initialized with the values in the program areas at the time the first workstation connects.

When a data structure contains numeric subfields that redefine a result field, the subfields must be initialized to numeric values before you use them in any calculation operations. When you use the SAVDS option, these fields must be initialized when each workstation connects instead of just once during the cycle. One way to do this is to condition the initializing statements with the indicator identifying the initial blank screen.

- If you entered INFSR in columns 54 through 59, then enter the name of a user-written calculation subroutine that receives control when a workstation error occurs during a NEXT, READ, primary input, EXCPT, or a normal cycle output operation. If you didn't specify the INFSR option, RPG II handles error recovery.

Specify the INFSR subroutine on the calculations specifications form. It can perform any function that is normally allowed in a subroutine. Make sure the name in factor 1 of the BEGSR operation on the calculations form matches the name you specify here. You can optionally specify a factor 2 entry in the ENDSR operation that indicates a point to which control is returned. (See 7.3.2.4.4.)

If you are dealing with multikey MIRAM files, enter the starting location of the key field in the data record. This number must be right-justified.

If you are dealing with MIRAM file sharing, enter one of the following file sharing environments (left-justified). ←

■ EXC

Indicates an exclusive environment. This is the default if you didn't specify ACCESS in columns 54 through 59. While you are using the file, your program can read from or write to it but no other program can access it. ←

■ EXCR

Indicates an environment where your program can read from or write to the file but other programs can also read from it. ←

■ SRD

Indicates an environment where your program can read from the file but other programs can read from or write to it. You can only specify this environment for input files. ←

■ SRDO

Indicates a read-only environment. Your program and other programs can read from the file but no program can write to it. You can only specify this environment for input files. ←

■ SADD

Indicates an environment where both your programs and other programs can read from or write to the file. ←

Table 5-7 summarizes the file sharing environment.

*Table 5-7. Summary of File Sharing Environments*

Environment	Your Program	Other Programs
EXC	Read/write	-
EXCR	Read/write	Read
SRD	Read	Read/write
SRDO	Read	Read
SADD	Read/write	Read/write

If you are dealing with MIRAM files containing a record control byte, leave this field blank. ←

#### 5.2.18.4. Key Length (Columns 66 and 67)

You use this field to specify the length of the key field for multikey MIRAM files.

Enter from 1 to 80 for the length of the key field.

#### 5.2.18.5. Duplicate Record (Column 68)

You use this field to specify that duplicate records are allowed for multikey MIRAM files.

If you don't want duplicate records, leave this field blank. Duplicate records will then result in an error.

If you want duplicate records, enter D.

↓  
You can't use duplicate records on the key defined as the primary key for IMS action programs. You must specify duplicate records in the IMS configuration.  
↑

#### 5.2.18.6. Change Keys (Column 69)

You use this field to specify that keys are allowed to change during an update for MIRAM files.

If you don't want to change keys during update, leave this field blank. Then, a change of key will cause error processing.

If you want to change keys during update, enter C.

↓  
You specify the change key for IMS action programs in the configuration, not in the continuation line.

You must specify a primary key for IMS action programs with multikey files by entering a P in column 69. The primary key must be the same as the one defined in the IMS configuration. You can't use duplicates or change keys on the primary key.

↑  
If you want to use another key besides the primary key for sequential processing, enter an R in column 69. You can override this specification for random processing by using the SETK operation.



### **5.2.19. Name of Label Exit or Name of User Device Routine (Columns 54 through 59)**

You use this field to specify the name of your external label processing routine if you have specified E or N in column 53, or to specify the name of your external device handling routine if you specified SPECIAL in columns 40 through 46.

The name you enter in this field can range from one to six alphanumeric or special characters. This entry must be left-justified. The first character must be alphabetic; the rest of the characters may be alphabetic or numeric.

If SPECIAL is specified in columns 40 through 46, this entry must specify the name of your external device handling routine that provides all the required input/output functions for the nonstandard input/output devices.

If E or N is specified in column 53, this entry must specify the name of your external label processing routine that provides all the required label processing functions. In either case, you must include the external routine in your program at link-edit time.

During execution time, the RPG II input/output interface routine will access the external user device handling routine. The function of the interface is to determine the required operation and pass pertinent information to the user routine. The information is supplied through five registers known as linkage registers. The role of each linkage register is as follows:



<u>Register</u>	<u>Contents</u>
-----------------	-----------------

0	Type of operation to be performed:  0 = read a record 4 = write a record 8 = close the file
1	Address of input/output buffer area in the RPG II program
13	Address of the register save area in the RPG II program
14	Address of the location in the calling program (object program) to which control should return after the user routine is executed
15	Address of entry point in your device handling routine. Return code from your device handling routine:  0 = normal 4 = EOF for input or terminate for output Other code = terminate the job

It is the responsibility of your device handling routine to open the device the first time it is called.

#### **5.2.20. Number of Bytes in Main Storage to Be Reserved for Index (Columns 60 through 65)**

You use this field to specify the number of bytes in main storage you want reserved for the top index of an ISAM file. For IRAM or MIRAM files, use the INDEX continuation statement. By using this field, you minimize or eliminate the hardware search of the top index during random retrieval or addition of records to an indexed sequential file, thus increasing your processing speed.

If you do not want to reserve main storage space for the index or if the file is an output, input, or update file to be processed sequentially, leave this field blank.

If you want to reserve main storage space for the index, enter the number of bytes in this field. The minimum number is 256, and the maximum is 32,767. The actual number of bytes used by a given indexed sequential file for the top index is shown in the volume table of contents (VTOC) listing for that file. This entry must be right-justified; leading zeros may be omitted.

For IMS action programs, this field is ignored.

**5.2.21. File Addition/Unordered Load (Column 66)**

You use this field to indicate that new records are being added to an existing sequential output file or indexed input, output, or update file. The new records will be added to the file in their proper key sequence in relation to the records that the file already contains. In the IBM System/3 mode this field is also used to indicate that records in unordered key sequence are to be loaded and an indexed file is to be created with the records in their proper key sequence. The entry in this field is used in conjunction with the entries in the file type field (column 15) and columns 16 through 18 of the output format specifications form. These entries are summarized in Table 5-8. If you are not going to add new records to the file or load records in unordered key sequence (IBM System/3 mode), leave this field blank.

If you are going to add new records to the file, enter an A in this field. You should specify the addition of new records to an existing file only when absolutely necessary, because this function requires more main storage than any other indexed file function.

If you want to load records in unordered key sequence (IBM System/3 mode only) and create an indexed file, enter a U in this field. This field is valid only for IRAM files.

When you use this field, you must leave the cylinder overflow space percentage (X10) field (column 67) blank.

Table 5-8. Summary of File Addition Entries

Entries			Function Performed
File Description Specifications Form		Output Format Specifications Form, Columns 16 Through 18	
(File Type) Column 15	(File Addition) Column 66		
I (input)	Blank	No file addition entry	The existing file is processed without adding new records or updating existing records.
I (input)	A	ADD	New records are added to the existing file without updating the existing records.
O (output)	Blank	No file addition entry	A new file is created or an existing sequential file is extended. The extend option must be specified at job control time.
O (output)	A	ADD	New records are added to the existing indexed file or an existing sequential file is extended. The job control extend option is not used. RPG II sets the extend option.
O (output)	U	No file addition entry	IBM System/3 mode, IRAM file. Load records in unordered key sequence and create indexed file.
U (update)	Blank	No file addition entry	The existing file is processed and updated without adding new records.
U (update)	A	ADD	The existing file is processed, existing records are updated, and new records are added.

Examples 1 and 2 demonstrate the specifications necessary for adding to and deleting from an indexed file.

Example 1:

FILE DESCRIPTION SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	FILE NAME	FILE TYPE		FILE PROCESSING MODE		EXTENSION OR LINE COUNTER CODE	NOT USE U	LABELS	NUMBER OF BYTES IN MAIN STORAGE TO BE RESERVED FOR INDEX	AUTO OR BLANK LENGTH
				FILE DESIGNATION	END OF FILE	KEY OR RECORD ADDRESS FIELD LENGTH	RECORD ADDRESS TYPE					
1	0	F	OUTPUT	0	F	80	80	4	1	DISK	S	A

OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	FILE NAME	SPACE				SKIP				OUTPUT INDICATORS				FILE NAME	*AUTO	DATA FORMAT PBLR	END POSITION IN OUTPUT RECORD	CODES				CONSTANT OR EDIT WORD		
				BEFORE	AFTER	BEFORE	AFTER	AND	AND	AND	AND	NEGATIVE VALUE INDICATION	COMMAS INSERTED	ZERO BALANCE TO PRINT	CODES					AC						
2	0	0	OUTPUT	D	A	D	D																			
3	0	2	0																							
4	0	3	0																							

Line Number      Explanation

- 1                      The A in column 66 indicates that new records will be added to the file.
- 2 through 4        Output specifications needed for the addition. Note columns 16 through 18 on line 2 contain the word ADD.

**NOTE:**

A sequential file may be extended by adding records to the end of the file in a similar manner. In this case, however, there would be no index information in columns 29 through 32 and columns 35 through 38.



**5.2.22. Cylinder Overflow Space Percentage (X10) (Column 67)**

You use this field only when you initially create an indexed file that you intend to add new records to at a later time. It specifies the percentage of each cylinder that is reserved for adding new records to the file.

If you are not creating an indexed file that new records are to be added to, leave this field blank.

If you are creating an indexed file that new records are to be added to, enter 1 through 8 to indicate the percentage, in increments of 10 to each cylinder that is reserved for adding new records. When you use this field, the file addition field (column 66) must be blank.

This field is not used for IRAM files (IBM System/3 mode).

This field is ignored for IMS action programs.





**5.2.23. Number of Extents (Columns 68 and 69)**

You use this field to specify whether the disk file you are creating or processing in the IBM System/3 mode will have all volumes of the file online (all volumes are mounted), or if it will have some of the volumes offline (volumes are mounted one at a time).

If you are creating or processing a single volume file or a multivolume with all volumes online, leave this field blank or enter a 1.

If you are creating or processing a multivolume file with some of the volumes offline, enter a number ranging from 2 through 50. The entry must be right-justified.

The way a file is created governs the way it is processed; that is, a multivolume file created with all volumes online must be processed with all volumes online and a file created with some of the volumes offline must be processed with only one volume online at a time.

Files created or processed by relative record number must have all volumes online. Files processed consecutively, indexed sequentially, or randomly by key may be created with all volumes online or with one volume online at a time.

This field is ignored for IMS action programs.

**5.2.24. Tape Rewind Option (Column 70)**

You use this field to specify whether a magnetic tape file is to be rewound at initialization or termination time. The entries for this field are summarized in Table 5-9.

*Table 5-9. Summary of Tape Rewind Options*

Entry	Activity	
	Initialization Time	Termination Time
R or blank	Rewind without lock	Rewind without lock
N	No rewind	No rewind
U	Rewind with lock (Operator intervention will be required before the file on the tape drive can be accessed again.)	Rewind with lock (Operator intervention will be required before the file on the tape drive can be accessed again.)

This field is ignored for IMS action programs.

### 5.2.25. File Conditioners (Columns 71 and 72)

You use this field to specify an external indicator that controls the opening of an input, output, update, or combined file. When you specify an external indicator for a file, the indicator must be set on prior to program execution by a //△SET△UPSI job control statement. If this is not done, the file cannot be opened. If you do not want to control the opening of the file with an external indicator, leave this field blank.

If you do want to control the opening of the file, enter U1 through U8 in this field. We will cover the use of these external indicators when we discuss how you use indicators in your program.

This field is ignored for IMS action programs.

### 5.3. EXAMPLES OF ENTRIES ON THE FILE DESCRIPTION SPECIFICATIONS FORM

Figure 5-2 shows you examples of typical entries on the file description specifications form.

Explanation of entries in Figure 5-2:

#### Line Number    Explanation

→ 010        The file named INPUT is the primary input file (I in column 15 and P in column 16). It is in fixed format (F in column 19). Each record is 80 characters in length (80 in columns 24 through 27). The matching fields in the file are in ascending sequence (A in column 18). All records in the file must be processed (E in column 17). Dual input/output areas are used in processing this file (2 in column 32). The file is referenced on the file extension specifications form as a chaining file (E in column 39). The file is read from a card reader (READER in columns 40 through 46).

020        The file named MASTER is a chained input file (I in column 15 and C in column 16). It is a fixed-blocked format (F in column 19 and 160 in columns 20 through 23). Each record is 80 characters in length (80 in columns 24 through 27). The records are retrieved by alphanumeric keys (A in column 31). The file is an indexed sequential file (I in column 32). The key field in the record is 8 characters in length and begins in position 1 of the record (8 in columns 29 and 30 and 1 in columns 35 through 38). The file contains standard labels (S in column 53). The file is read from a disk (DISC in columns 40 through 46).

→ 030        The file named RAF is an input record address file (I in column 15 and R in column 16). It is in fixed format (F in column 19). Each record is 80 characters in length (80 in columns 24 through 27). The length of each record address field is 8 characters (8 in columns 29 and 30). The file is further described on the file extension specifications form (E in column 39). The file is read from a card reader (READER in columns 40 through 46).



Line Number Explanation

- 040 The file named UPDATE is an update file that is used for input and is updated after each record is processed (U in column 15). It is a secondary file (S in column 16). The file is in fixed-blocked format (F in column 19 and 160 in columns 20 through 23). Each record is 40 characters in length (40 in columns 24 through 27). The file will be processed randomly via a record address file (R in column 28). The key field is 8 characters in length and begins in position 1 of the record (8 in columns 29 and 30 and 1 in columns 35 through 38). The keys are alphanumeric (A in column 31). The file is an indexed sequential file (I in column 32). New records may be added within the file or at the end of the file (A in column 66). One thousand bytes are reserved in main storage for the indexed sequential cylinder index (1000 in columns 60 through 65). The file is contained on a disk (DISC in columns 40 through 46).
- 050 The file named TABLE is an input table or array file (I in column 15 and T in column 16). It is in fixed format (F in column 19). Each record is 40 characters in length (40 in columns 24 through 27). The file is further described on the file extension specifications form (E in column 39). The file is read from a card reader (READER in columns 40 through 46).
- 060 The file named OUTBLE is an output file (O in column 15). The file is in fixed-blocked format (F in column 19 and 100 in columns 20 through 23). Each record is 50 characters in length (50 in columns 24 through 27). Standard labels are used (S in column 53). The file is to be written on disk (DISC in columns 40 through 46).
- 070 The file named LINECNT is an output file (O in column 15). The file is in variable format (V in column 19). The maximum record is 60 characters in length (60 in columns 24 through 27). The file is further described on the line counter specifications form (L in column 39). Standard labels are used (S in column 53). The file is written on tape (TAPE in columns 40 through 46). The tape is not rewound at either initialization or termination (N in column 70).
- 080 The file named COMB is a combined file (C in column 15). It is a secondary input file (S in column 16). The file is in fixed-unblocked format (F in column 19 and columns 20 through 23 are left blank). Each record is 80 characters in length (80 in columns 24 through 27). As a combined file, information is read from the input cards and, during processing, additional information is punched onto the input cards. As a result, a card-read-punch device is required (CRP in columns 40 through 46).
- 090 The file named PRINT is an output file (O in column 15). The file is in fixed-unblocked format (F in column 19 and columns 20 through 23 are left blank). Each record is 120 characters in length (120 in columns 24 through 27). OF is used as the overflow indicator for this file (OF in columns 33 and 34). The file is written on printer (PRINTER in columns 40 through 46).

Line Number Explanation

- 100 The file named DEVICE is a secondary input file (I in column 15 and S in column 16). The file is in fixed-unblocked format (F in column 19 and columns 20 through 23 are left blank). The file is read from a nonstandard peripheral device for which the user has supplied an input/output program (SPECIAL in columns 40 through 46). The user-supplied input/output program is named IOCTRL (IOCTRL in columns 54 through 59). The file is used only when the external indicator U1 is on (U1 in columns 71 and 72).
- 110 The file named DISPLAY is a display file (D in column 15). The file is in fixed-unblocked format (F in column 19 and columns 20 through 23 are left blank). Each record is 40 characters in length (40 in columns 24 through 27). The file is displayed on a console device (CONSOLE in columns 40 through 46).
- 120 The file named TAPE is the primary input file (I in column 15 and P in column 16). The file is in variable-unblocked format (V in column 19 and 200 in columns 20 through 23). The maximum record size is 200 characters in length (200 in columns 24 through 27). Dual input/output areas are used in processing this file (2 in column 32). The file is read from a tape (TAPE in columns 40 through 46). The tape is rewound at both initialization and termination (U in column 70).
- 130 The file named DEMAND is an input demand file (I in column 15 and D in column 16). The file is in fixed format (F in column 19). Each record is 80 characters in length (80 in columns 24 through 27). The file is read from a card reader (READER in columns 40 through 46). ←
- 140 The file named DIRECT is a secondary input file (I in column 15 and S in column 16). The file is in fixed-unblocked format (F in column 19 and columns 20 through 23 are left blank). Each record is 80 characters in length (80 in columns 24 through 27). The file is a direct file (D in column 32). The file is processed randomly (R in column 28) and the records are retrieved by relative record numbers (R in column 31). The file is read from a disk (DISC in columns 40 through 46).
- 150 The file named TAG is a tag file (I in column 15 and R in column 16). The file is in fixed-blocked format (F in column 19 and 80 in columns 20 through 23). Each record is 10 characters in length (10 in columns 24 through 27). The length of the key field is 10 characters (10 in columns 29 and 30). The file is a tag (ADDROUT) file (T in column 32). The file is further defined on the file extension specifications form (E in column 39). The file is read from a disk (DISC in columns 40 through 46).



## 6. Input Format Specifications Form

### 6.1. GENERAL DESCRIPTION

You must have an input format specifications form (Figure 6-1) in all programs. This form describes the data records in each input file.

As you can see, the form is divided into two areas: record identification and field description.

You use the record identification area to:

- Identify the input files
- Identify the different types of records in each file
- Indicate if a specific sequence of data records is required in a file
- Indicate if one or more of a given type of record can appear in a file
- Indicate if a specific type of record must be present in a file

After you have made the record identification entries for a file, you then use the field description area to describe the individual fields in each type of data record. When you describe a field, you specify the data format of the field, the location and length, the number of decimal positions if the field is a numeric field, the field name, whether the field is a control field or a matching or chaining field, when the field is used by the program, and the indicators that are on if the contents of a numeric field meet certain conditions.

### 6.2. RECORD IDENTIFICATION ENTRIES (COLUMNS 7 THROUGH 42)

In the following subsections you'll see what each record identification field on the form is used for and how the entries in these fields affect your program.

**NOTE:**

*When using a workstation, specify a blank record as the first record on the input format specifications form.*





### 6.2.1. File Name (Columns 7 through 13)

You use this field to identify the input file that contains the data records you want to describe. The file that is being described can only be an input, update, or combined file. A file name must be entered in this field on the first record identification line for each file that you describe. The file name must be the same as the name that you specified for this file in columns 7 through 13 on the file description specifications form.

### 6.2.2. Sequence (Columns 15 and 16)

You use this field to indicate whether or not a specific sequence of data records is required within a given file. An entry is required in this field on the first specification line for each record type you describe for a file. The entries in this field are used in conjunction with the number field (column 17) and the optional field (column 18). These entries are summarized in Table 6-1.

If a record type does not have to be in any specific sequence in relation to the rest of the record types in the file, enter AA through ZZ in this field.

If a record type must be in a specific sequence in relation to the rest of the record types in the file, enter 01 through 99 in this field. If more than one record type must be in a specific sequence, you must enter 01 in this field for the first record type in the sequence. You can use any other 2-digit combination for other record types in the sequence; however, these entries must be in ascending order.

If you use more than one record identification entry for a record type (AND or OR lines indicated by AND or OR in columns 14 through 16), you must place the sequence entry on the first line of the record identification group.

If you use alphabetic (AA through ZZ) or numeric (01 through 99) entries for different record types in the same file, the record types with alphabetic entries must precede the record types with numeric entries on the form.

When a record is out of sequence, the H0 indicator is set on, and the error code C (hexadecimal C3) is placed in the \*ERROR field (a field that RPG II stores error information in). The H0 indicator remains on until the completion of the current operating cycle. If it is not set off before the next cycle begins, the program goes through end-of-job processing and terminates. If you don't want your program to terminate when this type of error occurs, you can prevent it by including a routine on the calculation specifications form that tests the \*ERROR field for error condition C when the H0 indicator is set on and sets the H0 indicator off when the error code is present.

### 6.2.3. Number (Column 17)

You use this field to specify whether one or more records of a given type may appear in a file. The entries in this field are used in conjunction with the entries in the sequence field (columns 15 and 16) and the optional field (column 18). These entries are summarized in Table 6-1.

If the entry in the sequence field is alphabetic, leave this field blank.

If the entry in the sequence field is numeric, enter a 1 if only one record of this type may appear in the file before another record type is encountered or enter an N if one or more records of this type may appear together in the file.

#### 6.2.4. Optional (Column 18)

You use this field to specify whether a specific type of record must be present in the file. The entries in this field are used in conjunction with the entries in the sequence field (columns 15 and 16) and the number field (column 17). These entries are summarized in Table 6-1.

If this record type must be present in the file or the entry in the sequence field is alphabetic, leave this field blank.

If this record type may or may not be present in the file, enter an O in this field.

Table 6-1. Summary of Sequence, Number, and Optional Field Entries

Entries			Interpretation
Sequence (Columns 15-16)	Number (Column 17)	Optional (Column 18)	
AA-ZZ	Blank	Blank	The record type does not have to be in any specific sequence in relation to the rest of the record types in the file.
01-99	1	Blank	Only one record of this type may appear in the file before another type of record is encountered.
01-99	1	O	This type of record is optional; it may or may not appear in the file. If it does appear, only one record of this type may appear in the file before another type of record is encountered.
01-99	N	Blank	This type of record must be present in the file and one or more records of this type may appear together in the file.
01-99	N	O	This type of record is optional, it may or may not appear in the file. If this type of record appears in the file one or more records of this type may appear together in the file.

#### 6.2.5. Record Identifying Indicator (Columns 19 and 20)

You use this field to associate an indicator with the record type defined on this line. Each record type that you define requires an entry in this field. A record identifying indicator is not always required in IBM System/3 mode. It is only necessary if you are concerned about different record types. This field is used in conjunction with the record identification codes (columns 21 through 41). When a record is selected for processing that satisfies the identification requirements for a particular record type, the record identifying indicator associated with the record type is set on. This indicator can be used to condition when calculation and output operations are performed in your program. If a record identifying

indicator is set on, it remains on until you set it off or until the end of the cycle, when RPG II sets it off. You can specify a general indicator (O1 through 99), a halt indicator (H1 through H9), or a control level indicator (L1 through L9 or LR) as the record identifying indicator for a record type. The general indicators (O1 through 99) are the ones that are generally used in this field. The setting on of one does not affect the setting of any of the others. These indicators can appear in any order; however, you must specify a unique indicator for each record for each record type. Except for chaining and chained files, only one general indicator is on at one time.

When processing for a workstation terminal file, each screen format or record, including the first screen format which is always a blank dummy, should be identified on the input specification.

You can also use the halt indicators (H1 through H9) or the control level indicators (L1 through L9, or LR). We do not recommend that you use these indicators as record identifying indicators because, unless you use extreme care, the results are unpredictable. When using data structures, enter DS in this field.

When you use input format specifications for data structures, they must follow all other input format specifications. ←

#### **6.2.5.1. Look-Ahead Feature**

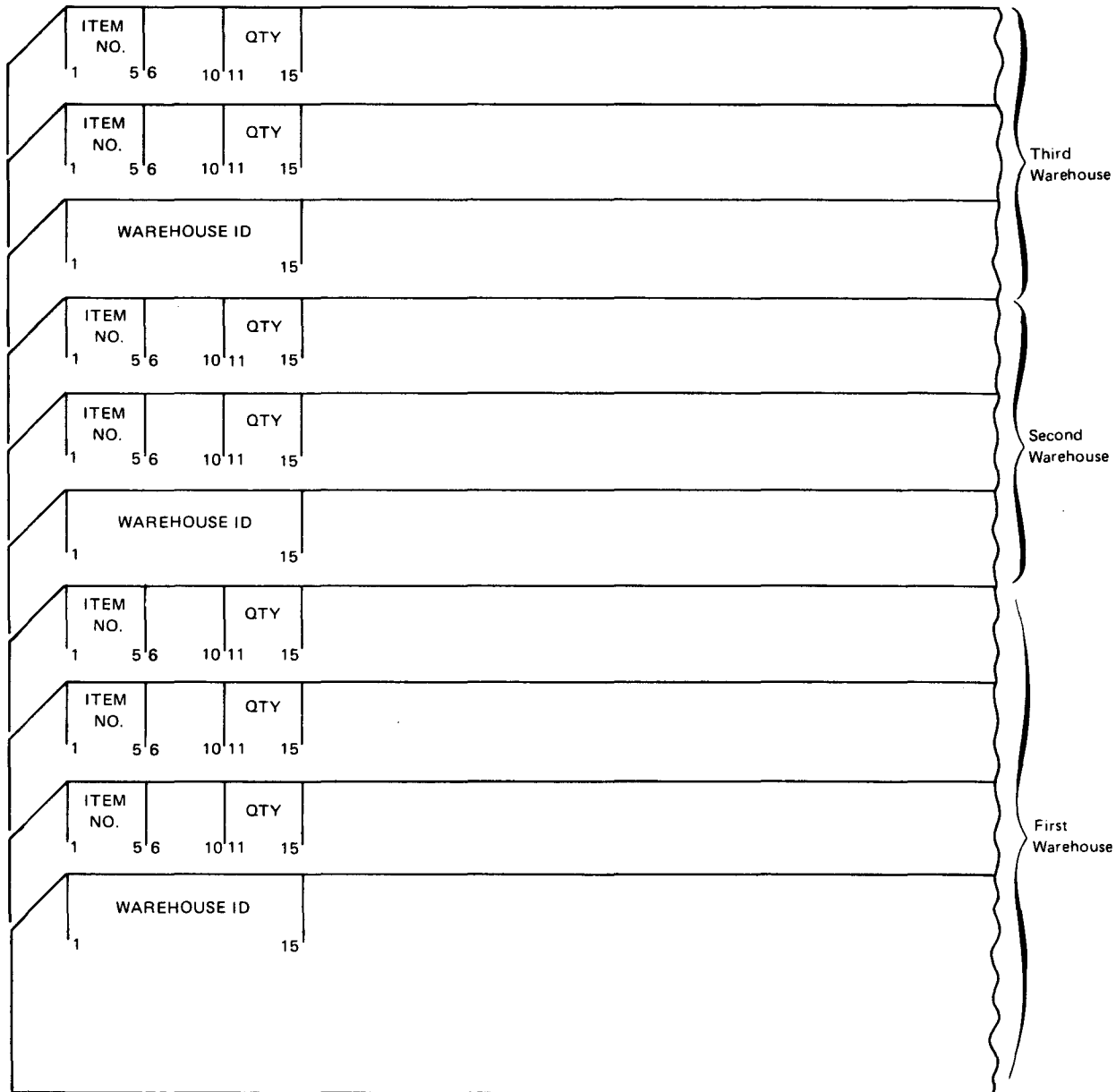
If you want to use the RPG II look-ahead feature, enter \*\* in this field to indicate that the associated data fields are look-ahead fields. The look-ahead feature allows you to look at information in the next record available for processing. You can look at the file that is currently selected for processing, as well as other files that are present but not selected for processing during this cycle. This feature is usually used with the FORCE operation to override normal file selection in multifile processing. The entry \*\* cannot be specified for AND or OR lines, chained files, demand files, files that require sequence checking (a numeric entry in columns 15 and 16), or workstation files.

#### **6.2.5.2. Spread Card Feature**

If you want to use the RPG II spread card feature, enter TR in this field to indicate that the associated fields are contained on spread cards. The spread card feature allows you to reduce the number of input cards, used with an input or combined file on a card device, by placing more than one detail record on a single punched card called a spread card. A spread card consists of a header portion followed by as many trailer portions (detail records) as are desired (within the space of a single card). Each trailer portion can contain several fields, but all trailer portions must contain the same fields in the same order. A trailer portion cannot be split between two spread cards.

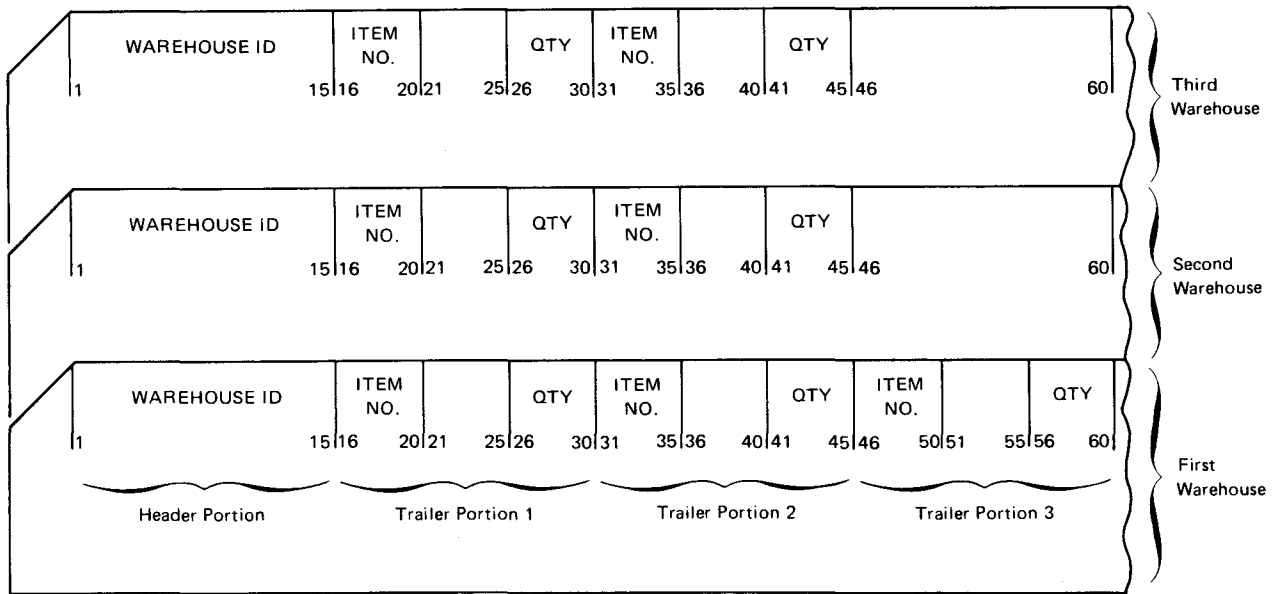
This feature may not be used in IMS action programs.

Figure 6-2 illustrates how the number of input cards can be reduced by using spread cards. As you can see, the original file consists of warehouse identification records followed by individual transaction records. The spread card file shows how the number cards can be reduced by using the warehouse identification records as header portions and the transaction records as the trailer portions.



a. Original file

Figure 6-2. Spread Card Example (Part 1 of 2)



b. Spread card file

Figure 6-2. Spread Card Example (Part 2 of 2)

During processing, the header portion of a spread card and one trailer portion are processed as one logical record in each program cycle; that is, the header portion and trailer portion are processed during one program cycle, the header portion and trailer portion 2 are processed in the next cycle, and so on.

When you use spread card records, the record identification codes must be in the header portion and N must be specified in column 17 (on the first record identification line) if sequence checking is specified (a numeric entry in columns 15 and 16). The header portion fields must be described first. These fields may be specified as control level fields (L1 through L9), input chaining fields (C1 through C9), or matching fields (M1 through M9). After you have described the header portion fields, you describe the trailer portion fields. You do this by first entering TR in columns 19 and 20 and the line immediately following the last line of the header portion field descriptions and then specifying the trailer portion fields on the succeeding lines. The first and last positions of the trailer portion must be defined and referenced. The trailer portion fields may be specified as input chaining fields (C1 through C9) but may not be specified as control level or matching fields. Since all trailer portion fields are the same and are the same order, only one set of field descriptions is required. Figure 6-3 shows an example of how to specify spread cards.



The groups of record identification code fields do not have to be used in sequence. For example, entries can be made in columns 35 through 38 without making entries in columns 21 through 27 or 28 through 34. If more than one group is used, the data record is checked for the identifying codes in the order that you specify them. Checking ends when conditions are met that set on a record identifying indicator. If the conditions are not met for any of the record types you have specified, the HO indicator is set on and the error code A (hexadecimal C1) is placed in the \*ERROR field. The HO indicator remains on until the completion of the current operating cycle. If it is not set off before the next cycle begins, the program goes through end-of-job processing and terminates. This type of error can be provided for in one of two ways. The first way consists of specifying a record identifying indicator for undefined record types on the input format specifications form immediately after the last defined record type. This allows the identification code checking process to end normally when an undefined record type is encountered. If the record does not meet the requirements for any of the defined record types, it will meet the requirements for undefined record types. Figure 6-4 shows an example of how you provide for undefined record types on the input format specifications form.

Three record types are defined in Figure 6-4 on lines 010 through 050. If a record is read that meets the requirements on any of these lines, the associated record identifying indicator is set on. If a record is read that does not meet the requirements for the defined record types on lines 010 through 050, it is an undefined record type, and it will set on record identifying indicator 04.

The second way you can provide for undefined record types is to include a routine on the calculation specifications form that tests the \*ERROR field for error condition A when the HO indicator is set on and, if present, it sets off the HO indicator.



RPG II

INPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_

PAGE NO	FORM TYPE	LINE NO	FILE NAME	RECORD IDENTIFICATION																		FIELD LOC												
				SEQUENCE 01 99 OR AA ZZ		NUMBER	RECORD IDENTIFYING INDICATOR OR ..	RECORD IDENTIFICATION CODES									STACKER SELECT	DATA FORMAT P B L R	FROM															
				A	N			1 OR N	O OPTIONAL	1			2			3																		
				O	R	POSITION	N NOT			C Z D	CHARACTER	POSITION	N NOT	C Z D	CHARACTER	POSITION	N NOT	C Z D	CHARACTER															
1	2	3	5	6	14	15	16	17	18	19	20	21	24	25	26	27	28	31	32	33	34	35	38	39	40	41	42	43	44	47	48			
	0,1	0	MASCUST	AA		01		5	2	Z	D																							
	0,2	0		BB		02		6	6	D	9		4	4	N	C	5						7	6	N	Z	A							
	0,3	0		AND				5	3	C	U																							
	0,4	0		CC		03		2	0	Z	A																							
	0,5	0		OR				1	2	C	4																							
	0,6	0		DD		04																												

Figure 6-4. Providing for Undefined Record Types

**6.2.6.1. Position (Columns 21 through 24, 28 through 31, and 35 through 38)**

You use this field to specify the position in the data record that contains the record identification code. For card files, you enter the card column that contains the record identification code. For tape and disk files you enter the character position (byte). This entry can range from 0001 through 9999. The entry must be right-justified; leading zeros may be omitted.

**6.2.6.2. N=NOT (Columns 25, 32, and 39)**

You use this field to specify whether or not the record identification code defined by the C/Z/D and character fields (columns 26 and 27, 33 and 34, or 40 and 41) must be present in order to set the record identifying indicator on.

If the record identification code must be present, leave this field blank.

If the record identification code must not be present, enter an N in this field.

**6.2.6.3. C/Z/D - Characters, Zone, Digit (Columns 26, 33, and 40)**

You use this field to specify what part of the character in the character field (columns 27, 34, or 41) is checked for a record identification code. You can specify that you want the entire character checked, the zone portion only, or the digit portion only.

If you want the entire character checked, enter a C in this field. If you make this entry, the entire eight bits of the character position are checked. Only the presence of the specified character will satisfy the condition.

If you want to check the zone portion of the character (the leftmost four bits), enter a Z in this field. Any character that has the same zone portion as the specified character will satisfy the condition except in those cases where &, - (minus), or blank is the specified character. In these cases, RPG II interprets their presence as shown in Table 6-2. The interpretation in Table 6-2 is made to provide compatibility with Hollerith code.

Table 6-2. Interpretation of &, —, and Blank Characters

Entry		Character in Specified Position in Data Record that Satisfies Condition
C/Z/D (Column 26)	Character (Column 27)	
Z	&	A-I, &, or any character with an $1100_2$ (X'C') zone
Z	— (minus)	J-R, — (minus), or any character with an $1101_2$ (X'D') zone
Z	Blank	0-9, blank, or any character with an $1111_2$ (X'F') zone



If you want to check the digit portion of the character, enter a D in this field. If you make this entry, only the digit portion of the character position (the rightmost four bits) is checked. Any character that has the same digit portion as the specified character will satisfy the condition. For example, if B is the specified character, B, K, S, 2, or any character with a  $0010_2$  (binary 2) digit portion will satisfy the condition. ←

#### **6.2.6.4. Character (Columns 27, 34, and 41)**

You use this field to specify the character you want compared to the character contained in a particular position in an input record. You can enter any EBCDIC character, including a blank. The comparison depends upon the entry you made in the C/Z/D field (columns 26, 33, and 40).

If you specify more than one record type for a file, the record identification codes for each record type should be different so that it will be impossible for an input record to satisfy the condition for more than one record type.

#### **6.2.7. Stacker Select (Column 42)**

You use this field to specify the stacker that you want the cards placed in when you use a card device that has more than one stacker.

The 0604 card punch subsystem is the only subsystem that you can specify a stacker selection for in your program.

If you are using the 0604 card punch subsystem and you want the cards placed in the normal stacker, leave this field blank. If you want the cards placed in the alternate stacker, enter 2 through n in this field.

If you are using an 0605 card punch subsystem, or 0716 card reader subsystem, or an 0717 card reader subsystem, you must leave this field blank because you cannot specify stacker selection for these subsystems.

If you use stacker selection and you have AND lines for a record type, the stacker selection entry need only be made once. If you have OR lines for a record type, you must make the stacker selection entry on each line in the OR relationship.

This field is ignored for IMS action programs.

### **6.2.8. AND/OR Relationship (Columns 14 through 16)**

You use this field to establish an AND or an OR relationship. You use an AND relationship when more than three record identification codes are required to identify a record type and an OR relationship when you need to identify two or more record types that have common data fields.

You establish an AND relationship by placing the first three record identification codes for the record type on the first line and the additional codes on succeeding lines, with an AND entered in columns 14 through 16 on each additional line that is used. The number field (column 17), optional field (column 18), record identifying indicator (columns 19 and 20), and the stacker select field (column 42) must be left blank on AND lines. You establish an OR relationship by placing OR in columns 14 and 15. The record identifying indicator for the additional record type that has common data fields is placed in columns 19 and 20, and the record identification codes in columns 21 through 41 on the next line after the first group of record identification lines for the common data fields. If additional record types must be defined for the common data fields, these can be placed on succeeding OR lines. If you do not specify a record identifying indicator or an OR line and the conditions on this line are satisfied, the record identifying indicator on the line immediately above is set on. The sequence field (column 16), number field (column 17), and the optional field (column 18) must be left blank on OR lines. The stacker select field (column 42), however, may be used on OR lines.

### **6.3. FIELD DESCRIPTION ENTRIES (COLUMNS 43 THROUGH 70)**

In the following subsections you'll see what each field description field on the form is used for and how the entries in these fields affect your program. The field description entries for a given record type are always written on the lines immediately following the record identification lines. For example, if lines 010, 020, and 030 contain the record identification entries for a given record type, you would write the first field description entry for this record type on line 040, the second on line 050, and so on. Columns 7 through 42 on a field description line must be left blank.

#### **6.3.1. Data Format (Column 43)**

You use this field to specify the data format of the input record field you are describing on this line. An input record field can be in alphanumeric format, unpacked numeric format, binary format, or packed decimal format, or it can be a numeric field that is preceded or followed by a plus or minus sign.

If the input record field is in alphanumeric or unpacked numeric format, leave this field blank.

If the input record field is in binary format, enter a B in this field.

If the input record field is in packed decimal format, enter a P in this field.

If the input record field is a numeric field preceded by a plus or minus sign, enter an L in this field.

If the input record field is a numeric field followed by a plus or minus sign, enter an R in this field.

### 6.3.2. Field Location (Columns 44 through 51)

You use this field to specify the location and length of the input record field you are describing on this line. The starting position is specified in the FROM area and the ending position in the TO area. For workstation terminal fields, these values are the locations on the input record, not on the screen. The length of the field is the number of character positions (bytes) the field occupies within the overall area. The maximum field length for numeric fields is 8 bytes if the field is packed and 15 bytes if it is unpacked. The maximum length for alphanumeric fields is 256 bytes. These entries must be right-justified; leading zeros may be omitted.

If L or R is specified in the data format field (column 43), the entries in the FROM and TO areas must allow for the preceding or following sign. This means that if the data in a numeric field occupies five character positions and L or R is specified, the field actually occupies six character positions – five positions for data and one for the sign. For example, assume that a numeric field occupies character positions 9 through 13 and L is specified. You would have to specify that this field occupies character positions 8 through 13 to allow for the preceding sign. Similarly, if you specified R for the same field, you would specify that this field occupies character positions 9 through 14 to allow for the following sign.

If B (binary) is specified in data format field, the entries in the FROM and TO areas can only indicate a 2- or 4-character position field because a binary format field must be either two or four bytes in length. The maximum decimal value allowed in a 2-byte field is 9999, and the maximum for a 4-byte field is 99999999. If the decimal value in the binary field will not exceed 9999, specify a 2-byte field. If the maximum value will range between 10000 and 99999999, specify a 4-byte field.

If you are dealing with workstation files and entered INFDS in columns 54 through 59 of the file description specifications form, you specify one of the following words that identify the location of self-defining subfields. (These words must be left-justified in columns 44 through 50.)

#### ■ \*STATUS

Identifies the 5-digit numeric subfield containing a code that identifies the error condition. The format of the field is 012nn, where nn is the error code from screen format services and corresponds to the screen format error message number (SFnn; see UP-8076).

#### ■ \*OPCODE

Identifies the 5-character alphanumeric subfield containing a value that indicates which workstation operation was executing when the error occurred. The value READ, NEXT, or WRITE is placed in \*OPCODE.

- **\*RECORD**

Identifies the 8-character alphanumeric subfield containing the format name when the \*OPCODE subfield contains the value WRITE or containing blanks when the \*OPCODE subfield doesn't contain the value WRITE.

- **\*SIZE**

Identifies the 4-digit numeric subfield containing the character size of the display screen (1920 or 0960).



- **\*MODE**

Identifies a 2-digit numeric field that always contains the value zero. This word is provided so that you can compile an IBM System/34 program without getting an error.

- **\*INP**

Identifies a 2-digit numeric field that always contains the value zero. This word is provided so that you can compile an IBM System/34 program without getting an error.

- **\*OUT**

Identifies a 2-digit numeric field that always contains the value zero. This word is provided so that you can compile an IBM System/34 program without getting an error.



### 6.3.3. Decimal Positions (Column 52)

You use this field to specify the number of decimal positions in a numeric field or a numeric array element.

If the field is an alphanumeric field, leave this field blank.

If the field is a numeric field or a numeric array element, enter the number of decimal positions in the field (0 through 9). The number of decimal positions cannot exceed the length of the field.

#### 6.3.4. Field Name (Columns 53 through 58)

You use this field to specify the name of the field, array, or array element you are describing on this line. Once the field, array, or array element is named, you can reference it throughout your program by using that name. This entry must be left-justified.

If you specify a field or array name, enter the name in this field in the form of one to six alphanumeric characters. The first character of the name must be alphabetic.

If you are specifying an array element name, enter the name in this field in the form:

array-name, index

The index is the number of the element in the array. If you specify an array name or array element, the control level field (columns 59 and 60), the matching or chaining field (columns 61 and 62), and the field record relation field (columns 63 and 64) must be blank. If you specify an array name, the field indicators (columns 65 through 70) must be blank.

You can also use the RPG II special field names PAGE, PAGE1 through PAGE7 and \*ERROR in this field. PAGE and PAGE1 through PAGE7 are used to control automatic sequential page numbering on a printed report. An \*ERROR is used for program testing (debugging). We will cover the use of these field names when we discuss printing techniques and program testing.

If you are dealing with workstation files and entered INFDS in columns 54 through 59 of the file description specifications form, you must assign field names in columns 53 through 58 that reference the subfields you defined in columns 44 through 50.



### 6.3.5. Control Level (Columns 59 and 60)

You use this field to specify that the associated input record field is a control field. When an input record field is specified as a control field, its contents are compared with the contents of the control field of the previously processed record with the same control (L) level. If the contents of the control fields are not equal, a control break occurs (normal processing is interrupted), the specified control level indicator is set on, and the program performs the total time processing conditioned by the associated control level indicator. If the associated input record field is not a control field, or it is a binary or look-ahead field, or a chained or demand file is involved, leave this field blank.

Control level indicators are prohibited when using workstation terminal files. If the associated input record field is a control field, enter a control level indicator (L1 through L9) in this field.

You can specify control level indicators in any sequence; that is, gaps may be left in the specification sequence. For example, if you need four control level indicators in your program, you could specify L1, L2, L3, and L4 or you could specify L1, L4, L6, and L8.

If you specify more than one control level indicator in your program, the setting on of one control level indicator causes all lower control level indicators to be set on. For example, if the L6 control level indicator is set on, the L1 through L5 control level indicators are also set on. In addition, when the last record (LR) indicator is set on, all control level indicators (L1 through L9) are set on.

RPG II also provides you with the ability to have two or more fields in an input record considered as one control field. You can do this by specifying the same control level indicator for each field that is to make up the composite control field. These fields do not have to occupy contiguous positions in the input record.

When split control fields are specified, the first field listed on the form is placed in the high order position of the composite control field, the second field is placed immediately adjacent to the first field, the third field is placed immediately adjacent to the second field, and so on. For example, assume that three field names NAME, RANK, and SERNO are to be considered as one control field. If NAME is listed first on the form, RANK second, and SERNO third, the composite control field that is created is as follows:

NAME	RANK	SERNO
------	------	-------

If RANK is listed first, SERNO second, and NAME third, the composite control field that is created is as follows:

RANK	SERNO	NAME
------	-------	------

There are a number of things to take into consideration when you use control fields. These are:

- Control levels may be written in any order and the control levels need not be successively assigned; e.g., L5 may appear before L2.

- A control break is most likely to occur after the first record containing a control field is read. This control field is compared to a storage area that contains no data. Total calculations and total output operations are bypassed for the first record containing a control field because two control fields are not being compared.
- Control fields are initially set to zero or to the lowest alternate collating sequence value specified.
- If the same control level indicator is used in different files and record types, both the associated control fields must be the same alphabetic or numeric type and the same length.
- In the same record type, record columns in control fields may overlap, even though assigned different control level indicators. The total number of columns assigned as control fields must not be greater than 144, counting each control level only once.
- Because field names are ignored in control level operations, fields from different record types, assigned the same control level indicator, may have the same name.
- When numeric control fields are compared to see if a control break has occurred:
  - They are compared as if there were no decimal position; e.g., 3.95 is the same as 395.
  - Only the digit portion is compared and is considered to be positive; e.g., plus one is the same as minus one.
  - Only the digit portion of each character in numeric control fields is compared to see if the information has changed. This occurs when any one of several control fields with the same control level indicator is described as numeric.
- Unwanted control breaks may occur if different record types in a file do not have the same number of control fields. See 12.2.2 for more information.

Split control fields also require some special considerations.

- No other specifications may come between the lines that describe split control fields.
- If one portion of a split control field is numeric, the whole field is considered numeric.
- A split control field may be made up of either a packed decimal field or an unpacked decimal field. It cannot be made up of both.
- If the control field names are different:
  - For one control level indicator, a control field may be split in some record types and not in others. However, the length of the control field must be the same in all record types.
  - The length of the portions of a split control field may vary for different record types. However, the total length of the portions must be the same.



- A numeric split control field may have more than 15 characters, if any one portion does not exceed 15 characters and the sum of all control fields is not greater than 144 characters (counting each control level only once).

### 6.3.6. Matching Fields or Chaining Fields (Columns 61 and 62)

You use this field to specify which input record fields are used as matching or chaining fields when you process files in your program using matching records or input chaining.

If the input record field is not a matching or chaining field, leave this field blank. If the input record field is a matching field, enter a matching field indicator (M1 through M9) in this field.

If the input record field is a chaining field, enter a chaining field indicator (C1 through C9) in this field.

Matching fields are prohibited when using workstation terminal files.

#### 6.3.6.1. Matching Fields

Matching fields cannot be specified for arrays, chained files, binary fields, or look-ahead fields. The matching field indicators do not condition operations, and they are not affected by nor do they affect control level operations. They are used only to specify the fields that are used for matching records.

All record types that you specify matching fields for must be in the same sequence; that is, the sequence specified in the sequence field (column 18) of the file description specifications form must be the same for all record types using matching fields.

The same series of matching field indicators must be specified for all record types. If the matching fields in the first record type are specified by M2, M3, and M4, every record type with matching fields must use this series.

The length of a matching field for a given matching field indicator must be the same for all record types. If the length of the matching field specified by M1 in the first record type is 10, the length of the matching field specified by M1 in every record must be 10.

Matching fields need not be specified in order. If M1, M2, and M3 are used to specify matching fields, the line with M3 could appear between the lines containing M1 and M2. When more than one matching field indicator is specified, the record fields that the indicators are assigned to are considered as one composite field in which the field with the highest matching field indicator occupies the high order position of the composite field and the one with the lowest matching field indicator occupies the low order position.

When any matching field indicator in a file record (M1 through M9) is specified with field record relation, all matching field indicators in that record must be specified once without field record relation. If all matching field indicators are not common to all record types in an OR relationship, dummy matching field indicators should be used to satisfy the above condition. These dummy matching fields are ignored during processing.

If all matching fields in a record are specified with field record relation, each field record relation indicator must have a complete set of matching fields associated with it.

If you specify matching fields for a single sequential file (input, combined, or update), sequence checking is performed on the matching fields in each record in the file. When a record is out of sequence, the HO indicator is set on, and the error code B (hexadecimal C2) is placed in the \*ERROR field. The HO indicator remains on until the completion of the current operating cycle. If it is not set off before the next cycle begins, the program goes through end-of-job processing and terminates. This type of error can be provided for by including a routine on the calculation specifications form that tests the \*ERROR field for error code B when the HO indicator is set on; if present, it sets the HO indicator off.

When you use matching fields to match records from the primary file with those from secondary files (matching records technique), the MR (matching record) indicator is set on when the matching fields in the record in the primary file completely match those in a record from a secondary file. The MR indicator remains on during the detail time processing of a matching record and through the total time processing that follows the record. The MR indicator can be used to condition calculation and output operations.

When a record from a primary file completely matches records in more than one secondary file, the record from the primary file is processed first and the records from the secondary files are processed in the order that these files are specified on the file description form.

### 6.3.6.2. Chaining Fields

With RPG II, there are two methods of chaining:

- Input chaining by using the C1 through C9 chaining indicators on the input format specifications form.
- Using the CHAIN operation on the calculation specifications form.

Of the two methods, the CHAIN operation is preferred since it is more efficient. Since we are dealing with the input format specifications form, our discussion here will be limited to input chaining using the C1 through C9 chaining field indicators.

A maximum of nine chaining fields may be specified for a record in the chaining file. You can specify the chaining field by entering a chaining field indicator (C1 through C9) in columns 61 and 62 on the line where the field is defined. If you specify a chaining field, you must also enter the chaining field indicator in columns 9 and 10 on the file extension specifications form on the same line you specify both the name of the chaining file and the chained file. Arrays, binary fields, look-ahead fields, and numeric fields with preceding or following signs cannot be used as chaining fields.

You can also specify split chaining fields (either contiguous or noncontiguous fields) by using the same chaining field indicator on each line where the fields that make up the split chaining field are described. For example, assume that you have three fields named CNMBR, CNAME, and ADDR (located in record positions 7 through 12, 19 through 48, and 55 through 80 respectively) that you want to specify as a split chaining field and you

intend to use chaining field indicator C1. In this case, you would indicate the split chaining field by entering C1 in columns 61 and 62 on each line where these fields are described. When you specify a split chaining field, the lines that specify it cannot be interspersed with other chaining field lines.

When you use input chaining and a no-record-found error occurs during program execution (a record was read from the chaining file and a record with the same value as that in the chaining field was not found in the chained file), the H0 indicator is set on. The error code R (hexadecimal D9) is placed in the \*ERROR field if a direct file is involved, or the error code U (hexadecimal E4) is placed there if an indexed sequential file is involved. The H0 indicator remains on until the completion of the current operating cycle. If it is not set off before the next cycle begins, the program goes through end-of-job processing and terminates. You can provide for this type of error by including a routine on the calculation specifications form that tests the \*ERROR field for either error code R or U when the H0 indicator is set on. If either is present, it sets off the H0 indicator.

### 6.3.7. Field Record Relation (Columns 63 and 64)

You use this field when there is an OR relationship (OR in columns 14 and 15) between the records in an input file and when the fields in these records are not in the same location. This field is used to specify when a particular field is used by your program; that is, the field described on this line is only used when a particular record type is read or when any record type in the OR relationship is read.

If the field described on this line is used when any record type in the OR relationship is read, leave this field blank.

If the field described on this line is only used when a particular record type is read, you associate this field with the record type by entering the record identifying indicator for the record type (from columns 19 and 20) in this field.

If the field described on this line is only used when a matching condition occurs (file processing using matching records technique - the matching fields of the primary file match those of a secondary file and the matching record indicator, MR, is set on), enter MR in this field.

If the field described on this line is only used when an external indicator (U1 through U8) has been set on, enter the appropriate external indicator in this field.

When you use control level fields (specified in columns 59 and 60) or matching or chaining fields (specified in columns 61 and 62), those fields without a field record relation indicator must appear on the input format specifications form before those with field record relation indicators. Those fields with the same field record indicator must be grouped together. When the indicator is on, they will take precedence over those fields of the same level without field record relation indicators.

When you use split chaining fields with field record relation indicators, you must specify a complete chaining field for each record relation indicator you use.

When all matching fields are used with field record relation indicators, you must specify a complete set of matching fields for each record relation indicator you use.

If one matching field is used without a field record relation indicator, you must specify a complete set of matching fields without a field record relation indicator.

### **6.3.8. Field Indicators (Columns 65 through 70)**

As you can see, this area of the form contains three fields: PLUS (columns 65 and 66), MINUS (columns 67 and 68), and ZERO or BLANK (columns 69 and 70). You use these fields to test the contents of the input record field described on this line for a particular condition by entering an indicator in the appropriate field. If the condition is present, the indicator is set on. You can use this indicator to condition subsequent calculation and output operations.

You can use the PLUS field to test the contents of a numeric field for a plus condition (the value in the field is greater than zero), the MINUS field to test the contents of a numeric field for a minus condition (the value in the field is less than zero), or the ZERO or BLANK field to test contents of a numeric field for a zero condition (the field contains all zeros) or an alphanumeric field for a blank condition (the field contains all blanks).

If you have a numeric input record field, you can use any or all fields to test it. If you specify the same indicator for all fields, the indicator is set on regardless of the contents of the input record field.

If you have an alphanumeric input record field, you can only use the ZERO or BLANK field to test it. The PLUS and MINUS fields cannot be used to test an alphanumeric field. These fields must be left blank for alphanumeric fields.

A field indicator is initially set on when an input record is read that contains the field defined on this line whose contents meet the conditions specified (plus, minus, zero, or blank). The indicator remains on until the same type of record is read in again. At this time, the field from the new record is tested against the specified conditions; the indicator is set on if the conditions are met or it is set off if the conditions are not met.

If you do not want to test an input record field for a particular condition, leave the appropriate field blank.

If you want to test an input record field for a particular condition, enter an indicator in the appropriate field. You can use the general indicators (01 through 99) or the halt indicators (H1 through H9) in these fields. If you specify a halt indicator, your program will halt after it processes the input record that meets the specified conditions.

If you have specified an array name in the field name field (columns 53 through 58), the field indicators must be blank.

## **6.4. EXAMPLES OF ENTRIES ON THE INPUT FORMAT SPECIFICATIONS FORM**

Figure 6-5 shows you examples of typical entries on the input format specifications form.





## Explanation of entries in Figure 6-5:

<u>Page No.</u>	<u>Line No.</u>	<u>Explanation</u>
01	010	The records on the input file named LEVFILE are not arranged in any special sequence (AA in columns 15 and 16). Indicator 01 (01 in columns 19 and 20) is set on if a record is read that contains the character L in record position 1 (1 in columns 21 through 24, C in column 26, and L in column 27).
01	020-040	RANK (RANK in columns 53 through 58 on line 020), SERNO (SERNO in columns 53 through 58 on line 030) and NAME (NAME in columns 53 through 58 on line 040) are noncontiguous alphanumeric fields located in record positions 60 through 80, 1 through 10, and 30 through 44, respectively (60 in columns 44 through 47, 80 in columns 48 through 51, and blank in column 52 on line 020; 1 in columns 44 through 47, 10 in columns 48 through 51, and blank in column 52 on line 030; 30 in columns 44 through 47, 44 in columns 48 through 51, and blank in column 52 on line 040). These fields are defined as a split control field that RPG II will consider as one composite control field during processing because each field has the same control level specified for it (L1 in columns 59 and 60 on line 020, 030, and 040).
01	060	The records in the input file named CFILE are not arranged in any special sequence (BB in columns 15 and 16). Indicator 02 (02 in columns 19 and 20) is set on if a record is read that contains the character 5 in record position 21 (21 in columns 21 through 24, C in column 26, and 5 in column 27).
01	070-090	PARNO (PARNO in columns 53 through 58 on line 070), CLASS (CLASS in columns 53 through 58 on line 080), and REGN (REGN in columns 53 through 58 on line 090) are noncontiguous alphanumeric fields located in record positions 1 through 14, 32 through 34, and 40 through 45, respectively (1 in columns 44 through 47, 14 in columns 48 through 51, and blank in column 52 on line 070; 32 in columns 44 through 47, 34 in columns 48 through 51, and blank in column 52 on line 080; 40 in columns 44 through 47, 45 in columns 48 through 51, and blank in column 52 on line 090). These fields are defined as a split input chaining field that RPG II will consider as one composite input chaining field during processing because each field has the same chaining indicator specified for it (C2 in columns 61 and 62 on line 070, 080, and 090).

<u>Page No.</u>	<u>Line No.</u>	<u>Explanation</u>
01	110 and 120	The records on the input file NEWMAS are not arranged in any special order (CC in columns 15 and 16 on line 110). Indicator 03 (03 in columns 19 and 20) is set on if a record is read that contains the character H in record position 1 (1 in columns 21 through 24, C in column 26, and H in column 27 on line 110), the character F in record position 2 (2 in columns 28 through 31, C in column 33, and F in column 34 on line 110), the character G in record position 3 (3 in columns 35 through 38, C in column 40, and G in column 41 on line 110) and the character E in record position 4 (4 in columns 21 through 24, C in column 26, and E in column 27 on line 120).
01	130	Indicator 04 (04 in columns 19 and 20) is set on if a record is read that contains the character A in record position 5 (5 in columns 21 through 24, C in column 26, and A in column 27).
01	140	Indicator 05 (05 in columns 19 and 20) is set on if a record is read that contains the character B in record position 7 (7 in columns 21 through 24, C in column 26, and B in column 27).
01	150 and 160	If indicator 03 (03 in columns 63 and 64 on lines 150 and 160) is on, the first matching field (M1 in columns 61 and 62 on line 150) is DEP (DEP in columns 53 through 58 on line 150) and it is located in record positions 1 through 4 (1 in columns 44 through 47 and 4 in columns 48 through 51 on line 150). The second matching field (M2 in columns 61 and 62 on line 160) is LOCT (LOCT in columns 53 through 58 on line 160) and it is located in record positions 5 through 8 (5 in columns 44 through 47 and 8 in columns 48 through 51 on line 160).
01	170 and 180	If indicator 04 is on (04 in columns 63 and 64 on lines 170 and 180) the first and second matching fields have the same names as those on lines 150 and 160 but they are located in record positions 11 through 14 and 15 through 18 respectively.
01	190 and 200	If indicator 05 is on (05 in columns 63 and 64 on lines 190 and 200) the first and second matching fields have the same names as those on lines 150 and 160 but they are located in record positions 21 through 24 and 25 through 28, respectively.
02	010	The records on the input file OLDMAS are not arranged in any special sequence (DD in columns 15 and 16). Indicator 06 (06 in columns 19 and 20) is set on if a record is read that contains the character A in record position 1 (1 in columns 21 through 24, C in column 26, and A in column 27).



<u>Page No.</u>	<u>Line No.</u>	<u>Explanation</u>
02	020	Indicator 07 (07 in columns 19 and 20) is set on if a record is read that contains the character B in record position 1 (1 in columns 21 through 24, C in column 26, and B in column 27).
02	030	Indicator 08 (08 in columns 19 and 20) is set on if a record is read that contains the character C in record position 1 (1 in columns 21 through 24, C in column 26, and C in column 27).
02	040 and 050	If neither indicator 07 nor 08 is on (columns 63 and 64 are blank on lines 040 and 050), the first matching field (M1 in columns 61 and 62 on line 040) is named DEP (DEP in columns 53 through 58) and it is located in record positions 1 through 4 (1 in columns 44 through 47 and 4 in columns 48 through 51 on line 040). The second matching field (M2 in columns 61 and 63 on line 050) is named LOCT (LOCT in columns 53 through 58 on line 050) and it is located in record positions 5 through 8 (5 in columns 44 through 47 and 8 in columns 48 through 51 on line 050).
02	060	If indicator 07 is on (07 in columns 63 and 64), the first and second matching fields then have the same names as those on lines 040 and 050, but the first matching field is located in record positions 9 through 12. In this case, the location of the first matching field is changed and the second remains the same.
02	070	If indicator 08 is on (08 in columns 63 and 64), the first and second matching fields have the same names as those on lines 040 and 050, but the second matching field is located in record positions 13 through 16. In this case, the location of the second matching field is changed and the first remains the same.
02	090	The records in the input file named NFILE are not arranged in any special sequence (EE in columns 15 and 16). Indicator 09 (09 in columns 19 and 20) is set on if a record is read that contains the character 4 in record position 1 (1 in columns 21 through 24, C in column 26, and 4 in column 27), does not contain the character 5 in record position 2 (2 in columns 28 through 31, N in column 32, C in column 33, and 5 in column 34), and contains the character A through I or any character with an X'C' zone in record position 3 (3 in columns 35 through 38, Z in column 40, and A in column 41).
02	100	ALNUM (ALNUM in columns 53 through 58) is an alphanumeric field (columns 43 and 52 are blank) that is located in record positions 1 through 8 (1 in columns 44 through 47 and 8 in columns 48 through 51).

<u>Page No.</u>	<u>Line No.</u>	<u>Explanation</u>
02	110	NMRC (NMRC in columns 53 through 58) is a numeric field that is located in record positions 9 through 15 (9 in columns 44 through 47 and 15 in columns 48 through 51) and has 3 decimal positions (3 in column 52). If the field contains zeros, indicator 10 is set on (10 in columns 69 and 70).
02	120	PACK (PACK in columns 53 through 58) is a numeric field in packed decimal format (P in column 43) that is located in record positions 18 through 20 (18 in columns 44 through 47 and 20 in columns 48 through 51) and has no decimal positions (0 in column 52).
02	130	BNRY (BNRY in columns 53 through 58) is a binary field (B in column 43) that is located in record positions 21 and 22 (21 in columns 44 through 47 and 22 in columns 48 through 51) and has one decimal position (1 in column 52).
02	140	LEFT (LEFT in columns 53 through 58) is a numeric field with the sign to the left of the data (L in column 43) that is located in record positions 26 through 30 (26 in columns 44 through 47 and 30 in columns 48 through 51) and has two decimal positions (2 in column 52).
02	150	RITE (RITE in columns 53 through 58) is a numeric field with the sign to the right of the data (R in column 43) that is located in record positions 31 through 38 (31 in columns 44 through 47 and 38 in columns 48 through 51) and has four decimal positions (4 in column 52).

## 7. Calculation Specifications Form

### 7.1. GENERAL DESCRIPTION

The calculation specifications form (Figure 7-1) is not required for every program. It is required, however, when your program has to perform operations on input data or data that results from other calculations. As you can see, the form is divided into three areas: conditions, calculation, and resulting indicators.

You use the conditions area to specify when and under what conditions you want a calculation operation performed; that is, you specify that you want the operation performed at detail time, total time, or during the execution of a subroutine, and you also specify what indicators must be on or off in order for the operation to be performed.

You use the calculation area to specify the operation you want performed, the data the operation is performed upon, and where you want the result of the operation to be stored. You use the resulting indicators area to specify the tests you want performed on the results of an operation and the indicators you want set on to condition subsequent calculation and output operations if the conditions of the specified tests are met.

### 7.2. CONDITIONS ENTRIES (COLUMNS 7 THROUGH 17)

In the following subsections you'll see what each conditions field on the form is used for and how the entries in these fields affect your program.

#### 7.2.1. Control Level (Columns 7 and 8)

You use this field to specify when a calculation operation is performed.

If you want the calculation operation performed at detail time, leave this field blank.

If you want the calculation operation performed at total time even though a control break has not occurred, enter LO in this field. A typical use of this indicator in this field would be if you are printing a report and you want to accumulate totals for each page of the report even though there is no control break at the end of a page.



If you want the calculation operation performed at total time when a control break occurs, enter the appropriate control level indicator (L1 through L9) in this field. This indicator must be the same as that specified for the control field in columns 59 and 60 on the input format specifications form.

If you want the calculation operation performed when the last input record has been processed or when the LR (last record) indicator is set on during detail or total calculations, enter LR in this field. If you have more than one input file, the calculation operation is performed after the processing of input records from all files that are processed to end-of-file (those files that you have specified E in column 17 on the file description specifications form). If the calculation operation is part of a subroutine, enter SR in this field. When you use this field, the detail calculation operations (blank) must appear first, the total calculation operations (LO, L1 through L9) second, and the subroutine calculation operations (SR) third.

### 7.2.2. Indicators (Columns 9 through 17)

You use these fields to specify the indicators that are to be tested to determine if the calculation operation on this line is to be performed. Up to three indicators can be specified on a line. The indicators are specified in columns 10 and 11, 13 and 14, and 16 and 17. Columns 9, 12, and 15 (labeled N=NOT) are used to specify whether the associated indicator must be on or off in order for the calculation operation to be performed.

You leave these fields blank if you want the calculation operation always performed. If an indicator must be on before a calculation operation is performed, you specify this by placing that indicator in any of the indicator fields on this line. If an indicator must be off, you specify this by placing an N in one of the N=NOT fields and the indicator in the adjacent field. You can specify a general indicator (O1 through 99), a control level indicator (LO through L9), the last record indicator (LR), the matching record indicator (MR), an overflow indicator (OA through OG, OV), a halt indicator (HO, H1 through H9), an external indicator (U1 through U8), or function key indicator (KA through KN and KP through KW).

If you specify a record identifying indicator (from columns 19 and 20 on the input format specifications form), the calculation operation is performed only if a record of the type specified by the record identifying indicator is read.

If you specify a field indicator (from columns 65 through 70 on the input format specifications form), the calculation operation is performed only if the input data field meets the conditions specified by the field indicator.

If you specify a resulting indicator from a preceding calculation operation (columns 54 through 59 on this form), the calculation operation on this line is performed only if the condition specified for the result of the preceding calculation is met.

If you specify the matching record indicator (MR), the calculation operation is performed only when the matching record indicator is on.

If you specify a halt indicator (HO, H1 through H9), the calculation operation is performed when the specified indicator is set on. These indicators are normally used to cause a calculation operation to be performed when an error has been detected in the input data or in a previous calculation operation.





### 7.3. CALCULATION ENTRIES (COLUMNS 18 THROUGH 53)

In the following subsections you'll see what each calculation field on the form is used for and how the entries in these fields affect your program.

#### 7.3.1. Factor 1 and Factor 2 (Columns 18 through 27 and Columns 33 through 42)

You use these fields to specify the factors involved in the calculation operation you want performed. Factor 1 or factor 2 can be:

- a field name (one to six characters defined on the input format or calculation specifications forms);
- the name of an internal or external subroutine (defined elsewhere in the program);
- a table name (defined on the file extension specifications form);
- an array name (defined on the file extension specifications form);
- an array element;
- an operation label;
- an RPG II special field name (PAGE, PAGE1 through PAGE7, UDATE, UMONTH, UYEAR, D?TE, and \*ERROR);
- a file name;
- a numeric literal (1 to 10 digits that may be preceded by a sign and that may have a decimal point if required);
- an alphanumeric literal (one to eight characters enclosed in apostrophes);
- a hexadecimal literal (a series of hexadecimal digits enclosed in apostrophes and preceded by an X); or
- ➔ ■ a figurative constant (\*BLANK, \*BLANKS, \*ZERO, and \*ZEROS).

Destination names used as factor 1 with TAG and ENDSR operations and as factor 2 with GOTO operations must be from one to six characters long and begin with an alphabetic character.



The entry you make in factor 1 or factor 2 must be left-justified. The operation you want performed determines whether factor 1 or factor 2 is used and what type of entry you can make. We'll discuss these requirements when we cover the individual operations.

### 7.3.2. Operation (Columns 28 through 32)

You use this field to specify the operation you want performed using factor 1, factor 2, the result field, and the resulting indicators fields. The operation codes that you use in this field are discussed in the subsections that follow. Each discussion describes what the operation does and shows in chart form whether entries in the other fields (conditions fields, factor 1 and factor 2 fields, result fields and resulting indicators fields) are optional, required, or blank. These requirements are summarized in Table 7-1.

#### 7.3.2.1. Arithmetic Operations

The fields, literals, table elements, or array elements that you use in arithmetic operations must be numeric and their length cannot exceed 15 digits.

All arithmetic operations are performed algebraically. This means that the algebraic rules for signs apply.

If you are performing addition and the factors have like signs, the sign of the result is the same as the sign of the factors. If the factors have unlike signs, the sign of the result is the same as the sign of the larger factor.

If you are performing subtraction, the sign of the subtrahend (factor 2) is changed and the factors are added according to the rules for addition.

If you are performing multiplication and the factors have like signs, the sign of the result is plus. If the factors have unlike signs, the sign of the result is minus.

If you are performing division and the factors have like signs, the sign of the result (quotient) is plus. If the factors have unlike signs, the sign of the result is minus. The sign of the remainder is always the same as that of factor 1 (dividend). The result of an arithmetic operation is always signed (+ or -). A plus sign is a hexadecimal C and a minus sign is a hexadecimal D.

When you are dealing with fields that have decimal positions, automatic decimal alignment occurs before the arithmetic operation is performed. Assume that you have two 6-position fields, one with three decimal positions, the other with five decimal positions and you want to add these fields. The first field is 789.010 and the second is 5.43212. Before decimal alignment these fields appear as:

First field    789.010  
Second field 5.43212

The decimal point is not stored as part of data. The position of the decimal point is determined by number of decimal positions you specify for the field in the field description and this information is stored as part of the field characteristics. After decimal alignment takes place the fields appear as:

First field	789.010
Second field	<u>5.43212</u>
Result =	794.44212

When the decimal points are aligned, the add operation is performed. You must take care to ensure that the result field is large enough to hold the result and that you specify the proper number of decimal positions. If you do not, the result of your operation will be truncated on either end depending on the characteristics you specify for the result. For our example, the correct result field characteristics are field length = 8 and decimal positions = 5. If you specified a field length of 5 and 3 decimal positions, the contents of the result field would be 94.442 instead of 794.44212. The high order 7 and low order 1 and 2 are dropped. If truncation occurs, the resulting indicators may not be set as required.

#### 7.3.2.1.1. Add (ADD)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	ADD	Required	Required	Optional

The ADD operation algebraically adds the contents of factor 2 to factor 1 and stores the result in the result field.

#### 7.3.2.1.2. Zero and Add (Z-ADD)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	Z-ADD	Required	Required	Optional

The Z-ADD operation adds the contents of factor 2 to a field of zeros and stores the result in the result field.

**7.3.2.1.3. Subtract (SUB)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	SUB	Required	Required	Optional

The SUB operation algebraically subtracts the contents of factor 2 from factor 1 and stores the result in the result field.

**7.3.2.1.4. Zero and Subtract (Z-SUB)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	Z-SUB	Required	Required	Optional

The Z-SUB operation subtracts the contents of factor 2 from a field of zeros and stores the result in the result field. This causes the sign of the field to be changed.

**7.3.2.1.5. Multiply (MULT)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	MULT	Required	Required	Optional

The MULT operation algebraically multiplies the contents of factor 1 by the contents of factor 2 and stores the result in the result field. Note that when this operation is used, the result field must be large enough to hold the product; however, keep in mind the maximum size that can be specified for the result field for arithmetic operations is 15.

**7.3.2.1.6. Divide (DIV)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	DIV	Required	Required	Optional

The DIV operation algebraically divides the contents of factor 1 (dividend) by factor 2 (divisor) and stores the result (quotient) in the result field. If there is a remainder, it is discarded unless the next operation is a move remainder (MVR) operation. If half adjust (column 53) is specified with a DIV operation, the result will be rounded based on the value of the remainder, and the remainder will then be discarded. Half adjust cannot be specified if the next operation is an MVR operation. When the DIV operation is used, the field length restrictions are as shown in the following formulas:

$$fl_1 + (dp_2 - dp_1 + dp_r) \leq 15$$

$$fl_2 - (dp_2 - dp_1 + dp_r) \leq 15$$

where:

$fl_1$  = factor 1 (dividend) field length.

$fl_2$  = factor 2 (divisor) field length.

$dp_1$  = number of decimal positions in factor 1.

$dp_2$  = number of decimal positions in factor 2.

$dp_r$  = number of decimal positions in result field.

If half adjust is specified, the following formula applies:

$$fl_1 + (dp_2 - dp_1 + dp_r) \leq 14$$

**7.3.2.1.7. Move Remainder (MVR)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	MVR	Blank	Required	Optional

The MVR operation places zeros in result field and then moves the remainder of the immediately preceding DIV operation into the result field. If the DIV operation used factors with decimal positions, the result field for the MVR operation must be large enough to provide for the proper number of decimal positions for the remainder. The number of decimal positions required for the remainder is determined by the following formulas:

$$\left. \begin{array}{l} \text{Whichever} \\ \text{is} \\ \text{greater} \end{array} \right\} \left\{ \begin{array}{l} (dp_2 + dp_r) \\ \text{or} \\ dp_1 \end{array} \right\} = \text{number of decimal positions required for remainder}$$

where:

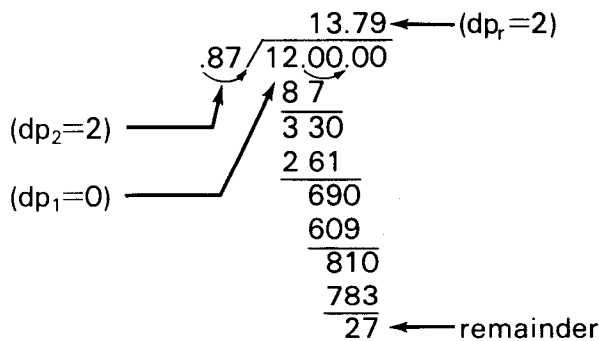
$dp_2$   
Is the number of decimal positions in factor 2 (divisor) of immediately preceding DIV operation.

$dp_r$   
Is the number of decimal positions in result field (quotient) of immediately preceding DIV operation.

$dp_1$   
Is the number of decimal positions in factor 1 (dividend) of immediately preceding DIV operation.

The remainder will have the same sign as the dividend of the immediately preceding DIV operation.

To illustrate how you determine the number of decimal positions required for the remainder in the result field, assume that you want to divide 12 by .87. You want the result field (quotient) to have two decimal positions, and you want to use the MVR operation immediately following the DIV operation. In this case, the number of decimal positions in factor 2 (divisor) is 2 ( $dp_2 = 2$ ), the number of decimal positions in the result field (quotient) is 2 ( $dp_r = 2$ ), and the number of decimal positions in factor 1 (dividend) is 0 ( $dp_1 = 0$ ).



If we use the formula,

$$\begin{array}{l} dp_2 + dp_r = 4 \\ dp_1 = 0 \end{array}$$

Since  $dp_2 + dp_r = 4$ , 4 is greater than  $dp_1 = 0$ . The number of decimal positions for the MVR is 4. You can check the accuracy of this by multiplying the quotient by the divisor and then subtracting the product from the dividend.

$$\begin{array}{r}
 13.79 \leftarrow \text{quotient} \\
 \times .87 \leftarrow \text{divisor} \\
 \hline
 9653 \\
 11032 \\
 \hline
 11.9973
 \end{array}$$

$$\begin{array}{r}
 12.0000 \leftarrow \text{original dividend} \\
 -11.9973 \\
 \hline
 .0027 \leftarrow \text{remainder}
 \end{array}$$

### 7.3.2.1.8. Square Root (SQRT)

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Optional	Blank	SQRT	Required	Required	Blank

The SQRT operation determines the square root of the contents of factor 2 and stores it in the result field. If factor 2 is zero, the result will be zero. If factor 2 contains a negative value and the operation is executed, the H0 indicator will be set on and the program will terminate. Half adjust (column 53) may be specified with this operation. Half adjust takes place after the square root has been computed, and then the adjusted square root is placed in the result field.

### 7.3.2.1.9. Cross-Foot (XFOOT)

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Optional	Blank	XFOOT	Required	Required	Optional

The XFOOT operation adds the contents of the elements of the array specified in factor 2 and places the sum in the result field. Half adjust (column 53) may be specified with this operation. Half adjust takes effect after the sum has been computed, and then the adjusted sum is placed in the result field.

### 7.3.2.2. Move Operations

The move operations allow you to move part or all of a field (factor 2) to the result field.

There are two types of move operations: those that move alphanumeric or numeric characters from one field to another (MOVE, MOVEA, and MOVEL) and those that move the zone portion of a particular byte in one field to a particular position in another field (MLLZO, MLHZO, MHHZO, and MHLZO).

You can use the MOVE and MOVEL operations to change a numeric field to an alphanumeric field and vice versa. To change a numeric field to an alphanumeric field, you specify factor 2 as numeric and the result field as alphanumeric. To change an alphanumeric field to a numeric field, you specify factor 2 as alphanumeric and the result field as numeric.

When you move an alphanumeric field or its zone to a numeric field, the sign of the result field is set to hexadecimal C if it is plus or hexadecimal D if it is minus. If the sign zone already contains a hexadecimal C or D, the sign is not changed.

When you move a numeric field or its zone to an alphanumeric field, a plus sign is changed to a hexadecimal F and a minus is left as a hexadecimal D.

In the SPERRY UNIVAC 9200/9300 mode, the sign of the result field is changed as follows: hexadecimal 6 or 9 is changed to hexadecimal D; hexadecimal A through F is unchanged; and all others are changed to hexadecimal C.

In the IBM 360/20 mode the sign of the result field is changed as follows: hexadecimal 6 is changed to hexadecimal D only if the sign byte contains hexadecimal 06; hexadecimal C through F is unchanged; and all other cases are changed to hexadecimal C.

When you use a move operation no decimal alignment is performed. You must also remember that no test is performed to ensure that valid numeric digits (0 through 9) are moved to a numeric result field. If invalid digits (any characters other than 0 through 9) are moved and you attempt to perform an arithmetic operation with the field, an arithmetic exception will result.

#### 7.3.2.2.1. Move (MOVE)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	MOVE	Required	Required	Blank

The MOVE operation causes the contents of factor 2, starting with the data characters in the rightmost positions, to be moved into the result field. If factor 2 is shorter than the result field, the leftmost characters in the result field are not disturbed. If factor 2 is longer than the result field, the leftmost characters in factor 2 in excess of the result field length are not moved.

An alphanumeric field can be converted to a numeric field and vice versa. This is accomplished by moving an alphanumeric field in factor 2 into a numeric result field in the former case and vice versa in the latter case.

If an alphanumeric field is converted to a numeric field, each character is converted to its corresponding numeric character and then moved to the numeric result field. In the case of the rightmost character in an alphanumeric field, the zone portion is converted to a + or - as applicable; that is, all characters with a hexadecimal C zone (A-I and S-Z) are positive (+), and all with a D zone (J-R) are negative (-). This sign is then used as a sign of the result field.

If a numeric field is converted to an alphanumeric field, all characters except the rightmost are moved without change into the alphanumeric result field. If the rightmost character of a numeric field is signed, it is converted to its alphanumeric equivalent, that is +1=A, +2=B, ... +9=I (hexadecimal 1C becomes hexadecimal C1) and -1=J, -2=K, ... -9=R (hexadecimal 1D becomes hexadecimal D1). If it is unsigned, it is moved without change. The effect of field length and field conversion on MOVE operations is shown in the following examples.

#### Examples - Factor 2 Shorter Than Result Field:

##### 1. Alphanumeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	SP2BC	3598777 <sup>+</sup>
After MOVE operation	SP2BC	35SP2BC

##### 2. Alphanumeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	SP2BC	35798777 <sup>+</sup>
After MOVE operation	SP2BC	35727223 <sup>+</sup>

##### 3. Numeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	1987654	987651129
After MOVE operation	1987654	981987654

##### 4. Numeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	1987654	FACEFRONT
After MOVE operation	1987654	FA1987654

#### Examples - Factor 2 Longer Than Result Field

##### 1. Alphanumeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	BPPDSP2BC	98777
After MOVE operation	BPPDSP2BC	SP2BC



## 2. Alphanumeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	BPPDSP2BC	98777 <sup>+</sup>
After MOVE operation	BPPDSP2BC	27223 <sup>+</sup>

## 3. Numeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	1987654	98765
After MOVE operation	1987654	87654

## 4. Numeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	1987654	FACEF
After MOVE operation	1987654	87654

Examples - Factor 2 and Result Field Are the Same Length:

## 1. Alphanumeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	SP2BC	98777
After MOVE operation	SP2BC	SP2BC

## 2. Alphanumeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	SP2BC	98777
After MOVE operation	SP2BC	27223

## 3. Numeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	98777	18966
After MOVE operation	98777	98777

## 4. Numeric field to alphanumeric field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	98777	SP2BC
After MOVE operation	98777	9877P

**7.3.2.2.2. Move Array (MOVEA)**

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Optional	Blank	MOVEA	Required	Required	Blank

The MOVEA operation moves data characters from factor 2 to the result field. The move proceeds from left to right, one character at a time. It begins with the first element of an array if only the array name is specified or the first character if a field name or literal is specified. If an array name with an index is specified, it begins with the indicated element. The MOVEA operation terminates when the last position of factor 2 or the result field is reached; that is, it may end in the middle of an array element.

Both factor 2 and the result field must be alphanumeric, and at least one of them must be an array. Factor 2 and the result field can both be arrays; however, they cannot be the same array.

The following examples show how the MOVEA operation works. In these examples, assume the following:

ARA1 is a 5-element array with two character elements that contains AABCCDDEE.

ARA2 is a 4-element array with three character elements that contains 11222333444.

FIELD1 is a 12-character field that contains OPQRSTUVWXYZ.

X is a 1-byte index field that contains the value 3.

Y is a 1-byte index field that contains the value 2.

Examples:

<u>Operation</u>	<u>Factor 2</u>	<u>Result Field</u>	<u>Contents of Result Field After MOVEA Operation</u>
MOVEA	ARA1	ARA2	AABCCDDEE44
MOVEA	ARA1,3	ARA2	CCDDEE333444
MOVEA	ARA1	ARA2,Y	111AABCCDDE
MOVEA	'GRANDA'	ARA1	GRANDADDEE

<u>Operation</u>	<u>Factor 2</u>	<u>Result Field</u>	<u>Contents of Result Field After MOVEA Operation</u>
MOVEA	ARA1,X	ARA2	CCDDEE333444
MOVEA	FIELD1	ARA2,4	111222333OPQ
MOVEA	ARA1,Y	FIELD1	BBCDDEEWXYZ

### 7.3.2.2.3. Move Left (MOVEL)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	MOVEL	Required	Required	Blank

The MOVEL operation causes the contents of factor 2, starting with the data characters in the leftmost positions, to be moved into the result field. If factor 2 is shorter than the result field, the rightmost characters in the result field are undisturbed. If factor 2 is longer than the result field, the rightmost characters in factor 2 in excess of the result field length are not moved.

If factor 2 is shorter than the result field, the sign of the result field remains the same when data is moved to a numeric field. If factor 2 is the same length or longer than the result field, the sign of factor 2 becomes the sign of the result field.

An alphanumeric field can be converted to a numeric field and vice versa. This is accomplished by moving an alphanumeric field in factor 2 into a numeric result field in the former case and vice versa in the latter case.

If an alphanumeric field is converted to a numeric field, each character is converted to its corresponding numeric equivalent and then moved to the numeric result field. In the case of the rightmost character in an alphanumeric field, the zone portion is converted to a + or -; that is, all characters with a hexadecimal C zone (A-I) are + and all with a D zone (J-R) are -. This sign is then used as the sign of the result field regardless of whether the rightmost character is moved to the result field.

If a numeric field is converted to an alphanumeric field, all characters except the rightmost are moved without change into the alphanumeric result field. If the rightmost character is signed and it is to be moved to the alphanumeric result field, it is converted to its alphanumeric equivalent before it is moved; that is, +1=A, +2=B, ... +9=I and -1=J, -2=K, ... -9=R. The effect of field length and field conversion on the MOVEL operation is shown in the following examples:

Examples - Factor 2 Shorter Than Result Field:

1. Numeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVEL operation	19876	987651129 <sup>+</sup>
After MOVEL operation	19876	198761129

2. Alphanumeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVEL operation	SP2BC	357987777
After MOVEL operation	SP2BC	272237777

3. Numeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVEL operation	19876	FACEFRONT
After MOVEL operation	19876	19876RONT

4. Alphanumeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVEL operation	SP2BC	357987777A
After MOVEL operation	SP2BC	SP2BC7777A

Examples - Factor 2 Longer Than Result Field:

1. Numeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVEL operation	1987654	98765 <sup>+</sup>
After MOVEL operation	1987654	19876

2. Numeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVEL operation	1084567	FACEF
After MOVEL operation	1084567	10845

## 3. Alphanumeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	BPPDSP2BC	10845
After MOVE operation	BPPDSP2BC	27742

## 4. Alphanumeric field to alphanumeric field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	BPPDSP2BC	FACEF
After MOVE operation	BPPDSP2BC	BPPDS

Examples – Factor 2 and Result Field Are the Same Length:

## 1. Numeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	19876	10845
After MOVE operation	19876	19876

## 2. Numeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	19876	FACEF
After MOVE operation	19876	19870

## 3. Alphanumeric field to numeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	BPPDN	80144 <sup>+</sup>
After MOVE operation	BPPDN	27745

## 4. Alphanumeric field to alphanumeric result field

	<u>Factor 2</u>	<u>Result Field</u>
Before MOVE operation	BPPDN	FACEF
After MOVE operation	BPPDN	BPPDN



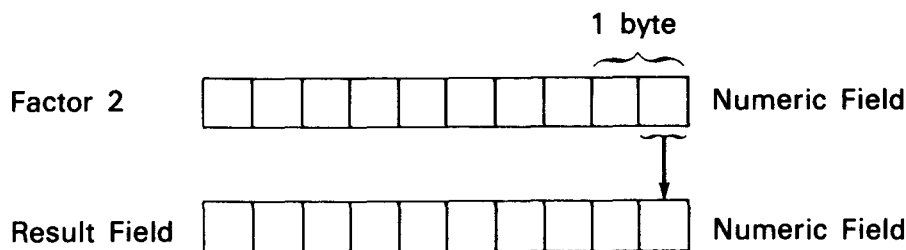
**7.3.2.2.4. Move Low Zone to Low Zone (MLLZO)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	MLLZO	Required	Required	Blank

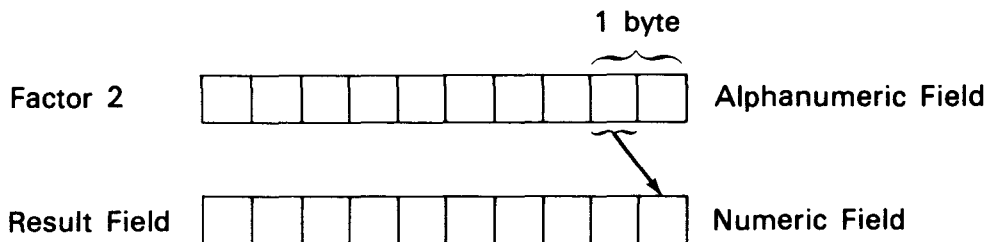
The MLLZO operation moves the zone portion of the rightmost byte of factor 2 to the rightmost byte of the result field. Factor 2 and the result field may be alphanumeric or numeric. The following examples show how the zone portion is moved in each case.

Examples:

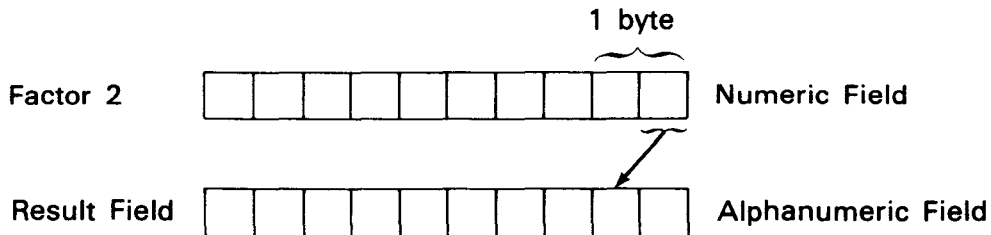
1. Numeric field to numeric result field.



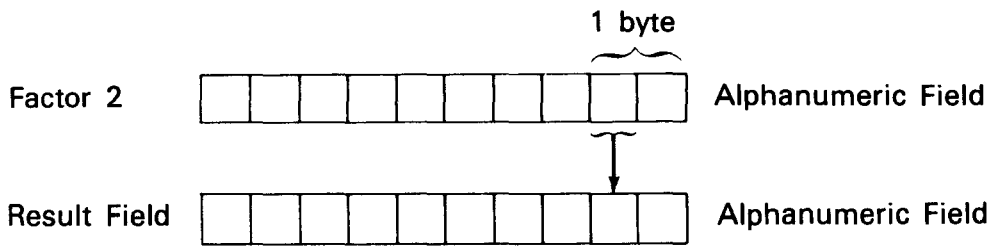
2. Alphanumeric field to numeric result field.



3. Numeric field to alphanumeric result field.



4. Alphanumeric field to alphanumeric result field.



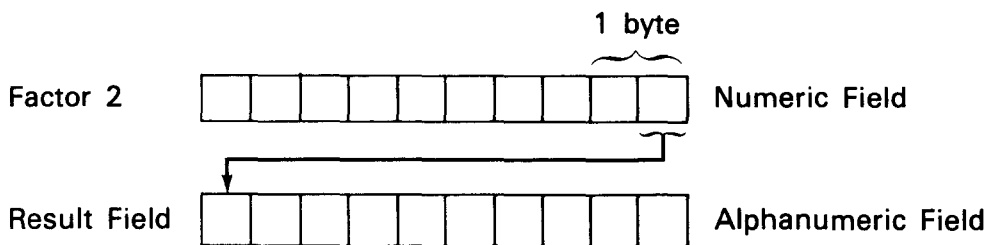
7.3.2.2.5. Move Low Zone to High Zone (MLHZO)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	MLHZO	Required	Required	Blank

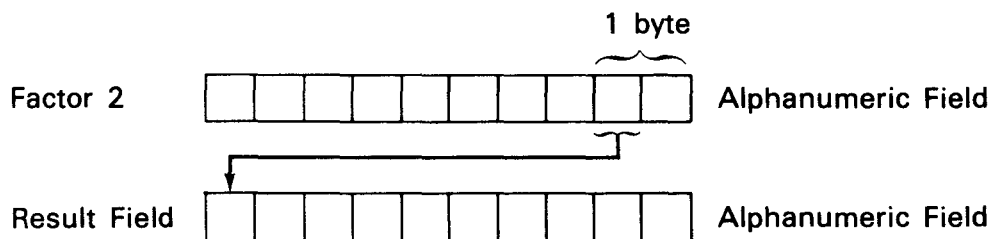
The MLHZO operation moves the zone portion of the rightmost byte of factor 2 to the leftmost byte of the result field. Factor 2 may be alphanumeric or numeric; however, the result field must be alphanumeric. The following examples show how the zone portion is moved in each case.

Examples:

1. Numeric field to alphanumeric result field.



2. Alphanumeric field to alphanumeric result field.



**7.3.2.2.6. Move High Zone to Low Zone (MHLZO)**

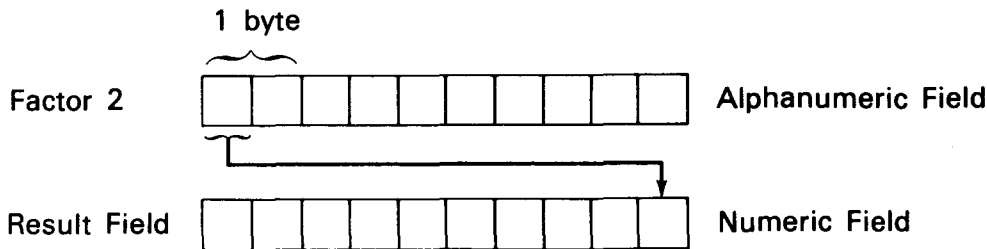
CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	MHLZO	Required	Required	Blank

The MHLZO operation moves the zone portion of the leftmost byte of factor 2 to the rightmost byte of the result field. Factor 2 must be alphanumeric; however, the result field may be alphanumeric or numeric.

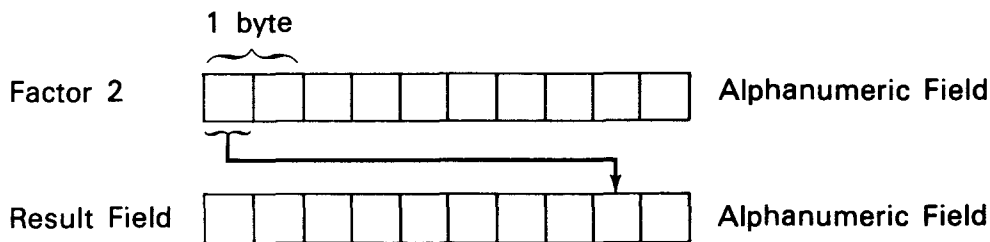
The following examples show how the zone portion is moved in each case.

Examples:

1. Alphanumeric field to numeric result field.



2. Alphanumeric field to alphanumeric result field.



**7.3.2.2.7. Move High Zone to High Zone (MHHZO)**

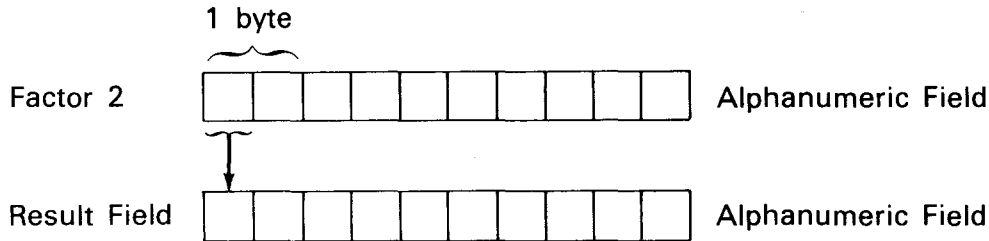
CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	MHHZO	Required	Required	Blank



The MHHZO operation moves the zone portion of the leftmost byte of factor 2 to the leftmost byte of the result field. Both factor 2 and the result must be alphanumeric. The following example shows how the zone portion is moved.

Example:

Alphanumeric field to alphanumeric result field



### 7.3.2.3. Compare and Test Operations

The compare and test operations allow you to test the contents of fields for certain conditions. You do this by specifying the fields or literals to be compared or tested and then you indicate the condition you want them tested for by entering an indicator in the appropriate resulting indicator field (columns 54 and 55, 56 and 57, or 58 and 59). If the condition is not met, that indicator is not set on. If it is met, the indicator is set on.

These operations are generally used to condition subsequent calculation or output operations; that is, by using them you can control when other operations are performed. For example, if a certain condition is not present you want one series of operations and, when the condition is present, you want another series of operations performed.

#### 7.3.2.3.1. Compare (COMP)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	COMP	Required	Blank	Required

The COMP operation compares the contents of factor 1 with factor 2. The result of the operation is then checked for the condition that is specified in the resulting indicator fields (columns 54 through 59); that is, factor 1 is greater than factor 2, factor 1 is less than factor 2, or factor 1 is equal to factor 2. A combined condition test can also be specified (for example, whether factor 1 is greater than or equal to factor 2) by using the same indicator in the two applicable resulting indicator fields. If the condition is satisfied, the specified indicator is set on. Factor 1 and factor 2 may be alphanumeric or numeric fields or literals. An array name cannot be specified in a COMP operation. An array element, however, can be specified. In this case, the entry in factor 1 or factor 2 must be in the form: array name, index.

If alphanumeric fields are compared, the comparison is based upon the internal EBCDIC collating sequence, except when an alternate collating sequence has been specified. When the fields are unequal in length, the leftmost character of the shorter field is aligned with the leftmost character of the longer field. The shorter field is then filled with blanks in as many character positions as are required to make it equal in length to the longer field.

If numeric fields are compared, the comparison is algebraic. A field with a plus sign is always greater than a field with a minus sign. When the fields are unequal in length, they are aligned at the implied decimal point. The shorter field is then filled with zeros in as many character positions to the left or right of the decimal point as are required to make it equal in length to the longer field.

A hexadecimal literal is considered alphanumeric and may only be compared with an alphanumeric field.

### 7.3.2.3.2. Test Bit (TESTB)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	TESTB	Required	Required	Required

The TESTB operation is used to test specific bits in a 1-character alphanumeric field. The field that is to be tested is specified in the result field. This field must be an alphanumeric field. Factor 2 is used to specify which bits in the result field are to be tested. The bits that are to be tested can be specified by a 1-character field or the numbers of the bits (0 through 7) enclosed in apostrophes.

If a 1-character field is specified in factor 2, the bits that are set on in this field will cause the corresponding bits in the result field to be tested. For example, if factor 2 contains hexadecimal 57 (01010111), the corresponding bits of the result field will be checked to see if they are on. In this case, bits 1, 3, 5, 6, and 7 would be checked. This could also be accomplished by specifying '13567' in factor 2.

If the bits specified by factor 2 are all zeros in the result field, the resulting indicator specified in columns 54 and 55 is set on.

If the bits specified by factor 2 are mixed (0's and 1's) in the result field, the resulting indicator specified in columns 56 and 57 is set on.

If the bits specified by factor 2 are all ones in the result field, the resulting indicator specified in columns 58 and 59 is set on.

If a field containing all zeros is specified in factor 2, no resulting indicators will be set on.

If only one bit is to be tested, a resulting indicator must not be specified in columns 56 and 57 (mixed).

**7.3.2.3.3. Test Numeric (TESTN)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	TESTN	Blank	Required	Required

The TESTN operation is used to determine if an alphanumeric result field contains numeric characters. If all the characters are numeric, the indicator specified in columns 54 and 55 is set on. If the field contains leading blanks followed by numeric characters, the indicator specified in columns 56 and 57 is set on. If the field contains all blanks, the indicator in columns 58 and 59 is set on. Each character, except the low order character, must contain a hexadecimal F in the zone portion and a digit (0 through 9) in the digit portion to be considered numeric. The low order character may contain a hexadecimal C, D, or F in the zone portion and a digit in the digit portion.

The TESTN operation is used to validate data prior to moving it into a numeric field for arithmetic and editing operations.

**7.3.2.3.4. Test Zone (TESTZ)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	TESTZ	Blank	Required	Required

The TESTZ operation tests the zone portion of the leftmost position of an alphanumeric result field for the condition specified in the resulting indicator field (columns 54 through 59); that is, the zone portion is plus (hexadecimal C zone - includes A through I,&), the zone portion is minus (hexadecimal D zone - includes J through R, -), or the zone portion is any zone other than hexadecimal C or D (includes S through Z, and all other characters). If the condition is satisfied, the specified indicator is set on.

If the zone portion is plus, the indicator specified in columns 54 and 55 is set on.

If the zone portion is minus, the indicator specified in columns 56 and 57 is set on.

If the zone portion is any other zone, the indicator specified in columns 58 and 59 is set on.

### 7.3.2.4. Branching and Exit Operations

The operations in your program are normally performed in the order they appear on the calculation specifications form. The branching and exit operations allow you to change this sequence. For example, when certain conditions are present you may want to perform a series of operations that apply only in this case, or you may have a subroutine (a series of operations performed at more than one point in your program) that you want performed at this point.

#### 7.3.2.4.1. Branch (GOTO)

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Optional	Blank	GOTO	Required	Blank	Blank

The GOTO operation is used to transfer control to a point in your program other than the next sequential operation. The point you want to branch to is specified in factor 2. This is the name you have given to your destination point; that is, the name of the TAG or ENDSR operation you want to branch to.

You can branch either backward or forward from detail-to-detail calculations, total-to-total calculations, or detail-to-total calculations. You cannot, however, branch from total-to-detail calculations.

You cannot use a GOTO operation that is outside of an internal subroutine to branch to a TAG or ENDSR within the subroutine. You can, however, use a GOTO operation within a subroutine to branch to the ENDSR operation in that routine.

#### 7.3.2.4.2. Tag (TAG)

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Blank	Required	TAG	Blank	Blank	Blank

The TAG operation is used in conjunction with the GOTO operation to specify the name of the point to which the program may branch. This name, which can be one to six characters long, is specified in factor 1. When the GOTO operation is used, factor 2 of that operation must specify the same name as the one specified in the TAG operation for the point to which the program is to branch. If a TAG operation is branched to by a total time GOTO operation, the control level field (columns 7 and 8) for that TAG operation must contain LO through LR.

**7.3.2.4.3. Begin Internal Subroutine (BEGSR)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Blank	Required	BEGSR	Blank	Blank	Blank

The BEGSR operation is the first statement in an internal subroutine (a subroutine that you have included in your program). It provides the name of the subroutine and indicates the point on the calculation specifications form where it begins. The subroutine name is specified in factor 1.

The BEGSR operation can be referenced only by an EXSR operation that specifies the same subroutine name in factor 2 as the one specified in the BEGSR operation.

All statements in an internal subroutine must have SR entered in the control level field (columns 7 and 8).

**7.3.2.4.4. End Internal Subroutine (ENDSR)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Blank	Optional	ENDSR	Optional	Blank	Blank

The ENDSR is the last statement in an internal subroutine. It indicates the point on the calculation specifications form where the subroutine ends. When the ENDSR operation is executed, it transfers control to the operation immediately following the EXSR operation that caused the subroutine to be executed.

The ENDSR operation can be branched to by a GOTO operation that is within the subroutine. In this case, factor 1 of the ENDSR operation must contain a label that can be used by the GOTO operation.



The EXSR operation transfers control to an internal subroutine. Factor 2 specifies the name of the internal subroutine. This name must also appear in factor 1 of the BEGSR operation of the subroutine that is to be executed.

When the subroutine is completed, control is returned to the operation immediately following the EXSR operation that caused the subroutine to be executed.

#### 7.3.2.4.6. Exit to Linked Subroutine (EXIT)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	EXIT	Required	Blank	Blank

The EXIT operation is used to specify a point in the calculations where control is transferred to an external subroutine that was linked to your program at linkage editor time. Factor 2 specifies the name of the external subroutine. When the EXIT operation is encountered, control is transferred to the external subroutine, the subroutine is executed, and control is returned to the program at the statement immediately following the EXIT operation.

#### 7.3.2.4.7. External Subroutine Access to RPG II Program (RLABL)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Blank	Blank	RLABL	Blank	Required	Blank

The RLABL operation allows an external subroutine (linked at linkage editor time to your program) to access a field, table, array, or indicator that is used in your program. The result field specifies the field, table, array, or indicator that is to be accessed. When the result field is a table or array name, the address plus 16 of the table linkage or array linkage field is supplied by the RPG II program. The formats of the linkage fields are shown in the system messages programmer/operator reference, UP-8076 (current version).

If an indicator is to be accessed by an external subroutine, the entry in the result field of the RLABL operation must be INnn (nn is any 2-character indicator except 1P or LO).

If more than one RPG external subroutine must access a field, table, array, or indicator, it is not necessary to provide an RLABL operation for each subroutine. One RLABL for each field, table, array, or indicator to be accessed is all that is needed.

To link your program to a FORTRAN or COBOL subroutine, every EXIT to the external subroutine must be followed immediately by the RLABL fields.

**7.3.2.4.8. RPG II Program to External Subroutine Access (ULABL)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Blank	Blank	ULABL	Blank	Required	Blank

The ULABL operation allows the RPG II program to access fields contained in an external subroutine (linked at linkage editor time to your program). The result field specifies the name of the field that is to be accessed. The field length must be specified in columns (number of digits if numeric) 49 through 51 and the number of decimal positions in column 52. The result field must not specify an array or table name or INxx. The field is in packed decimal format.

**7.3.2.5. Indicator Setting Operations**

The indicator setting operations allow you to set indicators on or off in your program. With these operations you can set any indicator on or off except the LO indicator, the matching record indicator (MR), or the first page indicator (1P). The last record (LR) indicator may be set on, but may not be set off.

**7.3.2.5.1. Set On (SETON)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	SETON	Blank	Blank	Required

The SETON operation causes the resulting indicators specified in columns 54 and 55, 56 and 57, and 58 and 59 to be set on.

**7.3.2.5.2. Set Off (SETOF)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	SETOF	Blank	Blank	Required

The SETOF operation causes the resulting indicators specified in columns 54 and 55, 56 and 57, and 58 and 59 to be set off.



### 7.3.2.6. Bit Setting Operations

The bit setting operations allow you to set individual bits in a 1-character alphanumeric field on or off.

#### 7.3.2.6.1. Set Bit On (BITON)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	BITON	Required	Required	Blank

The BITON operation is used to set bits on in a 1-character alphanumeric field. This field is specified in the result field. Factor 2 is used to specify which bits in the result field are to be set on. The bits that are to be set on can be specified by a 1-character field or the numbers of the bits enclosed in apostrophes. The bits are numbered 0 through 7 from left to right.

If a 1-character field is specified in factor 2, the bits that are set on in this field will cause the corresponding bits to be set on in the result field. The following example shows how this occurs.

	<u>Factor 2</u>	<u>Result Field</u>
Before BITON operation	11000001	01011100
After BITON operation	11000001	11011101

If the numbers of the bits to be set on are specified in factor 2, those bits in the result field that correspond to the bit numbers will be set on. The following example shows how this occurs.

	<u>Factor 2</u>	<u>Result Field</u>
Before BITON operation	'017'	01011100
After BITON operation	'017'	11011101

**7.3.2.6.2. Set Bit Off (BITOF)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	BITOF	Required	Required	Blank

The BITOF operation is used to set bits off in a 1-character alphanumeric field. It sets bits off in the same manner that the BITON operation sets bits on. As in the BITON operation, the field in which the bits are to be set off is specified in the result field and the bits that are involved are specified in factor 2 either by specifying a 1-character field or the numbers of the bits (0 through 7) enclosed in apostrophes.

**7.3.2.7. Look-Up Operations**

The look-up operations allow you to search a table or array in main storage for a specific element and retrieve it for use in subsequent calculations.

**7.3.2.7.1. Look-Up (LOKUP)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	LOKUP	Required	Optional	Required

The LOKUP operation is used to search a table or array in main storage and retrieve the required function or element that is to be used in subsequent calculations. The search argument is specified in factor 1. The table, array, or array element where the search is to begin is specified in factor 2. The result field can be used to specify a second table from which an element that corresponds to the element that satisfied the search condition can be retrieved. The result field must be blank if an array or array element is specified in factor 2.

When you specify an array element (in the form array name, index) in factor 2, the search begins with the element specified by the index. If the element is located, the index value is set to the position number of the index. If the element is not located, the index is set to 1.

The resulting indicators are used to specify the type of search that is to be performed. The search can be made for an element that is higher (an entry in columns 54 and 55), lower (an entry in columns 56 and 57), equal (an entry in columns 58 and 59), higher or equal (an entry in columns 54 and 55 and in columns 58 and 59), or lower or equal (an entry in columns 56 and 57 and in columns 58 and 59). If the search condition is satisfied, the indicator that was used to specify the search condition is set on. If columns 54 and 55, or 56 and 57 are used, the tables or arrays must be in either ascending or descending order.

### 7.3.2.8. File Processing Operations

The file processing operations allow you to retrieve records, override normal record selection, display or alter data, and write output records during calculation time.

#### 7.3.2.8.1. Retrieve Record from a Chained File (CHAIN)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	CHAIN	Required	Blank	Optional

- ➔ The CHAIN operation is used to retrieve records from indexed or direct files or to write records to a direct output file. The file that the records are retrieved from or written to is called a chained file.
- ➔ Factor 2 contains the name of the chained indexed or direct file specified on the file description form.
- ➔ If records are retrieved from an indexed file, factor 1 specifies the key of the record being sought. Factor 1 may be a field name, an array element, a table name, or a literal. The result field must be blank in this case.

If records are retrieved from a direct file by relative record number (R in column 31 of the file description specifications form) or written to a direct file being created by relative record number (blank in column 31 of the file description specifications form), factor 1 contains the relative record number of the record to be read/written. Factor 1 may be a field name, an array element, a table name, or a numeric literal.

The high field (columns 54 and 55) and low field (columns 56 and 57) of the resulting indicators can be used with the CHAIN operation to specify an indicator that is to be set on when a no-record-found condition occurs. If an indicator is not present in either the high field or the low field and a no-record-found condition occurs, the HO (halt O) indicator is set on, and the program terminates unless provision is made in the program to set this indicator off.

The equal field (columns 58 and 59) of the resulting indicators can be used to specify an indicator that is to be set on when a record is found. Every resulting indicator specified for a CHAIN operation is always set on or off, depending upon its significance and the success of the I/O operation; it is never left unaltered. If resulting indicators signifying no-record-found are set on, the indicator signifying record found is set off, and vice versa.

In IMS action programs, the CHAIN operation always uses the primary key when accessing multikey MIRAM files.

### ➔ 7.3.2.8.1A. Retrieve Record from a Chained File (REFER) – IMS Action Programs Only

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Optional	Required	REFER	Required	Optional	Blank

The REFER operation is used to retrieve records from indexed multikey files. The file that the records are retrieved from is called a chained file.

Factor 2 contains the name of the chained indexed file specified on the file description form.

Factor 1 specifies the key of the record being sought. Factor 1 may be a field name, an array element, a table name, or a literal.

↓  
The high field (columns 54 and 55) and low field (columns 56 and 57) of the resulting indicators can be used with the REFER operation to specify an indicator that is to be set on when no record is found. If no indicator is present in the high or low field and no record is found, the HO (halt 0) indicator is set on, and the program terminates unless the program sets this indicator off.

↑  
The equal field (columns 58 and 59) of the resulting indicators can be used to specify an indicator that is to be set on when a record is found. Every resulting indicator specified for a REFER operation is always set on or off, depending upon its significance and the success of the I/O operation; it is never left unaltered. If resulting indicators signifying no-record-found are set on, the indicator signifying record found is set off, and vice versa.

You can use the result field to specify the duplication count when accessing records with duplicate keys. The result field must be a numeric unsigned integer, and it can be a field name, an array element, or a numeric literal.

↓  
To access records with duplicate keys in a nonaction program, use the SETLL operation for a demand file. Then use a series of READ operations until the desired duplicate key is reached.

↑  
You'll get an IMS error if you try to update a record after reading it by using the REFER operation. To update this record, reread it using the CHAIN operation.

**7.3.2.8.2. Select Key Structure (SETK)**

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Optional	Required	SETK	Required	Blank	Blank

The SETK operation allows you to select the key structure that retrieves an indexed file. Factor 1 (column 18) must contain a number from 1 to 5. After the SETK operation selects the key structure, that structure is used until you change it. Factor 2 contains the file name. The SETK sets the file to the beginning of the key structure specified each time the operation is executed.

For IMS action programs, you can only use the SETK operation for chained or demand files. The SETK operation doesn't effect the usage of the CHAIN verb.

**7.3.2.8.3. Read Record (READ)**

CONDITIONS (Columns 9–17)	FACTOR 1 (Columns 18–27)	OPERATION (Columns 28–32)	FACTOR 2 (Columns 33–42)	RESULT FIELD (Columns 43–48)	RESULTING INDICATORS (Columns 54–59)
Optional	Blank	READ	Required	Blank	Optional

The READ operation causes a record to be immediately retrieved from a demand file. Factor 2 specifies the name of the demand file.

The equal field (columns 58 and 59) of the resulting indicators may be used with the READ operation to specify an indicator that is to be set on when the demand file reaches end-of-file. (Columns 54 through 57 must be blank.) If an indicator is not present in this field and the end-of-file is reached, the H0 (halt 0) indicator will be set on and the program will terminate.

If an indicator is present and is on before the READ operation, it will not be set off even if a record is read from the file. Therefore, it may appear that end-of-file has been reached when it has not.



**7.3.2.8.4. Set Lower Limits (SETLL)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	SETLL	Required	Blank	Blank

The SETLL operation is used to specify a lower limit for subsequent indexed sequential demand file processing via the READ operation. This allows sequential between-limits type processing of a file, beginning with any arbitrary record. The SETLL operation can be used at any time during calculations and as often as an out-of-sequence next record (lower limit) is desired.

The first READ operation processed following a SETLL operation will retrieve the record specified by the SETLL operation or the next higher record if the one specified does not exist. Subsequent READ operations will retrieve consecutive records until end-of-file is reached or a new lower limit is specified by another SETLL operation. Factor 1 may be a field, table name, array element, or a literal. The value specified as factor 1 represents the key of the record at which the subsequent READ processing is to begin. The factor 1 value must agree in length and type with the key of the demand file specified on the file description specifications form. Factor 2 must be the name of an indexed sequential demand file as specified on the file description specifications form.

**7.3.2.8.5. Override Normal Record Selection (FORCE)**

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	FORCE	Required	Blank	Blank

The FORCE operation permits you to override the normal record selection process; that is, with this operation you specify from which file the next record is to be read rather than allow the RPG II program logic to do it for you. Factor 2 specifies the name of the primary or secondary input, combined, or update file from which the next record is to be selected for processing. The first record that is processed in your program cannot be forced. It must be selected by the normal process.

The FORCE operation can be used only at detail time. When it is executed, it will remain in effect only for the next input cycle.

If more than one FORCE operation is issued during a processing cycle, the last one that was executed is effective. All other FORCE operations are overridden. When a FORCE operation is specified for a file that has reached the end-of-file condition, the normal record selection process selects the next record.

When the FORCE operation is used to select a record, the record is treated as if it had no matching fields. As a result, the MR (matching record) indicator is always off when a forced record is being processed.

#### 7.3.2.8.6. Display (DSPLY)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Optional	DSPLY	Required	Optional	Blank

The DSPLY operation permits you to enter information or have information displayed via the system console during the execution of your program. Factor 2 contains the name of the file that is specified for the CONSOLE entry on the file description specifications form. Factor 1 can be used to specify a field name, table name, array element, or literal that is to be displayed on the system console. The result field can be used to specify a field name, table name, or array element whose contents are to be altered by the operator.

If factor 1 is specified and the result field is blank, the contents of factor 1 are displayed on the system console and processing continues.

If the result field is specified and factor 1 is blank, the contents of the result field are displayed, and the program waits for the operator to alter the field or indicate that the program is to resume processing. If factor 1 and the result field are specified, the contents of factor 1 and the result field are displayed, and the program waits for the operator to alter the contents of the result field or indicate that the program is to resume processing.

When numeric data is entered, leading zeros are not required because RPG II automatically right-justifies the data. Similarly, alphanumeric data is left-justified.

Note that a \$ is not permitted in the first character position of the data to be displayed.

The DSPLY operation may not be specified for IMS action programs.

#### 7.3.2.8.7. Exception Lines (EXCPT)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	EXCPT	Blank	Blank	Blank



The EXCPT operation permits output to be written while calculations are being performed. When the EXCPT operation is encountered, all output lines on the output format specifications form that have an E in the type field (column 15) and whose output indicators (columns 23 through 31) are satisfied are written. When this is completed, the calculations continue with the next statement after the EXCPT operation.

#### 7.3.2.8.8. Debug (DEBUG)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Optional	DEBUG	Required	Optional	Blank

If 1 is specified in column 15 of the control card specifications form, the DEBUG operation causes all indicators that are on and any field or literal that is specified to be printed when the operation is encountered. If 1 is not specified in column 15 of the control card specifications form, the DEBUG operation is ignored.

Factor 1 may contain a field name or literal. This entry is used to identify the DEBUG printout. It may not be more than eight characters. Factor 2 specifies the name of the output file that is to contain the DEBUG printouts. The record lengths for this file (specified in columns 24 through 27 of the file description specifications form) must be at least 80. The result field may be used to specify a field name, table name, or array name that is to be printed as a second output line.

#### 7.3.2.8.9. Next Workstation Input (NEXT)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Required	NEXT	Required	Blank	Optional

The NEXT operation forces input from a particular workstation of a multiple workstation file. If you specify the NEXT operation more than once between input operations, only the last operation is in effect. You can only use the NEXT operation with a workstation file (WORKSTN in columns 40 through 46 on the file description specifications form).

In factor 1, enter the name of a 2-character field that contains the device identification or enter a 2-character alphanumeric literal that is the device identification. The next input to the program comes from this device. Input from all other workstations is held up.

In factor 2, enter the name of the WORKSTN file for which the operation is requested.

In the resulting indicator field, enter an indicator in columns 56 and 57. If an exception/error occurs on the NEXT operation, this indicator is set on. If you specify the INFSR subroutine (5.2.18.2) and don't enter an indicator in columns 56 and 57, the subroutine automatically receives control when an exception/error occurs. If you don't specify the INFSR subroutine and also don't enter an indicator in columns 56 and 57, the program halts when an exception/error occurs.

### 7.3.2.9. Time of Day Operations

The time of day operations allow you to access the system time of day and the system date.

#### 7.3.2.9.1. Time (TIME)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	TIME	Blank	Required	Blank

The TIME operation allows you to access the system time of day and date. The result field must specify the name of a numeric field with no decimal places. If you specify a 6-digit field, the system time of day in the form hhmmss is retrieved. If you specify a 12-digit field, the system time of day and date in the form hhmmssymmdd is retrieved.

#### 7.3.2.10. System Shutdown (SHTDN)

CONDITIONS (Columns 9-17)	FACTOR 1 (Columns 18-27)	OPERATION (Columns 28-32)	FACTOR 2 (Columns 33-42)	RESULT FIELD (Columns 43-48)	RESULTING INDICATORS (Columns 54-59)
Optional	Blank	SHTDN	Blank	Blank	Required

The SHTDN operation sets on the resulting indicator if a system shutdown is requested by the operator. The indicator then conditions the termination of the program in an orderly fashion, such as printing some partial totals and going to normal end-of-job. The resulting indicator field (columns 54 and 55) must contain 01 through 99, L1 through L9, U1 through U8, H1 through H9, or LR.

#### 7.3.2.11. Operations Summary

A summary of operations is shown in Table 7-1.

Table 7-1. Summary of Operations (Part 1 of 2)

Conditions (Columns 9-17)	Factor 1 (Columns 18-27)	Operation (Columns 28-32)	Factor 2 (Columns 33-42)	Result Field (Columns 43-48)	Resulting Indicators (Columns 54-59)
Optional	Required	ADD	Required	Required	Optional
Blank	Required	BEGSR	Blank	Blank	Blank
Optional	Blank	BITOF	Required	Required	Blank
Optional	Blank	BITON	Required	Required	Blank
Optional	Required	CHAIN	Required	Blank	Optional
Optional	Required	COMP	Required	Blank	Required
Optional	Optional	DEBUG	Required	Optional	Blank
Optional	Required	DIV	Required	Required	Optional
Optional	Optional	DSPLY	Required	Optional	Blank
Blank	Optional	ENDSR	Optional	Blank	Blank
Optional	Blank	EXCPT	Blank	Blank	Blank
Optional	Blank	EXIT	Required	Blank	Blank
Optional	Blank	EXSR	Required	Blank	Blank
Optional	Blank	FORCE	Required	Blank	Blank
Optional	Blank	GOTO	Required	Blank	Blank
Optional	Required	LOKUP	Required	Optional	Required
Optional	Blank	MHHZO	Required	Required	Blank
Optional	Blank	MHLZO	Required	Required	Blank
Optional	Blank	MLHZO	Required	Required	Blank
Optional	Blank	MLLZO	Required	Required	Blank
Optional	Blank	MOVE	Required	Required	Blank
Optional	Blank	MOVEA	Required	Required	Blank
Optional	Blank	MOVEL	Required	Required	Blank
Optional	Required	MULT	Required	Required	Optional
Optional	Blank	MVR	Blank	Required	Optional
Optional	Required	NEXT	Required	Blank	Optional
Optional	Blank	READ	Required	Blank	Optional
Blank	Blank	RLABL	Blank	Required	Blank
Optional	Required	SETK	Required	Blank	Blank
Optional	Required	SETLL	Required	Blank	Blank

Table 7-1. Summary of Operations (Part 2 of 2)

Conditions (Columns 9-17)	Factor 1 (Columns 18-27)	Operation (Columns 28-32)	Factor 2 (Columns 33-42)	Result Field (Columns 43-48)	Resulting Indicators (Columns 54-59)
Optional	Blank	SETOF	Blank	Blank	Required
Optional	Blank	SETON	Blank	Blank	Required
Optional	Blank	SHTDN	Blank	Blank	Required
Optional	Blank	SQRT	Required	Required	Blank
Optional	Required	SUB	Required	Required	Optional
Blank	Required	TAG	Blank	Blank	Blank
Optional	Blank	TESTB	Required	Required	Required
Optional	Blank	TESTN	Blank	Required	Required
Optional	Blank	TESTZ	Blank	Required	Required
Optional	Blank	TIME	Blank	Required	Blank
Blank	Blank	ULABL	Blank	Required	Blank
Optional	Blank	XFOOT	Required	Required	Optional
Optional	Blank	Z-ADD	Required	Required	Optional
Optional	Blank	Z-SUB	Required	Required	Optional

### 7.3.3. Result Field (Columns 43 through 53)

This field is made up of four subfields: name, field length, decimal positions, and half adjust. The subfields are described in 7.3.3.1 through 7.3.3.4.

#### 7.3.3.1. Name (Columns 43 through 48)

You use this field to specify the field name, table name, array name, or array element where the result of an operation is placed.

The entry in this field must be left-justified and it may consist of from one to six alphanumeric characters. The first character must be an alphabetic character. The RPG II special field names and look-ahead fields must not be used in this field, but PAGE field is permitted.

#### 7.3.3.2. Field Length (Columns 49 through 51)

You use this field to specify the length of the result field. The entry must be right-justified, leading zeros may be omitted.

If you have defined the result field on the input format specifications form or the file extension specifications form, you can leave this field blank.

If you have not defined the result field, you must make an entry in this field. This entry can range from 1 to 256. The maximum length for a numeric field is 15 and the maximum length for an alphanumeric field is 256.

If a packed numeric input field is used on the calculation specifications form, the number of digits in the field, rather than the number of bytes (character positions) that it occupies, must be considered when the length of the result field is specified. This length can be calculated by using the following formula:

$$\text{result field length} = 2(\text{input field length}) - 1$$

For example, assume that a packed numeric input field is specified as having a length of 7 in columns 44 through 51 of the input format specifications form. The length of the result field must be 13; that is,  $2(7) - 1$ , or 13.

#### **7.3.3.3. Decimal Positions (Column 52)**

You use this field to specify the number of decimal positions if the result field is numeric.

If the result field is alphanumeric, leave this field blank.

If the result field is numeric and it has been defined on the input format specifications form or the calculation specifications form, you can leave this field blank.

If the result field is numeric and it has not been previously defined, enter the number of decimal positions (0 through 9) in this field. If the result field is numeric and it has no decimal positions, you must enter a 0 in this field.

#### **7.3.3.4. Half Adjust (Column 53)**

You use this field to cause the value in a numeric result field to be rounded to the least significant decimal place. Rounding is accomplished by algebraically adding 5 (-5 if the result field is negative) to the value in the result field one position to the right of the last specified decimal position in that field.

If the associated operation is not an arithmetic operation, the result field is alphanumeric and therefore, rounding is not required; leave this field blank. If you want rounding performed, enter an H in this field.

If you specify rounding with a DIV (divide) operation, the MVR (move remainder) cannot be used with this DIV operation.

If the number of decimal positions in the arithmetic result is less than or equal to the number of decimal positions specified for the result field and rounding has been specified, the rounding specification has no effect.

The resulting indicators are set according to the value of the result field after rounding has taken place.

## 7.4. RESULTING INDICATOR ENTRIES (COLUMNS 54 THROUGH 59)

The following shows how each resulting indicator on the form is used and how the entries in these fields affect your program.

The resulting indicator fields consist of columns 54 and 55 (+, 1>2, or high field), columns 56 and 57 (-, 1<2, or low field) and columns 58 and 59 (0, 1=2, or equal field). The entries you make in these fields specify the tests you want performed or the result of an operation and the indicator you want set on if the conditions of the tests are met. You can specify a general indicator (O1 through 99), a halt indicator (HO through H9), an external indicator (U1 through U8), a control level indicator (L1 through L9), the last record indicator (LR), an overflow indicator (OA through OG, or OV), or a function key indicator (KA through KN and KP through KW) from columns 33 and 34 of the file description specifications form in any of these fields.

Any indicator that you specify is set off if the test conditions are not met. If a control level indicator is set on as the result of an operation, the lower level indicators are not set on.

When a halt indicator (HO through H9) is set on, the program will terminate unless the indicator is set off.

If you use a COMP, LOKUP, SETON, SETOF, TESTB, TESTN, or TESTZ operation, you must specify a resulting indicator.

### 7.4.1. +, 1>2, or High (Columns 54 and 55)

You use this field to specify a test for any of the following conditions and to specify the indicator you want set on if the condition is satisfied.

- The bits tested in a TESTB operation are all 0's.
- Factor 1 is greater than factor 2 in a COMP operation.
- The result of an arithmetic operation is plus.
- The zone tested in a TESTZ operation is a plus zone (EBCDIC characters A-I, &, and hexadecimal CA-CF).
- A CHAIN operation was performed and the record was not found.
- The alphanumeric field tested in a TESTN operation contains only numeric characters. All characters except the low order character have a hexadecimal F zone portion and a digit portion (0 through 9). The low order character has a hexadecimal C, D, or F zone portion and a digit portion (0 through 9).
- You can also use this field to search for the element in the factor 2 table/array that is closest to, yet greater than, factor 1 in a LOKUP operation.

#### 7.4.2. -, 1<2, or Low (Columns 56 and 57)

You use this field to specify a test for any of the following conditions and to specify the indicator you want set on if the condition is satisfied.

- The bits tested in a TESTB operation are mixed (0's and 1's).
- Factor 1 is less than factor 2 in a COMP operation.
- The result of an arithmetic operation is minus.
- The zone tested in a TESTZ operation is a minus zone (EBCDIC characters -, J-R, and hexadecimal DA-DF).
- A CHAIN operation was performed and the record was not found.
- The alphanumeric field tested in a TESTN operation contains numeric characters and leading blanks.

You can also use this field to search for an element in the factor 2 table/array that is closest to, yet less than, factor 1 in a LOKUP operation.

#### 7.4.3. 0, 1=2, or Equal (Columns 58 and 59)

You use this field to specify a test for any of the following conditions and to specify the indicator you want set on if the condition is satisfied.

- The bits tested in a TESTB operation are all 1's.
- Factor 1 is equal to factor 2 in a COMP operation.
- The result of an arithmetic operation is zero. If the result field that is being tested for 0's is an output field that will also be reset to 0 after it is placed in the output record (B in column 39 of the output format specifications form), the indicator specified on the calculation specifications form will not be set on after the field is reset to 0's. If, however, S is specified in column 42 of the control card, the first indicator in a test for 0's associated with this field will be set on after the field is reset to 0's.
- End-of-file was reached on a demand file during a READ operation.
- The zone tested in a TESTZ operation is neither plus nor minus (EBCDIC characters other than &, -, or those with a hexadecimal C or D zone).
- The alphanumeric field tested in a TESTN operation contains all blanks.
- A CHAIN operation was performed and the record was found.

You can also use this field to search for an element in the factor 2 table/array that is equal to factor 1 in a LOKUP operation.

#### 7.4.4. Comments (Columns 60 through 74)

You can use this field to insert comments that relate to the calculations on the individual lines of the form. When your program is compiled, these comments will have no effect on the compilation process; however, they will in the source program listing that is produced when your program is compiled. When you use this field you do not require an \* in column 7.

#### 7.5. EXAMPLES OF ENTRIES ON THE CALCULATION SPECIFICATIONS FORM

Figure 7-4 shows you examples of typical entries on the calculation specifications form.

Explanation of entries in Figure 7-4:

<u>Line Number</u>	<u>Explanation</u>
--------------------	--------------------

010	This operation is performed each time a detail record is read (columns 7 through 17 are blank). The operation consists of adding (ADD in columns 27 through 32) the contents of FIELD1 (FIELD1 in columns 18 through 28) to the contents of FIELD2 (FIELD2 in columns 33 through 42) and storing the result in the field named NET (NET in columns 43 through 48). The field named NET is six characters in length (6 in columns 49 through 51), contains two decimal places (2 in column 52), and the contents of the field are half-adjusted or rounded (H in column 53).
020	This operation is performed when indicator 01 is on (01 in columns 10 and 11). The operation consists of subtracting (SUB in columns 28 through 32) the contents of FIELD3 (FIELD3 in columns 33 through 42) from the contents of FIELD4 (FIELD4 in columns 18 through 27), and storing the result in the field named GROSS (GROSS in columns 43 through 48). If the result of this operation is plus, indicator 11 is set on (11 in columns 54 and 55). If the result is minus, indicator 12 is set on (12 in columns 56 and 57). If the result is zero, indicator 13 is set on (13 in column 58 and 59).
030	The operation is performed when indicator 02 is on (02 in columns 10 and 11) and indicator 03 is not on (N in column 12 and 03 in columns 13 and 14). The operation consists of a table lookup operation (LOKUP in columns 28 through 32). The search argument is 10 (10 in columns 18 through 27) and it is used to search the table TAB100 (TAB100 in columns 33 through 42). The table TAB200 (TAB200 in columns 43 through 48) is the table that contains the functions corresponding with the arguments in table TAB100. Indicator 14 (14 in columns 58 and 59) specifies that table TAB100 is to be searched for an element equal to the search argument. If it is found, the indicator will be set on and the corresponding element in table TAB200 will be made available.

(continued)



PAGE NO	FORM TYPE	LINE NO	CONTROL LEVEL LO-LB-LR-SR	CONDITIONS											CALCULATION										RESULTING INDICATORS						COMMENTS	PROGRAM IDENTIFICATION						
				INDICATORS											FACTOR 1	OPERATION	FACTOR 2	RESULT FIELD				ARITHMETIC																
				AND		AND		N-NOT	N-NOT	N-NOT	N-NOT	N-NOT	N-NOT	N-NOT				NAME	FIELD LENGTH	DECIMAL POSITIONS	HALF ADJUST	+	-	0														
				A	N	O	R								1 > 2	1 < 2	1 = 2																					
1	2	3	5	6	7	8	9	10	11	12	13	14	15	16	17	18	27	28	32	33	42	43	48	49	51	52	53	54	55	56	57	58	59	60	74	75	80	
		0,1	O	c												FIELD1	ADD	FIELD2	NET					62H														
		0,2	O	c			01									FIELD4	SUB	FIELD3	GROSS									11	12	13								
		0,3	O	c			02	N03								10	LOOKUP	TAB100	TAB200																			
		0,4	O	c			N02	04	N03								READ	DMAND																				
		0,5	O	c	L2												EXSR	SUBRT																				
		0,6	O	c	L2												§																					
		0,7	O	c	L2												§																					
		0,8	O	c	SR											SUBRT	BEGSR																					
		0,9	O	c	SR												§																					
		1,0	O	c	SR												§										09	10										
		1,1	O	c	SR	09											GOTO	END																				
		1,2	O	c	SR	10											GOTO	MIDPT																				
		1,3	O	c	SR												§																					
		1,4	O	c	SR												§																					
		1,5	O	c	SR											MIDPT	TAG																					
		1,6	O	c	SR												§																					
		1,7	O	c	SR												§																					
		1,8	O	c	SR											END	ENDSR																					
		1,9	O	c																																		
		2,0	O	c																																		

Figure 7-4. Examples of Entries on Calculation Specifications Form

Line Number Explanation

- 040 The operation is performed when indicator 04 is on (04 in columns 13 and 14) and indicator 02 and 03 are off (N in column 9, 02 in columns 10 and 11, N in column 15, and 03 in columns 16 and 17). The operation consists of retrieving a record (READ in columns 28 through 32) from a demand file named DMAND (DMAND in columns 33 through 42). When end-of-file is encountered, indicator 15 will be set on (15 in columns 58 and 59).
- 050 The internal subroutine SUBRT (SUBRT in columns 33 through 42) is executed (EXSR in columns 28 through 32) at total time when indicator L2 is on (L2 in columns 7 and 8).
- 060-070 Additional calculations to be performed at total time when indicator L2 is on.
- 080 The operation on this line is part of an internal subroutine (SR in columns 7 and 8.). All other lines in the subroutine will also contain SR in columns 7 and 8. This operation indicates the entry point (BEGSR in columns 28 through 32) of the subroutine named SUBRT (SUBRT in columns 18 through 27).
- 090-100 Additional steps of the subroutine. As a result of the operation on line 100, indicator 09 or 10 may be set on (09 in columns 54-55 and 10 in columns 56 and 57).
- 110 This operation is performed when indicator 09 is on (09 in columns 10 and 11). The program branches (GOTO in columns 28 through 32) to the line labeled END (END in columns 33 through 42). The line labeled END is line 180.
- 120 This operation is performed when indicator 10 is on (10 in columns 10 and 11). The program branches (GOTO in columns 28 through 32) to the line labeled MIDPT (MIDPT in columns 33 through 42). The line labeled MIDPT is line 150.
- 130-140 When both indicators 9 and 10 are off, these lines are executed.
- 150 This operation (TAG in columns 28 through 32) provides a label (MIDPT in columns 18 through 27) for a line that is to be branched to.
- 160-170 These lines are executed when the subroutine branches to line 150.
- 180 This operation indicates that this is the last line in the subroutine (ENDSR in columns 28 through 32) and that this line is labeled END (END in columns 18 through 27). When this point is reached, control is returned to the main program at the line immediately following the EXSR operation. In this case, control is returned to line 060.

## 8. Output Format Specifications Form

### 8.1. GENERAL DESCRIPTION

You must have an output format specifications form (Figure 8-1) in all programs. This form describes the output you want produced. There are two types of entries: output file identification and control, and field description and control.

You use the output file identification and control entries to identify the output files (printer, tape, disk, and punched card), to identify the types of records in each file, to control the spacing of the lines on printed reports, to control the stacker selection for punched card output, and to control when the output records are written.

You use the field description and control entries to define the individual fields of an output record, to describe the kind of data in each field, to control when a field is written, and to specify the editing that you want performed on the field.

### 8.2. OUTPUT FILE IDENTIFICATION AND CONTROL ENTRIES (COLUMNS 7 THROUGH 31)

In the following subsections you'll see what each output file identification and control field is used for and how the entries in these fields affect your program.

#### 8.2.1. File Name (Columns 7 through 13)

You use this field to specify the name of the output file that you want to receive the output records. The name that you enter in this field must be the name you specified for this file on the file description specifications form. The file name for each output file must appear on the first file identification line.

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES



PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/FETCH OVERFLOW		SPACE		SKIP		OUTPUT INDICATORS						FIELD NAME	DATA FORMAT P/B/L/R	CODES				NOT USED	PROGRAM IDENTIFICATION															
			TYPE H/D/T/E	FILE NAME	BEFORE	AFTER	BEFORE	AFTER	N NOT	AND	AND	N NOT	N NOT	N NOT			NONE	CR	COMMAS INSERTED	ZERO BALANCE TO PRINT			X	ACTION													
1	2	3	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45	70	71	74	75	80	
		0, 1	O																																		
		0, 2	O																																		
		0, 3	O																																		
		0, 4	O																																		
		0, 5	O																																		
		0, 6	O																																		
		0, 7	O																																		
		0, 8	O																																		
		0, 9	O																																		
		1, 0	O																																		
		1, 1	O																																		
		1, 2	O																																		
		1, 3	O																																		
		1, 4	O																																		
		1, 5	O																																		
		1, 6	O																																		
		1, 7	O																																		
		1, 8	O																																		
		1, 9	O																																		
		2, 0	O																																		

UD1-1165 REV. 8-82

Figure 8-1. Output Format Specifications Form

### 8.2.2. Type (Column 15)

You use this field to identify the type of output record with regard to what point in the processing cycle it is to be written, that is, at detail time, total time, or calculation time. You identify the record type by entering one of the following letters in this field: H (heading record), D (detail record), T (total record), or E (exception record).

A heading record is written at detail time and usually contains heading information for a printed report.

A detail record is written at detail time and usually contains data from input records or calculations performed at detail time.

A total record is written at total time (when a control break occurs) or when data is exhausted. It usually contains summaries or totals of data accumulated from detail time calculations. An exception record is written during detail or total calculation time when the EXCPT operation is encountered.

### 8.2.3. Stacker Select/ Fetch Overflow (Column 16)

You use this field to specify the stacker that you want the cards placed in when you use a card device that has more than one stacker or you use it to specify that you want the fetch overflow routine to be used to cause overflow processing at this point in the program, rather than at the completion of the total cycle.

The 0604 card punch subsystem has a normal stacker and an alternate stacker. The 0605 card punch subsystem has only one stacker.

If you are using the 0604 card punch subsystem and you want the cards placed in the normal stacker, enter a 1 in this field. If you want the cards placed in the alternate stacker, enter a 2 in this field.

If you are using the 0605 card punch subsystem, you must leave this field blank because this subsystem has only one stacker.

When a record is written to a **combined file**, the entry in this field overrides the stacker select entry in column 42 on the input format specifications form.

If you want the fetch overflow routine to be used to cause overflow processing at this point in your program, enter an F in this field. When you request the fetch overflow routine, and all output conditions specified in columns 23 through 31 are satisfied, the overflow indicator specified for this output file (columns 33 and 34 on the file description specifications form) is checked to see whether it is set on. If it is on, all overflow output records associated with the output file are written, and then the program returns to the line that requested the fetch overflow routine and writes the output record associated with it.

If you use stacker select or fetch overflow and you have AND lines for a record type, the entry need only be made once.

If you have OR lines, you must make the entry on each line of the OR relationship.

This field is ignored for IMS action programs.

#### **8.2.4. AND/OR Relationship (Columns 14 through 16)**

→ You use this field to establish an AND or OR relationship. You use an AND relationship when more than three indicators are needed to condition when an output record is written and an OR relationship when more than one indicator can condition when an output record is written. AND/OR lines cannot be used to condition output fields. The maximum number of AND/OR relationships per record definition is 80.

You establish an AND relationship by placing the first three output indicators on the first line, the additional entries on succeeding lines, and an AND in columns 14 through 16 on each additional line that is used. If you use fetch overflow or stacker select (column 16) in an AND relationship, it is specified on the first line. Columns 17 through 22 (space and skip fields) must be blank on AND lines.

You establish an OR relationship by placing OR in columns 14 and 15 and the alternate output indicator entries in columns 23 through 31 on the next line. If you need additional alternate output indicators, these can be specified on additional OR lines. You can make entries in 16 through 22 (stacker select/fetch overflow, space, and skip fields) on an OR line if required. If you leave these fields blank, the entries in these fields on the preceding line apply to this line.

#### **→ 8.2.5. Adding and Deleting Existing Indexed Sequential File Records and Deleting Transaction Buffers (Columns 16 through 18)**

You can add records to an existing indexed sequential input, output, or update file by placing an A in column 66 of the file description specifications form on the line that defines the file to be added to and by placing ADD in columns 16 through 18 on this form on the line that specified the record type (H, D, T, or E in column 15) you want to add. If an OR line follows an ADD line, the ADD also applies to that line.

You can delete records from an existing IMS indexed file by placing DEL in columns 16 through 18. The deleted record is marked with an X'FF' in the first byte of data. (Data records should never contain an X'FF' in the first byte.) Because deleted records physically remain in the file, the number of records logically deleted is recorded in the (filename)T field in the DTF module. Refer to the consolidated data management macroinstructions user guide programmer reference, UP-9979 (current version). A record must be read prior to deletion.

→ To delete a transaction buffer area (TBA) from an IMS action program, place DEL in columns 16 through 18.

### 8.2.6. Space (Columns 17 and 18)

You use these fields to control the spacing of printed reports (printer or line counter files). The before field (column 17) specifies the number of lines to be spaced before printing, and the after field (column 18) specifies the number of lines to be spaced after printing.

If you do not want to space before or after printing, enter a 0 in the appropriate field or leave the field blank.

If you want to space before or after printing, enter the number of lines (1 through 9) that you want spaced in the appropriate field. The maximum number of lines you can specify to be spaced depends upon the type of printer subsystem you are using.

If you specify both spacing and skipping for the same line, the operations are performed in the following order:

Skip before

Space before

Print

Skip after

Space after

If possible, you should specify space after rather than space before because the latter tends to slow printing.

### 8.2.7. Skip (Columns 19 through 22)

In all modes other than the IBM System/3 mode, you use these fields to specify the number of the channel on the printer form control tape that you want to skip to before or after printing. The before field (columns 19 and 20) specifies the number of the channel to be skipped to before printing and the after field (columns 21 and 22) specifies the number of the channel to be skipped to after printing.

Since more than one type of printer subsystem can be used, the following convention should be used to maintain device independence for programs that are to be compiled in the SPERRY UNIVAC OS/3 mode: only channels 01 through 07 should be entered in these fields and, of these, channel 01 should be used for forms overflow and channel 07 for home paper. The channel numbers that are specified will be automatically translated to the appropriate channel numbers for the particular printer subsystem. Channels 8 through 13 may be used; however, their use is not recommended. If they are used, they will also be automatically translated. This translation will also take place for programs that are to be compiled in the SPERRY UNIVAC 9200/9300 mode or the IBM 360/20 mode.

The channel numbers that can be specified for each compilation mode and the channel numbers that they are translated to for the various printer subsystems are shown in Table 8-1.

If you specify a channel number in this field, remember that the equivalent channel number (or the one that it is translated to) must be specified in a //ΔVFB statement in your job control stream or punched on the carriage control tape for the printer subsystem that is being used.

Table 8-1. Skip Channel Number Translation

Channel Number Specified			Channel Number Used		
SPERRY UNIVAC OS/3 Mode	SPERRY UNIVAC 92/9300 Mode	IBM 360/20 Mode	0768 Printer Subsystem	0770 Printer Subsystem	0773 Printer Subsystem
01	01	12	09 (forms overflow)	09 <sup>②</sup> (forms overflow)	01 (forms overflow)
02	02	02	02	02	02
03	03	03	03	03	03
04	04	04	04	04	04
05	05	05	05	05	05
06	06	06	06	06	06
07	07	01	14 <sup>①</sup> (home paper) 15 <sup>①</sup> (home paper)	07 <sup>②</sup> (home paper)	07 (home paper)
08	-	08,09	08	02	02
09	-	-	09	09 (forms overflow)	01 (forms overflow)
10	-	10	10	10	03
11	-	11	11	11	04
12	-	07	12	12	01 (forms overflow)
13	-	-	13	13	05

① There are two home paper channels for the 0768 printer subsystem. If channel 14 is used, the line spacing will be 6 lines per inch; if channel 15 is used the line spacing will be 8 lines per inch. If channel 07 is specified for a SPERRY UNIVAC OS/3 or 9200/9300 mode program or channel 01 for an IBM 360/20 mode program, the printer will skip to whichever channel (14 or 15) is punched on the carriage control tape or specified in the //ΔVFB statement in the job control stream.

② Channel 07 must always be punched for home paper and channel 09 must always be punched for forms overflow on the carriage control tape for the 0770 printer subsystem.



In the IBM System/3 mode, you use this field to specify the specific line on a printable form to be skipped to. Since this field is a 2-digit entry, the following convention is used for line numbers greater than 99: A0 through A9 is equivalent to lines 100 through 109 and B0 through B2 is equivalent to lines 110 through 112. If the skip code entry is less than the previous entry, the result is a skip to a new page without setting the overflow indicator on. If the entry is greater than the overflow line but less than the total number of lines specified for the printable form, the overflow indicator is turned on and remains on until all overflow lines are printed.

### 8.2.8. Output Indicators - Records (Columns 23 through 31)

You use these fields to specify the indicators you want tested to determine whether the record described on this line is to be written. Up to three indicators can be specified on a line. You specify the indicators in columns 24 and 25, 27 and 28, and 30 and 31, and you specify whether or not an indicator must be on or off in columns 23, 26, or 29 (N=NOT). You leave these fields blank if you want a record always to be written.

If an indicator must be on before a record can be written, you specify this by placing that indicator in any of the indicator fields on this line. If an indicator must be off, you specify this by placing an N in one of the N=NOT fields and the indicator in the adjacent field. You can specify a general indicator (O1 through 99), a control level indicator (LO through L9), the last record indicator (LR), a halt indicator (HO through H9), the matching record indicator (MR), the first page indicator (1P), the overflow indicator (OA through OG, OV) assigned to the file, or an external indicator (U1 through U8), or a function key indicator (KA through KN and KP through KW).

The 1P indicator is prohibited for workstation terminal files.

If you have two or more records that you want written when the same conditions are present, you can save time by specifying the conditions on the first record identification line and then entering an asterisk (\*) in column 25 on the succeeding lines. This indicates to RPG II that the conditions on the preceding line must be met before the record on the \* line can be written. If you use an \*, it must always be the same type as the preceding line. This means that if the preceding line is a heading line (H in column 15), a detail line (D in column 15), a total line (T in column 15), or an exception line (E in column 15), the \* line that follows must be the same type of line.

If you have an AND relationship, you must not specify an overflow indicator (OA through OG, OV), stacker select, or fetch overflow on the AND line.

If you have a total or exception line (T or E in column 15), you must not specify the first page indicator (1P) on that line.

If you have an exception line (E in column 15), you must not specify an overflow indicator for that line.

### 8.3. FIELD DESCRIPTION AND CONTROL ENTRIES (COLUMNS 23 THROUGH 70)

In the following subsections you'll see what each field description and control field is used for and how the entries in these fields affect your program. The field description and control entries for a given record are always written on the lines immediately following the output file identification and control entries for that record. For example, if line 010 contains the output file identification and control entries for a given record, you would write the first field description and control entries for this record on line 020, the second on line 030, and so on. When you use the field description and control fields, columns 7 through 22 on these lines must be left blank, and AND/OR relationships are not permitted.

#### 8.3.1. Output Indicators - Fields (Columns 23 through 31)

You use these fields to specify the indicators you want tested to determine whether the field described on this line is to be written. Up to three indicators can be specified on a line. You specify the indicators in columns 24 and 25, 27 and 28, and 30 and 31, and you specify whether or not an indicator must be on or off in columns 23, 26, or 29 (N=NOT). You can specify an asterisk (\*) in the same way as with record output indicators (8.2.8).

You leave these fields blank if you want a field always to be written.

If an indicator must be on before a field can be written, you specify this by placing that indicator in any of the indicator fields on this line. If an indicator must be off, you specify this by placing an N in one of the N=NOT fields and the indicator in an adjacent field. You can specify a general indicator (01 through 99), a control level indicator (L0 through L9), the last record indicator (LR), the matching record indicator (MR), a halt indicator (H0 through H9), the first page indicator (1P), an overflow indicator (OA through OG, OV), an external indicator (U1 through U8), or a function key indicator (KA through KN and KP through KW). If you have a field associated with a total or exception record (T or E in column 15 of the output file identification and control line for the associated record) you must not specify the first page indicator (1P) for the field. The 1P indicator is prohibited for workstation terminal files.

If two or more fields have the same conditions, you can use an asterisk (\*) in column 25. You cannot use an asterisk with the field names PAGE or PAGE1 through PAGE7. For further information, see 14.4.2.

#### 8.3.2. Field Name (Columns 32 through 37)

You use this field to identify the field in the output record that you want to be written, punched, or printed. The conditions that you specified for both the field and the record it is contained in must be satisfied before output takes place. This entry must be left-justified.

If the field is a constant or literal that you specified in columns 45 through 70, leave this field blank.

If you are specifying a field, table, or array name, enter the name in the form of one to six alphanumeric characters. The name you specify must be that of a field that was previously defined on the input format specifications form or the calculation specifications form, or it must be the name of a table or array that was previously defined on the file extension specifications form. When an array name is specified, the entire array is written. When a table name is specified, the element last found by a LOKUP operation is written. If no LOKUP operation was performed, the first element of the table is written. If an unsuccessful LOKUP operation follows a successful one, the element found by the successful LOKUP operation is written.

If you are specifying an array element name, enter the name in the form: array-name, index. The index is the number of the element in the array. When you specify an array element name, that element is written. You can also use the RPG II special field names PAGE, PAGE1 through PAGE7, \*ERROR, \*PLACE, UDATE, D?TE, UDAY, UMONTH, or UYEAR in this field. PAGE, and PAGE1 through PAGE7 are used to control automatic sequential page numbering. \*ERROR is used for program testing (debugging). \*PLACE is used to cause previously defined fields in an output record to be repeated in that output record. UDATE or D?TE allows you to retrieve the system date in the form mmddy (mm=month, dd=day, yy=year). UDAY allows you to retrieve the system date day (dd). UMONTH allows you to retrieve the system date month (mm). UYEAR allows you to retrieve the system date year (yy). We will cover the use of these field names when we discuss printing techniques and program testing.

### 8.3.3. Edit Codes (Column 38)

You use this field to specify edit codes that will automatically suppress leading zeros, punctuate (insert commas or slashes), or place a negative value indicator (- or CR) in a numeric field.

If the field is an alphanumeric field or you have specified a constant or edit word in columns 45 through 70, leave this field blank.

If you want to edit a numeric field, enter the appropriate edit code: 1 through 4, A through D, J through M, or X through Z. The effect of these codes is summarized on the output format specifications form. We will cover the use of these codes when we discuss printing techniques.

### 8.3.4. Blank After (Column 39)

You use this field to cause the contents of a field to be reset to zeros or blanks after the field is placed in the output record.

If you do not want a field reset, if the field is one of the RPG II special fields, or if you have specified a constant in columns 45 through 70, leave this field blank.

If you want to reset a numeric field to zeros or an alphanumeric field to blanks, enter a B in this field.

**8.3.5. End Position in Output Record (Columns 40 through 43)**

You use this field to specify the location in the output record where the rightmost character of a field, constant, or literal is to be placed. This entry must be right-justified; leading zeros may be omitted. This entry can range from 0001 through 9999. For workstation fields, this refers to the end position of the field, not the end position of the field as shown on the screen.

If the field is edited, the end position you specify must allow for any additional characters that are added by the edit operation. If you do not allow for this, your output fields may overlap.

If the field is an unpacked numeric field and you want to write it in packed decimal format, you must specify a P in column 44 of this form. In addition, you must consider that when an unpacked field is packed, the size of the field is reduced and therefore the size that you specify for the output field must be smaller than that of the unpacked field. You can calculate the size of a field after it is packed by using the following formula:

$$\text{packed field length} = \frac{\text{unpacked field length} + 1}{2}$$

When this formula is used, round up the quotient to the next highest whole number if it is not a whole number.

For example, assume that an unpacked numeric field is specified as having a length of six bytes (character positions) in columns 44 through 51 of the input format specifications form or columns 49 through 51 of the calculation specifications form. If you apply the formula, the length of the output field must be 4; that is  $\frac{6 + 1}{2} = \frac{7}{2} = 3.5$ , or 4 when rounded up.

If the field is a numeric field that is to be written with a preceding or following plus or minus sign (L or R in column 44), you must provide an additional position for the sign when you specify the end position for this field.

**8.3.6. Length of Screen Format Name (Columns 42 and 43)**

To use screen formats, you must name the format and specify the length of this name. The length is marked in these two columns with a Kn, where n is the length of the name. These two columns are used in conjunction with columns 45 through 54, where the format name for each record within a workstation terminal file is written.

**8.3.7. Data Format (Column 44)**

You use this field to specify the format of the data that is to be placed in the output record.

You must have this field blank if the field is an alphanumeric field or an unpacked numeric field, if the field is to be edited, or if you have specified a constant in columns 45 through 70.

If the field is a binary field, enter a B in this field.

If the field is to be written in packed decimal format, enter a P in this field.

If the field is a numeric field that is to be written with a preceding plus or minus sign, enter an L in this field.

If the field is a numeric field that is to be written with a following plus or minus sign, enter an R in this field.

### 8.3.8. Constant (Columns 45 through 70)

You use this field to specify an alphanumeric or hexadecimal constant or literal value that you want to be placed in the output record.

If you want to place an alphanumeric constant in the output record, enter the constant in this field in the form 'alphanumeric characters'. You can specify a maximum of 24 alphanumeric characters, including blanks. Any unspecified bytes of the field are set to blanks.

If you want to place a hexadecimal constant in the output record, enter the constant in this field in the form X 'hexadecimal digits'. Since each hexadecimal digit represents one-half byte of information, the constant you specify should consist of an even number of digits so that full bytes of information are produced. If the constant consists of an odd number of digits, full bytes of information are produced; however, the byte that the leftmost digit of the constant is placed in will contain zeros in the high-order bit positions. To illustrate this, consider the following examples:

<u>Constant Specified</u>	<u>Bytes Produced</u>
X'AAA'	00001010 10101010
X'AAAA'	10101010 10101010

An apostrophe in a constant must be represented by two apostrophes.

### 8.3.9. Edit Word (Columns 45 through 70)

You can usually handle the editing of a numeric field by specifying one of the edit codes in column 38. There are some cases, however, where special editing is needed that is not provided by any of the edit codes. You can handle these cases by specifying an edit word in this field that will provide you with the editing requirements you need.

If the field is an alphanumeric field or you have specified an edit code in column 38, leave this field blank.

If the field is a numeric field that requires special editing, enter a 1- to 24-character edit word in the form 'edit word' in this field. We will cover the use of edit words when we discuss printing techniques. See 14.3.2.



### 8.3.10. Format Name (Columns 45 through 54)

A screen format name must be given to each record within a workstation terminal file. The name is written in columns 45 through 54. The entire 10 columns need not be filled, but all entries must be left-justified. Whatever name you choose, the number of characters used must be recorded in columns 42 and 43. Indicators cannot condition the line containing the format name.

## 8.4. EXAMPLES OF ENTRIES ON THE OUTPUT FORMAT SPECIFICATIONS FORM

Figure 8-2 shows you examples of typical entries on the output format specifications form.

Explanation of entries in Figure 8-2:

<u>Line Number</u>	<u>Explanation</u>
010 and 020	The output file is named NEWMAS (NEWMAS in columns 7 through 14 on line 010). The record is an exception record (E in column 15 on line 010) that is written on the file when the MR indicator is off (N in column 23 and MR in columns 24 and 25 on line 010) and indicator 01 is on (01 in columns 27 and 28 on line 010). When this record is written, the field named INREC (INREC in columns 32 through 37 on line 020) is included in the record. The rightmost character of the field is to be located in record position 80 (80 in columns 40 through 43 on line 020).
030-060	A detail record (D in column 15 on line 030) is written on the output file named NEWMAS when the MR indicator is on (MR in columns 24 and 25 on line 030) and indicator 09 is off (N in column 26 and 09 in columns 27 and 28 on line 030). The fields named CODE, PARNUM, and ITEM are included when the record is written (CODE in columns 32 through 37 on line 040, PARNUM in columns 32 through 37 on line 050, and ITEM in columns 32 through 37 on line 060). The rightmost character of the CODE field is to be located in record position 1 (1 in columns 40 through 43 on line 040), the rightmost character of the PARNUM field is to be located in record position 7 (7 in columns 40 through 43 on line 050), and the rightmost character of the ITEM field is to be located in record position 24 (24 in columns 40 through 43 on line 060).
070-090	The output file is named DAILYAC (DAILYAC in columns 7 through 14 on line 070). The file is a printer file because spacing and skipping are indicated in columns 17 through 22. A heading record (H in column 15 on line 070) is to be printed when the 1P (first page) indicator is on (1P in columns 24 and 25 on line 070) or the OF (overflow) indicator is on (OF in columns 24 and 25 on line 080). The printer form skips to channel 07 before printing (07 in columns 19 and 20 on line 070) and spaces two lines after printing (2 in column 18 on line 070). The heading record consists of the constant DAILY△REPORT ('DAILY△REPORT' in columns 45 through 70 on line 090). The rightmost character in this constant is to be placed in print position 64 on the form (64 in columns 40 through 43 on line 090).

<u>Line Number</u>	<u>Explanation</u>
100-130	A detail record (D in column 15 on line 100) is printed when the MR indicator is on (MR in columns 24 and 25 on line 100) and indicator 09 is off (N in column 26 and 09 in columns 27 and 28 on line 100). The printer form is spaced one line after printing (1 in column 18 on line 100). The field named QUANT (QUANT in columns 32 through 37 on line 110) is always included when this record is printed (columns 23 through 31 are blank on line 110), the rightmost character of the field is to be printed in print position 7 (7 in columns 40 through 43 on line 110), and zeros are to be inserted in the field after it is placed in the output record (B in column 39 on line 110). The field named NPRI (NPRI in columns 32 through 37 on line 120) is included only when indicator 10 is on (10 in columns 24 and 25 on line 120), the rightmost character of the field is to be printed in print position 22 (22 in columns 40 through 43 on line 120), and leading zeros are suppressed when the field is printed (Z in columns 38 on line 120). The field named DPRI (DPRI in columns 32 through 37 on line 130) is included only when indicator 11 is on (11 in columns 24 and 25 on line 130), the rightmost character of the field is to be printed in print position 31, and leading zeros are suppressed when the field is printed (Z in column 38 on line 130).
140 and 150	A total record (T in column 15 on line 140) is to be printed when the L1 indicator is on (L1 in columns 24 and 25 on line 140). The printer form skips to channel 02 before printing (02 in columns 19 and 20 on line 140) and spaces two lines after printing (2 in column 18 on line 140). The total record consists of the constant TOTAL△SALES ('TOTAL△SALES' in columns 45 through 70 on line 150). The rightmost character of this constant (S) is to be placed in print position 17 on the form (17 in columns 40 through 43 on line 150).
160 and 170	A total record (T in column 15 on line 160) is to be printed when the L1 indicator is on (L1 in columns 24 and 25 on line 160). The field named TSLS (TSLS in columns 32 through 37 on line 170) is always included when this record is printed (columns 23 through 31 are blank on line 170), the rightmost character in the field is to be printed in print position 13 (13 in columns 40 through 42 on line 170), and the field is to be edited by using an edit word ('△△△,△\$0.△△' in columns 45 through 70 on line 170).



## 9. File Extension Specifications Form

### 9.1. GENERAL DESCRIPTION

The file extension specifications form (Figure 9-1) is not required for every program. It is required if you use tables or arrays in your program. It is also required if you use a chaining file, record address file, or a tag (ADDROUT) file to control the retrieval of records from a data file because the RPG II compiler needs additional information in these cases. If you use tables or arrays, you use this form to specify the name of each table or array, the number of elements in each record, the size of the table or array, and the length, format, number of decimal positions, and sequence of each element. If you use a chaining file, record address file, or tag (ADDROUT) file, you use this form to relate the controlling file to the data file.

### 9.2. FORM ENTRIES

In the following subsections you'll see what each field on the form is used for and how the entries in these fields affect your program.

#### 9.2.1. Record Sequence of Chaining File (Columns 7 and 8)

You use this field to specify the sequence of the records in the chaining file when you use input chaining (C1 through C9 chaining indicators). The record sequence that you specify must be the same as that specified for the chaining file in columns 15 and 16 on the input format specifications form. If you use the CHAIN operation, this field is not used. In SPERRY UNIVAC 9200/9300 and IBM 360/20 mode, this field must be blank. ←

#### 9.2.2. Number of the Chaining Field (Columns 9 and 10)

You use this field to specify the number of the chaining field when you use input chaining (C1 through C9 chaining indicators). The number of the chaining field that you specify must be the same as that specified for the chaining field in columns 61 and 62 on the input format specifications form. If you use the CHAIN operation, this field is not used.



### 9.2.3. From File Name (Columns 11 through 18)

You use this field in conjunction with the to-file-name field (columns 19 through 26) to specify the relationship between two or more files in your program. The entries in these fields are summarized in Table 9-1.

If you intend to load a table or array at compilation time, or if you intend to use the input format or calculation specifications form to create an array, leave this field blank.

If you intend to load a table or array at execution time, enter the name of the table or array file in this field.

If you are using a chaining file, record address, or tag (ADDRROUT) file to control the retrieval of records from a data file, enter the name of the controlling file in this field.

All entries in this field must be left-justified. The file name that you enter must be the same as that specified in columns 7 through 13 of the file description specifications form.

### 9.2.4. To File Name (Columns 19 through 26)

You use this field in conjunction with the from-file-name field (columns 11 through 18) to specify the name of the data file that is to be processed by a chaining, record address, or tag (ADDRROUT) file or to specify the name of the output file that a table or array file is to be written on at end-of-job. The entries in these fields are summarized in Table 9-1.

If you intend to use the input format or calculation specifications form to create an array or you do not want a table or array to be written on an output file at end-of-job, leave this field blank.

If you are using a chaining, record address, or tag (ADDRROUT) file to control the retrieval of records from a data file, enter the name of the data file in this field.

If you want to write a table or array file on an output file at end-of-job, enter the name of the output file in this field. A unique output file name must be specified for each table or array file that you want to write at end-of-job. A table or array output file must be specified as fixed format file. All entries in this field must be left-justified. The file name that you enter must be the same as that specified in columns 7 through 13 of the file description specifications form.

Table 9—1. Summary of From File Name and To File Name Entries

File Type	From File Name (Columns 11 through 18)	To File Name (Columns 19 through 26)
Array created by input format or calculation specifications form	Blank	Blank
Table or array loaded at execution time	The name of the file that contains the table or array	The name of the output file that the table or array is written on at end-of-job. If the table or array is not to be written, this entry must be blank.
Table or array loaded at compilation time	Blank	The name of the output file that the table or array is written on at end-of-job. If the table or array is not to be written, this entry must be blank.
Chaining file	The name of the chaining file that contains the chaining field	The name of the data file (chained file) that is processed by the chaining file
Record address file	The name of the record address file that contains the keys or record addresses	The name of the data file that is processed by the record address file
Tag (ADDROUT) file	The name of the tag file that contains the record addresses	The name of the data file that is processed by the tag (ADDROUT) file

### 9.2.5. Table or Array Name (Columns 27 through 32)

You use this field to specify the name of a table or array that is used in your program. A table or array file can consist of a single table or array (solitary format) or two tables or arrays (alternating format). The input records in solitary format contain the individual elements of a table or array. In alternating format, they contain the individual elements of two tables or arrays, and the elements are alternated. In this case, an element from the first table or array is always followed by an element from the second table or array in the input records.

If a table or array file is in solitary format, you use this field to specify the name of a single table or array.

If the table or array file is in alternating format, you use this field to specify the name of the first table or array. The name of the second table or array is specified in columns 46 through 51.

If you are specifying an array name for an array in solitary format or the name of the first array in alternating format, enter the array name in the form of one to six alphanumeric characters. An array name cannot begin with TAB. If you are specifying a table name for a table in solitary format or the name of the first table in alternating format, enter the table name in the form TABaaa, where aaa is any combination of alphanumeric characters.

All entries in this field must be left-justified.

The total number of tables and/or arrays (including alternate tables and arrays) in each program cannot exceed 325. ←

#### **9.2.6. Number of Entries per Record (Columns 33 through 35)**

You use this field to specify the number of table or array elements that are contained in each record. If you intend to use the input format or calculation specifications form to create an array, leave this field blank. If this field is blank, the array cannot be written on an output file at end-of-job; consequently, the to-file-name field (columns 19 through 26) must also be blank in this case.

If the table or array is not created by the input format or calculation specifications form, enter the number of table or array elements (001 through 999) in each record. Each record except the last must contain exactly the number of table or array elements that you specified.

All entries in this field must be right-justified; leading zeros may be omitted.

#### **9.2.7. Number of Entries per Table or Array (Columns 36 through 39)**

You use this field to specify the maximum number of elements (0001 through 9999) that can be contained in a table or array. If you anticipate expansion, you can specify a greater number of elements than is actually required. If a table contains fewer elements than the maximum specified, the unused space is filled with blanks or zeros as appropriate.

All entries in this field must be right-justified; leading zeros may be omitted.

#### **9.2.8. Length of Entry (Columns 40 through 42)**

You use this field to specify the length of each table or array element (001 through 256). The maximum length for numeric elements is 15 digits and the maximum length for alphanumeric elements is 256 characters.

If the format of the table or array elements is binary (B in column 43), the entry in this field must be four or nine digits.

If the table or array elements are preceded or followed by a sign (L or R in column 43), the length must include one additional position for the sign.

All entries in this field must be right-justified; leading zeros may be omitted.

### **9.2.9. Data Format (Column 43)**

You use this field to specify the format of the data in the table or array elements.

If the table or array elements are in alphanumeric format or unpacked decimal format, or if you are using the input format or calculation specifications form to create an array, leave this field blank.

If the table or array elements are in binary format, enter a B in this field.

If the table or array elements are in packed decimal format, enter a P in this field.

If the table or array elements are numeric and are preceded by a plus or minus sign, enter an L in this field.

If the table or array elements are numeric and are followed by a plus or minus sign, enter an R in this field.

### **9.2.10. Decimal Positions (Column 44)**

You use this field to specify the number of decimal positions in the table or array elements.

If the table or array elements are alphanumeric, leave this field blank.

If the table or array elements are numeric, you must enter the number of decimal positions (0 through 9) in this field. This entry is required even if the numeric elements do not contain any decimal positions; that is, if the table or array elements are in unpacked decimal format, or if B, P, L, or R is specified in column 43, an entry must be made in this field. The number of decimal positions you specify must be less than or equal to the length specified in columns 40 through 42.

### **9.2.11. Sequence (Column 45)**

You use this field to specify the sequence of data in a table or array.

If the sequence of data in the table or array is not critical to your program, leave this field blank.

If the data in the table or array is in ascending order, enter an A in this field.

If the data in the table or array is in descending order, enter a D in this field.

If A or D is specified, sequence checking is performed when the table or array is loaded. It is not performed if you use the input format or calculation specifications form to create an array.

If you specify a LOKUP operation with high or low resulting indicators on the calculation specifications form, you must enter an A or D in this field.

### 9.2.12. Table or Array Name (Columns 46 through 51)

You use this field to specify the name of the second table or array when alternating format is used. The name of the first table or array is specified in columns 27 through 32.

If a table or array is not specified or the table or array is in solitary format, leave this field blank.

If you are specifying the array name for the second array in alternating format, enter the array name in the form of one to six alphanumeric characters. An array name cannot begin with TAB.

If you are specifying the table name for the second table in alternating format, enter the table name in the form TABaaa, where aaa is any combination of alphanumeric characters.

All entries in this field must be left-justified.

### 9.2.13. Length of Entry (Columns 52 through 54)

You use this field to specify the length of each element (001 through 256) in the second table or array when alternating format is used. The maximum length for numeric elements is 15 digits and the maximum length for alphanumeric elements is 256 characters.

If the format of the table or array elements is binary (B in column 55), the entry in this field must be four or nine digits.

If the table or array elements are preceded or followed by a plus or minus sign (L or R in column 55), the length must include one additional position for the sign.

All entries in this field must be right-justified; leading zeros may be omitted.

### 9.2.14. Data Format (Column 55)

You use this field to specify the data in the elements of the second table or array when alternating format is used.

If the table or array elements are in alphanumeric format or unpacked decimal format, or if you are using the input format or calculation specifications form to create an array, leave this field blank.

If the table or array elements are in binary format, enter a B in this field.

If the table or array elements are in packed decimal format, enter a P in this field.

If the table or array elements are numeric and are preceded by a plus or minus sign, enter an L in this field.

If the table or array elements are followed by a plus or minus sign, enter an R in this field.

### **9.2.15. Decimal Positions (Column 56)**

You use this field to specify the number of decimal positions in the elements in the second table or array when alternating format is used. If the table or array elements are alphanumeric, leave this field blank.

If the table or array elements are numeric, you must enter the number of decimal positions (0 through 9) in this field. This entry is required even if the numeric elements do not contain any decimal positions; that is, if the table or array elements are in unpacked decimal format or P, B, L, or R is specified in column 55, an entry must be made in this field. The number of decimal positions you specify must be less than or equal to the length specified in columns 52 through 54.

### **9.2.16. Sequence (Column 57)**

You use this field to specify the sequence of the data in the second table or array when alternating format is used. The entry you make in this field must be the same as that specified in column 45.

If column 45 is blank, leave this field blank. If the data in the table or array is in ascending order, enter an A in this field.

If the data in the table or array is in descending order, enter a D in this field.

If A or D is specified, sequence checking is performed when the table or array is loaded. It is not performed if you use an input format or calculation specifications form to create an array.

If you specify a LOKUP operation with high or low resulting indicators on the calculation specifications form, you must specify A or D in this field.

### **9.2.17. Comments (Columns 58 through 74)**

You can use this field to make comments that relate to the entries on the individual lines on this form. When your program is compiled, these comments will have no effect on the compilation process; however, they will be printed on the source program listing that is produced.

## **9.3. EXAMPLES OF ENTRIES ON THE FILE EXTENSION SPECIFICATIONS FORM**

Figure 9-2 shows you examples of typical entries on the file extension specifications form.



FILE EXTENSION SPECIFICATIONS

PAGE NO	FORM TYPE	RECORD SEQUENCE OF CHAINING FILE		TO FILE NAME	TABLE OR ARRAY NAME	NUMBER OF ENTRIES PER RECORD	NUMBER OF ENTRIES PER TABLE OR ARRAY	LENGTH OF ENTRY	P B L R DATA FORMAT	DECIMAL POSITIONS	A D SEQUENCE	ALTERNATING FORMAT						
		01 99 or AA ZZ	C1 C3									TABLE OR ARRAY NAME	LENGTH OF ENTRY	P B L R DATA FORMAT	DECIMAL POSITIONS	A D SEQUENCE		
1 2 3 4 5 6	7 8 9 10 11 12 13 14 15 16 17 18 19	20 21 22 23 24 25 26 27	28 29 30 31 32 33 34 35 36 37 38 39 40 41 42	43 44 45 46	47 48 49 50 51 52 53 54	55 56 57												
010	E	AA	C1	INPUT	MASTER													
020	E			RAF	UPDATE													
030	E			TABLE	OUTBLE	TAB	100	20	200	10	0	A	T	A	B	2010	10	2
040	E					ARR			100	5	2							
050	E				PRINT	DOT		8	200	4		A	A	P	C		5	2
060	E			ARFILE		DOT		8	200	4		A	A	P	C		5	2

Figure 9-2. Examples of Entries on File Extension Specifications Form

Explanation of entries in Figure 9-2:

Line Number Explanation

- 010 The file named INPUT (INPUT in columns 11 through 18) is used to chain the file named MASTER (MASTER in columns 19 through 26). INPUT is the chaining file and MASTER is the chained file. The records on the file named MASTER are selected for processing by a chaining field that is contained in the records on the file named INPUT. The number of the chaining field is C1 (C1 in columns 9 and 10). This field is the one defined on the input format specifications form that contains C1 in columns 61 and 62. The records on the chaining file, INPUT, are not arranged in any special sequence (AA in columns 7 and 8).
- 020 The file named RAF (RAF in columns 11 through 18) is the record address file that is used to supply the addresses of the records that are to be processed from the file named UPDATE (UPDATE in columns 19 through 26).

Line Number Explanation

- 030 The file named TABLE (TABLE in columns 11 through 18) is a table file that is loaded at execution time and contains two tables in alternating format (TAB100 in columns 27 through 32 and TAB200 in columns 46 through 51). Each record contains 20 elements (20 in columns 33 through 35), and there are 200 elements in each table (200 in columns 36 through 39). Each element in TAB100 is 10 characters in length (10 in columns 40 through 42), the elements are numeric and have no decimal positions (0 in column 44), and the records are in ascending sequence (A in column 45). Each element in TAB200 is 10 characters in length (10 in columns 52 through 54), and the elements are numeric with 2 decimal positions (2 in column 56). After the tables are updated, they will be written on the file named OUTBLE (OUTBLE in columns 19 through 26).
- 040 The array ARR1 (ARR1 in columns 27 through 32) is created on the input format specifications form or the calculation specifications form, there are 100 elements in the array (100 in columns 36 through 39), each element is 5 characters in length (5 in columns 40 through 42), and the elements are numeric with 2 decimal positions (2 in column 44).
- 050 DOT and APC are arrays in alternating formats that are loaded at compilation time (DOT in columns 27 through 32 and APC in columns 46 through 51). Each record contains 8 elements (8 in columns 33 through 35), there are 200 elements in each array (200 in columns 36 through 39), each element in DOT is 4 characters in length (4 in columns 40 through 42) and alphanumeric (column 44 is blank), and DOT is in ascending sequence (A in column 45). Each element in APC is five characters in length (5 in columns 52 through 54), and the elements are numeric with two decimal positions (2 in column 56). These arrays are to be written on the file named PRINT at end-of-job (PRINT in columns 19 through 26).
- 060 The file named ARFILE (ARFILE in columns 11 through 18) is an array file that is loaded at execution time and contains two arrays in alternating format (DOT in columns 27 through 32 and APC in columns 46 through 51). Each record contains 8 elements (8 in columns 33 through 35), and there are 200 elements in each array (200 in columns 36 through 39). Each element in DOT is four characters in length (4 in columns 40 through 42) and alphanumeric (column 44 is blank), and DOT is in ascending sequence (A in column 45). Each element in APC is five characters in length (5 in columns 52 through 54) and the elements are numeric with two decimal positions (2 in column 56).

## 10. Line Counter Specifications Form

### 10.1. GENERAL DESCRIPTION

The line counter specifications form (Figure 10-1) is not required for every program. It is required if you want to store a report on an intermediate tape or disk file so that it can be printed at some more convenient time when your system is not being used to its fullest capacity. You use this form, in this case, to place in the intermediate file the printer carriage control information (home paper, forms overflow, and line skipping information) related to the lines on the report. When the intermediate file is ultimately printed (using the file processing utility as described in the data utilities user guide), the carriage control information that was placed in the file will cause the printer carriage to perform as if the report were being printed at the time your program was executed.

In the IBM System/3 mode, this form is used to indicate the length of the printer/output form and at what line overflow occurs for a printer output file. If a line counter specification is not present for a printer file, the form length and overflow line are determined from the // VFB statement in the job control stream.

### 10.2. FORM ENTRIES FOR ALL MODES OTHER THAN THE IBM SYSTEM/3 MODE

In the following subsections you'll see what each field on the form is used for and how the entries in these fields affect your program.

#### 10.2.1. File Name (Columns 7 through 14)

You use this field to specify the name of the intermediate output file you want your report written on. The name you specify must be the same as that specified for this file in columns 7 through 13 on the file description specifications form. This entry must be left-justified.

FILE EXTENSION SPECIFICATIONS

PAGE NO	FORM TYPE	LINE NO	RECORD SEQUENCE OF CHAINING FILE		TO FILE NAME	TABLE OR ARRAY NAME	NUMBER OF ENTRIES PER RECORD	NUMBER OF ENTRIES PER TABLE OR ARRAY	LENGTH OF ENTRY	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	A/D SEQUENCE	ALTERNATING FORMAT		COMMENTS	PROGRAM IDENTIFICATION																		
			01-98 or AA-ZZ	C1-C9									TABLE OR ARRAY NAME	LENGTH OF ENTRY																				
1	2	3	5	8	7	8	9	10	11	18	19	26	27	32	33	35	36	38	40	42	43	44	45	46	51	52	54	55	56	57	58	74	75	80
		0, 1	E																															
		0, 2	E																															
		0, 3	E																															
		0, 4	E																															
		0, 5	E																															

LINE COUNTER SPECIFICATIONS

PAGE NO	FORM TYPE	LINE NO	FILE NAME	1		2		3		4		5		6		7		8		9		10		11		12		PROGRAM IDENTIFICATION																													
				LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO	LINE NO	CH NO																														
1	2	3	5	6	14	15	17	18	19	20	22	23	24	25	27	28	29	30	32	33	34	35	37	38	39	40	42	43	44	45	47	48	49	50	52	53	54	55	57	58	59	60	62	63	64	65	67	68	69	70	72	73	74	75	80		
		0, 1	L																																																						
		0, 2	L																																																						
		0, 3	L																																																						
		0, 4	L																																																						
		0, 5	L																																																						

Figure 10-1. Line Counter Specifications Form

### **10.2.2. Line Number (Columns 15 through 17, 20 through 22, 25 through 27, ..., 70 through 72)**

You use this field to relate a line number on your report, as you have laid it out on the printer format chart, to a channel number that will cause the printer carriage to perform a line skipping, forms overflow, or home paper function for the report. This entry must be right-justified. Leading zeros may be omitted.

### **10.2.3. Channel Number (Columns 18 and 19, 23 and 24, 28 and 29, ..., 73 and 74)**

You use this field to specify the channel number that will cause the printer carriage to perform a line skipping, forms overflow, or home paper function. Channel 1 must be specified for the forms overflow function and channel 7 for the home paper function. All entries in this field must be right-justified. The leading zero may be omitted.

## **10.3. FORM ENTRIES FOR THE IBM SYSTEM/3 MODE**

In the following subsections you'll see how each field on the form is used in the IBM System/3 mode and how the entries in these fields affect your program in this mode.

### **10.3.1. File Name (Columns 7 through 14)**

You use this field to specify the name of the printer output file. The name you specify must be the same as that specified for this file in columns 7 through 13 on the file description specifications form. This entry must be left-justified.

### **10.3.2. Line Number - Number of Lines per Page (Columns 15 through 17)**

You use this field to specify the exact number of lines per page on the printed report. This entry must be right-justified and may range from 12 through 112. Leading zeros may be omitted. If less than 12 lines or more than 112 is specified, the minimum limit (12 lines) will be assumed.

### **10.3.3. Channel Number - Form Length Indicator (Columns 18 and 19)**

You use this field to indicate that the preceding entry (columns 15 through 17) is the form length. This is done by entering FL in this field.

### **10.3.4. Line Number - Overflow Line (Columns 20 through 22)**

You use this field to specify the line number associated with forms overflow; that is, the line on the printed page where forms overflow is to occur. This entry must be right-justified and may range from 1 to 112.

**10.3.5. Channel Number - Overflow Line Indicator (Columns 23 and 24)**

You use this field to indicate that the preceding entry (columns 20 through 22) is the overflow line. This is done by entering OL in this field.

**10.3.6. Line Number (Columns 25 through 27, ..., 70 through 72) and Channel Number (Columns 28 and 29, ..., 73 and 74)**

These fields are not used in the IBM System/3 mode.

**10.4. EXAMPLES OF ENTRIES ON THE LINE COUNTER SPECIFICATIONS FORM**

Figure 10-2 shows you examples of typical entries on the line counter specifications form.

**LINE COUNTER SPECIFICATIONS**

PAGE NO.	FORM TYPE	LINE NO.	FILE NAME	1		2		3		4		5		6		7		8		9																					
				LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.																				
1	2	3	5	6	7	14	15	17	18	19	20	22	23	24	25	27	28	29	30	32	33	34	35	37	38	39	40	42	43	44	45	47	48	49	50	52	53	54	55	57	
		0,1,0	L	LINECNT		5	0	7	11	9	0	2	38	0	3	14	8	0	1																						
		0,2,0	L	DEFBOUT		1	0	0	7	15	9	0	1																												
		0,3,0	L	TRIMOD,FL		5	6	F	L	14	0	0	L																												
		0,4	L																																						
		0,5	L																																						

Figure 10-2. Examples of Entries on Line Counter Specifications Form

Explanation of entries in Figure 10-2:

Line Number    Explanation

010            The file named LINECNT (LINECNT in columns 7 through 14) is a line counter file. The home paper channel (07) corresponds to line 5 of the report form (07 in columns 18 and 19, and 5 in columns 15 through 17). Channel 02 corresponds to line 19 of the report form (02 in columns 23 and 24, and 19 in columns 20 through 22). Channel 03 corresponds to line 38 of the report form (03 in columns 28 and 29, and 38 in columns 25 through 27). The forms overflow channel (01) corresponds to line 48 of the report form (01 in columns 33 and 34, and 48 in columns 30 through 32).

Line Number Explanation

- 020            The file named DEFOUT (DEFOUT in columns 7 through 14) is a line counter file. The home paper channel (07) corresponds to line 10 of the report form (07 in columns 18 and 19, and 10 in columns 15 through 17). The forms overflow channel (01) corresponds to line 59 on the report form (01 in columns 23 and 24, and 59 in columns 20 through 22).
- 030            IBM System/3 mode: The file named TRIMODFL (TRIMODFL in columns 7 through 14) is a printer file. The number of lines per page is 56 (56 in columns 15 through 17 and FL in columns 18 and 19). The overflow line is line 40 (40 in columns 20 through 22 and OL in columns 23 and 24).





# 11. Telecommunications Specifications Form

## 11.1. GENERAL DESCRIPTION

The telecommunications form (Figure 11-1) is not required for every program. You use this form to specify which files in your program are data communications files, whether the remote terminal is connected to a multiplexer, how the files are used (data on the file is to be received from or transmitted to a remote terminal), whether any hexadecimal character can be transmitted as data, how the connection is to be made when a remote terminal is connected on switched line, the type of remote terminal, what program indicator is set on if a transmission or reception error occurs, the amount of time allowed between terminal messages for your program to complete a program cycle, how the file is to be processed, the auxiliary device on a remote terminal that you want to use, and the terminal name.

The telecommunications specifications form should not be used for an IMS action program. All IMS telecommunications processing is handled by IMS.

## 11.2. FORM ENTRIES

In the following subsections you'll see what each field on the form is used for and how the entries in these fields affect your program.

### 11.2.1. File Name (Columns 7 through 13)

You use this field to specify the name of a data communications file from which your program is to receive data or to which it is to transmit data. The name that you specify must be the same name that you specified for this file in columns 7 through 13 of the file description specifications form.

If more than one interactive terminal is associated with the same file, the file name must be repeated on successive contiguous lines for each terminal. An asterisk (\*) can be placed in column 14 to indicate the file name is being repeated. If file names are repeated, columns 16 through 47, column 52, and columns 55 through 70 must be blank for all lines following the line that specifies the file name for the first time.

The file name must be unique for batch terminals, auxiliary devices, and output files.



### 11.2.2. Configuration (Column 15)

You use this field to specify whether or not a UNISCOPE 100 or UNISCOPE 200 Display Terminal is attached to a multiplexer.

If the terminal is not attached to a multiplexer, leave this field blank. If the terminal is attached to a multiplexer, enter one of the following characters in this field: A through L, N, O, Q, R, or T through Z. A unique character must be used for each multiplexer. If more than one terminal is attached to the same multiplexer, all of the terminals must be associated with the same file name, their specifications must appear on contiguous lines, and the file name and the character selected for the multiplexer must be repeated on each line.

### 11.2.3. Type of Station (Column 16)

You use this field to specify whether the data on the file is to be received from or transmitted to a remote terminal.

If the data on the file is to be received from a remote terminal, enter an R in this field. If you use this entry, the file must be specified as an input or combined file on the file description specifications form and the file name must appear on the input format specifications form.

If the data on the file is to be transmitted to a remote terminal, enter a T in this field. If you use this entry, the file must be specified as an output or combined file on the file description specifications form and the file name must appear on the output format specifications form.

### 11.2.4. Transparency (Column 19)

You use this field to specify whether or not the transparent mode, which allows any hexadecimal value to be transmitted as data, is used.

If the transparent mode is not used, leave this field blank or enter an N. In this case, the data must not contain any control characters.

If the transparent mode is used, enter a Y in this field.

### 11.2.5. Switched (Column 20)

You use this field to specify how the connection is to be made when the remote terminal is connected on a switched line. In this situation, the central processor and the remote terminal have to be connected before data can be transmitted or received.

If the remote terminal is not connected on a switched line, leave this field blank.

If the central processor automatically accepts calls from the remote terminal (auto answer), enter an A in this field. This entry is required for interactive terminals.

If the operator at the central processor must answer the call (manual answer), enter a B in this field.

If the program automatically dials the number of the remote terminal (autocall), enter an E or S in this field.

If the operator at the central processor must dial the number of the remote terminal (manual call), enter an M in this field.

#### **11.2.6. Remote Terminal (Columns 48 through 51)**

You use this field to specify the type of remote terminal that is being used. The entry must be left-justified.

If you are using a device that is operating in the binary synchronous communications mode (usually computer to computer), leave this field blank or enter BSC.

If a teletypewriter is used, enter TTY in this field.

If a UNISCOPE 100 Display Terminal is used, enter 100 in this field.

If a UNISCOPE 200 Display Terminal is used, enter 200 in this field.

If a DCT 500 data communications terminal is used, enter 500 in this field.

If a DCT 524 data communications terminal is used, enter 524 in this field.

If a DCT 1000 data communications terminal (batch mode only) is used, enter 1000 in this field.

If a 1004 card processor system is used, enter 1004 in this field.

If a DCT 2000 data communications terminal is used, enter 2000 in this field.

If an IBM 2780 data transmission terminal is used, enter 2780 in this field.

If a 9200/9300 series system is used, enter 9300 in this field.

#### **11.2.7. Permanent Error Indicator (Columns 53 and 54)**

You use this field to specify the program indicator you want set on when a permanent error occurs during the reception or transmission of a data record. The setting of the indicator can be used to cause an error handling routine (such as controlled program termination or message display) to be executed.

If you leave this field blank, your program is automatically terminated when a permanent error occurs.

If you want to specify an indicator that is to be set on when a permanent error occurs, enter the indicator in this field. You can specify a general indicator (01 through 99), a halt indicator (H1 through H9), a control level indicator (L1 through L9), or the last record indicator (LR).

### 11.2.8. Wait Time (Columns 55 through 57)

You use this field to specify the amount of time (in seconds) allowed between terminal messages for your program to complete a program cycle.

The entry in this field may range from 001 to 999. The entry must be right-justified; leading zeros may be omitted.

If the time limit is not set high enough to allow the program to complete all record processing, the permanent error indicator will be set on. The time limit does not apply between file transmissions.

### 11.2.9. Last File (Column 60)

When you have specified a data communications file as a secondary file on the file description specifications form, this file is normally processed in the order it appears on that form. You can use this field to specify that the data communications file be processed only after the primary file and all other secondary files have been processed.

If you want the file to be processed in the order it appears on the file description specifications form, leave this field blank.

If you want the file to be processed after the primary file and all other secondary files have been processed, enter an L in this field.

### 11.2.10. Remote Device (Columns 65 through 70)

You use this field to specify the selection of an auxiliary device that is attached to a remote terminal. Entries in this field must be left-justified and are used in conjunction with the entries in the remote terminal field (columns 48 through 51). These entries are summarized in Table 11-1.

If you do not want to select an auxiliary device, leave this field blank.

If you want to select an auxiliary card punch, enter PCH in this field.

If you want to select an auxiliary printer, enter PRNTR in this field.

If you want to select an auxiliary card reader, enter RDR in this field.

If you have specified 2780 (IBM 2780 data transmission terminal) in the remote terminal field (columns 48 through 51), 1442-1, 1442-2, and 1443 can be substituted for RDR, PCH, and PRNTR, respectively.

Table 11-1. Summary of Remote Terminal and Remote Device Entries

Remote Terminal	Auxiliary Device	Entries	
		Remote Terminal (Columns 48 through 51)	Remote Device (Columns 65 through 70)
SPERRY UNIVAC DCT 1000 Data Communications Terminal	Card punch	1000	PCH
SPERRY UNIVAC DCT 2000 Data Communications Terminal	Card punch	2000	PCH
SPERRY UNIVAC 1004 Card Processor System	Card punch	1004	PCH
SPERRY UNIVAC 9200/9300 Series	Card punch	9300	PCH
IBM 2780 Data Transmission Terminal	Card punch	2780	PCH or 1442-2
	Card reader	2780	RDR or 1442-1
	Printer	2780	PRNTR or 1443

### 11.2.11. Terminal Name (Columns 71 through 74)

You use this field to specify the name of the remote terminal that is associated with the data communications file. This name must be the name that was specified for the terminal when the network was defined.

### 11.3. EXAMPLES OF ENTRIES ON THE TELECOMMUNICATIONS SPECIFICATIONS FORM

Figure 11-2 shows you examples of typical entries on the telecommunications specifications form.

SPERRY UNIVAC

RPG II  
TELECOMMUNICATIONS SPECIFICATIONS

PAGE NO	FORM TYPE	LINE NO	FILE NAME	CONFIGURATION							PERMANENT ERROR INDICATOR			LAST FILE L OR BLANK			NOT USED	REMOTE DEVICE	TERMINAL NAME	PROGRAM IDENTIFICATION												
				NOT USED	4-LIN/0/0-RIT-Z	BIT	NOT USED	TYPE OF STATION	TRANSPARENCY	SWITCHED	NOT USED	NOT USED	NOT USED	NOT USED	NOT USED	NOT USED					NOT USED	NOT USED	NOT USED									
1	2	3	5	6	7	14	15	16	17	18	19	20	21	47	48	51	52	53	54	55	57	58	59	60	61	64	65	70	71	74	75	80
	T	010	OUTCOM			T			A					500	500	61	62	63	200									OUT1				
	T	020		*										500	500	61	62	63	200									OUT2				
	T	030		*										TTY	500	61	62	63	200									OUT3				

Figure 11-2. Examples of Entries on Telecommunications Specifications Form

Explanation of Entries in Figure 11-2:

Line Number Explanation

010-030 The communications file is named OUTCOM (OUTCOM in columns 7 through 13). The data on the file is to be transmitted (T in column 16). The transparent mode is not used (column 19 is blank). Switched lines are used and the called station automatically accepts the call from the calling station (A in column 20). There are three remote terminals associated with this file and the specifications on line 010 in columns 16, 19, 20, and 55 through 57 apply to all three terminals (\* in column 14 on lines 020 and 030). Two DCT 500 data communications terminals and one teletypewriter are used (500 in columns 48 through 51 on lines 010 and 020 and TTY on line 030), the permanent error indicators are 61, 62, and 63 (61 on line 010, 62 on line 020, and 63 on line 030), the wait time is 200 seconds for all terminals (200 in columns 55 through 57), and the terminal names are OUT1, OUT2, and OUT3 (OUT1 in columns 71 through 74 on line 010, OUT2 on line 020, and OUT3 on line 030).





**PART 3. USING RPG II**



## 12. Controlling RPG II

### 12.1. GENERAL

As you know, each RPG II program that you write goes through the same general program processing logic. Because no two programs are exactly alike, you have to control how the program processing logic is applied in each case so that the steps in a program are performed when you want them to be performed. You do this by using a series of indicators that work in conjunction with the RPG II program processing logic. These indicators are 2-character alphabetic, numeric, or alphanumeric entries. To use an indicator, you must define the conditions that set it on. This is done either by placing an entry in the appropriate field on the specifications forms or by defining an entry externally. When the indicator has been defined, you then specify the program step or series of steps that you want performed when the indicator is set on by placing that indicator in the field on the specifications form that is associated with the program step or steps.

### 12.2. INDICATORS DEFINED ON THE RPG II SPECIFICATIONS FORMS

There are five types of indicators that are defined on the specifications forms. They are the record identifying indicator, control level indicator, field indicator, resulting indicator, and overflow indicator.

#### 12.2.1. Record Identifying Indicator

A record identifying indicator is used to identify a type of record in your program. It consists of a 2-character entry that is placed in columns 19 and 20 of the input format specifications form. The permissible entries are 01 through 99, L1 through L9, LR, and H1 through H9. The entry that you place in these columns is the record identifying indicator that is defined for this type of record. This indicator is set on only when the record selected for processing contains the record identification codes that are specified for it in columns 21 through 41.

The record identifying indicator is set on when the record is selected for processing but before the input data is moved to the input field. It remains on until you set it off by using a SETOF operation during the cycle or until the end of the cycle when RPG II sets it off.

Figure 12-1 shows you some examples of how to specify record identifying indicators.



### 12.2.2. Control Level Indicator

A control level indicator is used to indicate that a particular field in the input record is a control field. It consists of a 2-character entry that is placed in columns 59 and 60 of the input format specifications form. The permissible entries are L1 through L9. The entry that you place in these columns is the control level indicator that is defined for the input record field that you have chosen as a control field. Each time a record is read, the data in the control field is compared with the data from the same field in the previous record. If the contents of these fields are not the same, a control break occurs, the specified control level indicator is set on, and the total time processing that is conditioned by this indicator is executed.

The lowest control level is L1 and the highest is L9. This ranking causes all lower-level control level indicators to be set on when a control break occurs. For example, if a control break occurs and L9 is specified as the control level indicator, control level indicators L1 through L8 will also be set on. The setting on of the lower level control indicators only occurs in conjunction with a control break. If a control level indicator is specified as a record identifying indicator in columns 19 and 20 of the input specifications form or as a resulting indicator in columns 54 through 59 of the calculation specifications form, only the indicator specified will be set on when the specified conditions are met.

When you have different types of records in your program and you use control level indicators, each type of record normally contains the same number of control fields. There are occasions, however, where you need to have more control fields in one record type than in another. In these cases, you run into the problem of unwanted control breaks.

For example, assume that you are writing a program that will list the individual shipments of a particular item, the total number of this item that was shipped, and the total of all items that were shipped from two different warehouses. In this application, there are two types of records: the warehouse identification record and the item record. The format of these records is shown in Figure 12-2.

WAREHOUSE RECORD

W	WAREHOUSE NUMBER (L2)	WAREHOUSE NAME										
1	2	3	4									15

ITEM RECORD

I	WAREHOUSE NUMBER	ITEM NUMBER (L1)	AMOUNT SHIPPED							
1	2 (L2) 3	4	6 7						9	

Figure 12-2. Unwanted Control Break Example — Input Records





01	ALBANY			TOTAL ITEMS SHIPPED	UNWANTED CONTROL BREAK OUTPUT
	200	3			
	200	2			
	201	5		TOTAL ITEMS SHIPPED	
		4		TOTAL ITEMS SHIPPED	
		4		TOTAL ITEMS SHIPPED	
		9		WAREHOUSE TOTAL	
02	PHILA			TOTAL ITEMS SHIPPED	
	200	6			
	200	3			
		9		TOTAL ITEMS SHIPPED	
	201	2			
		2		TOTAL ITEMS SHIPPED	
		11		WAREHOUSE TOTAL	
		20		GRAND TOTAL	

a. Incorrect output

01	ALBANY			
	200	3		
	200	2		
		5		TOTAL ITEMS SHIPPED
	201	4		
		4		TOTAL ITEMS SHIPPED
		9		WAREHOUSE TOTAL
02	PHILA			
	200	6		
	200	3		
		9		TOTAL ITEMS SHIPPED
	201	2		
		2		TOTAL ITEMS SHIPPED
		11		WAREHOUSE TOTAL
		20		GRAND TOTAL

b. Correct output

Figure 12-4. Unwanted Control Break Example — Outputs



### 12.2.3. Field Indicator

A field indicator is used to indicate that a particular field in the input record is to be tested to see if it is plus or minus or contains all zeros or blanks. Numeric fields can be tested for any of these conditions; alphanumeric fields can only be tested for all blanks. The field indicator consists of a 2-character entry that you place in the appropriate field in columns 65 through 70 of the input format specifications form. The permissible entries are 01 through 99 and H1 through H9. The entry that you place in these columns is the field indicator that is defined for the field to be tested. Each time a record is read, the data in the specified field is tested for the condition you indicated; if it is met, the field indicator is set on. The field indicator remains on until another record is read that does not meet the specified condition.

The same field indicator may be specified for two conditions, in which case the indicator will be set on when a record is read that satisfies either condition.

If S is specified in column 42 of the control card specifications form and a zero or blank indicator is specified for a field in columns 69 and 70 of the input format specifications form, the setting on and off of this indicator will differ from the normal process. In this case, the zero or blank indicator for a field will be set on if the field is reset to all zeros or blanks by a blank after specification (B in column 39) on the output format specifications form.

### 12.2.4. Resulting Indicator

A resulting indicator is used to indicate that the result of an operation is to be tested for a specified condition. A resulting indicator consists of a 2-character entry that you place in the appropriate field in columns 54 through 59 of the calculation specifications form. The permissible entries are 01 through 99, L1 through L9, LR, H0 through H9, OA through OG, OV, U1 through U8, KA through KN, and KP through KW. The entry that you place in these columns is the resulting indicator that is defined for the associated operation. The resulting indicator is set on if the associated operation is executed and the specified condition is met. The resulting indicator remains on either until the operation is executed again and the specified condition is not met or until you set it off by using a SETOF operation. The conditions that a resulting indicator can be used to indicate depend on the type of operation it is used with.

#### ■ Arithmetic Operation

The resulting indicator indicates the status of the contents of the result field. If a resulting indicator is specified in columns 54 and 55 (plus), the indicator will be set on if the contents of the result field are greater than zero. If an indicator is specified in columns 56 and 57 (minus), the indicator will be set on if the contents of the result field are less than zero. If an indicator is specified in columns 58 and 59 (zero), the indicator will be set on if the contents of the result field are zero. If half adjust is specified in column 53, the setting of the indicators will reflect the contents of the result field after half adjustment has taken place.

If S is specified in column 42 of the control card specifications form, the first zero or blank indicator for the result field will be set if the field is blanked-after on the output format specifications form (B is specified in column 39 of the output format specifications form). The same indicator may be used to indicate two conditions. In these cases, the indicator is set on if either condition is met.

### ■ Compare Operation

The resulting indicator indicates the relationship of factor 1 to factor 2. If a resulting indicator is specified in columns 54 and 55 (1>2), the indicator will be set on if factor 1 is greater than factor 2. If an indicator is specified in columns 56 and 57 (1<2), the indicator will be set on if factor 1 is less than factor 2. If an indicator is specified in columns 58 and 59 (1=2), the indicator will be set on if factor 1 is equal to factor 2.

The same indicator may be used to indicate two conditions. In these cases, the indicator is set on if either condition is met.

### ■ Test Bit Operation

The resulting indicator indicates the status of the bits that are tested in a 1-character alphanumeric field. The name of the field that contains the bits to be tested is specified in columns 43 through 48 of the result field, and which of these bits are to be tested is specified in factor 2. Factor 2 specifies the numbers of the bits to be tested (0 through 7) or the name of a 1-character alphanumeric field that is to be compared with the result field to see whether the corresponding bits in the result field are 1's. If a resulting indicator is specified in columns 54 and 55, the indicator will be set on if the specified bits are 0's. If an indicator is specified in columns 56 and 57, the indicator will be set on if the specified bits are mixed (0's and 1's). If an indicator is specified in columns 58 and 59, the indicator is set on if the specified bits are 1's.

The same indicator may be used to indicate two conditions. In these cases, the indicator is set on if either condition is met.

### ■ Test Numeric Operation

The resulting indicator indicates whether the result field contains numeric characters. If a resulting indicator is specified in columns 54 and 55, the indicator will be set on if all characters are numeric. If an indicator is specified in columns 56 and 57, the indicator will be set on if the field contains leading blanks followed by numeric characters. If a resulting indicator is specified in columns 58 and 59, the indicator will be set on if the field contains all blanks. Each character, except the low-order character, must contain a hexadecimal F in the zone portion and a digit (0 through 9) in the digit portion to be considered numeric. The low-order character may contain a hexadecimal C, D, or F in the zone portion and a digit in the digit portion.

### ■ Test Zone Operation

The resulting indicator indicates the result of testing the zone of the leftmost character of the result field. If a resulting indicator is specified in columns 54 and 55 (plus), the indicator is set on if a 12 zone (hexadecimal C - includes A through I or &) is present. If an indicator is specified in columns 56 and 57 (minus), the indicator is set on if an 11 zone (hexadecimal D - includes J through R or -) is present. If an indicator is specified in columns 58 and 59 (zero), the indicator is set on if any other zone is present.

The same indicator may be specified for two conditions. In these cases, the indicator will be set on if either condition is met.

### ■ Look-up Operation

The resulting indicator indicates how the look-up of a table or array is to be performed. If the resulting indicator is specified in columns 54 and 55 (high), the next higher element than the search argument specified in factor 1 is retrieved and the indicator is set on. If an indicator is specified in columns 56 and 57 (low), the next lower element than the search argument specified in factor 1 is retrieved and the indicator is set on. If an indicator is specified in columns 58 and 59 (equal), the element that is equal to the search argument specified in factor 1 is retrieved and the indicator is set on. An indicator cannot be specified for high or low unless the table or array is in sequence. The sequence is specified in columns 45 and 57 of the file extension specifications form. Indicators can be specified for high, low, or equal or for high and equal or low and equal. In the latter cases, the equal condition takes precedence and only when it is not found will the next higher or lower element be retrieved.

### ■ CHAIN Operation

The high field (columns 54 and 55) and low fields (columns 56 and 57) of the resulting indicators can be used with the CHAIN operation to specify an indicator that is set on when a no-record-found condition occurs. If an indicator is not present in either field and a no-record-found condition occurs, the HO (halt 0) indicator is set on and the program terminates unless provision is made in the program to set this indicator off.

### ■ SETOF and SETON Operations

Columns 54 through 59 are used to specify up to three indicators that are to be set off when the SETOF operation is executed or that are to be set on when the SETON operation is executed.

## 12.2.5. Overflow Indicator

An overflow indicator is used to control the printing of heading information when forms overflow occurs. It consists of a 2-character entry that is placed in columns 33 and 34 of the file description specifications form. The permissible entries are OA through OG and OV. The entry that you place in these columns is the overflow indicator that is defined for the printer or line counter file on this line. The overflow indicator is set on when the forms overflow channel is sensed either on the printer or on the line counter file. It is set on when a line is printed on or past the line of the form that is associated with the forms overflow punch on the carriage control tape or if the forms overflow punch is passed during a space operation. It is not set on, however, if the forms overflow punch is passed during a skip operation. A skip to a new page caused by an output specification not conditioned by an overflow indicator sets the overflow indicator off.

An overflow indicator cannot condition an exception line (E in column 15), but may condition fields within the exception record.

The overflow indicator is set off after detail output time. If it is set on during total time, it is set off after the next detail output time. If it is set on during detail time, it remains on for a complete cycle and is set off after detail output time.

Overflow output can be caused at any time during output by using the fetch overflow routine. The fetch overflow routine is used by placing an F in column 16 of the output format specifications form on any detail, total, or exception line that is not conditioned by an overflow indicator. Each output line is tested during output to see if its conditioning indicators are on. If they are and F has been specified in column 16, the overflow indicator for the associated file is checked to see if it is on. If the indicator is on and the overflow lines have not been written since it was set on, the overflow routine is fetched, the overflow lines associated with the file are written, and then the line that caused the overflow to be fetched is written. Figure 12-5 shows an example of the entries using fetch overflow for detail, total, and exception output. The records on lines 010, 030, 050, and 070 are detail records. The record on line 010 contains heading information that is printed on the top of each page when forms overflow occurs. When the overflow indicator (OF) is on, line 070 (F in column 16) causes the fetch overflow routine to be executed, which in turn causes line 010 to be printed next; that is, line 010 is printed and then line 070 is printed. When the fetch overflow routine is executed, the detail record printing sequence is line 030, line 050, line 010, and line 070.

OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW	SPACE		SKIP			OUTPUT INDICATORS						FIELD NAME	DATA FORMAT P/B/L/R	END POSITION IN OUTPUT RECORD	CONSTANT OR EDIT WORD																						
				BEFORE	AFTER	BEFORE	AFTER	N-NOT	N-NOT	N-NOT	AND	AND	AND	AND					AND	AND																				
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45		
		010	F																																					
		020																																						
		030																																						
		040																																						
		050																																						
		060																																						
		070																																						
		080																																						
		100																																						
		110																																						
		120																																						
		130																																						
		140																																						
		150																																						
		160																																						
		170																																						
		180																																						

Figure 12-5. Using Fetch Overflow

The records on lines 090, 110, and 130 are total records. Line 090 is printed when the L1 indicator is on. When the overflow indicator (OF) and the L2 indicator are on, line 110 (F in column 16) causes the fetch overflow routine to be executed, which in turn causes line 010 to be printed before line 110. Line 130 is printed when the L2 indicator is on. When the fetch overflow routine is executed, the total record printing sequence is: line 090, line 010, line 110, and line 130.

The records on lines 150 and 170 are exception records. When the overflow indicator (OF) is on, line 150 (F in column 16) causes the fetch overflow routine to be executed, and this in turn causes line 010 to be printed before line 150. When the fetch overflow routine is executed, the exception record printing sequence is line 010, line 150, and line 170.

### 12.2.6. Function Key Indicator

The function key indicators (KA through KN and KP through KW) correspond to function keys 1 to 22. When one of the function keys and the transmit key are pressed, the corresponding indicator is set on and the transmitted data is available to the program. All function key indicators are set off before a read to a workstation file is performed. If the function key is pressed, one indicator is set on after input.

You use the function key indicators:

- To condition calculation operations in columns 9 through 17 of the calculation specifications form
- To condition output operations in columns 23 through 31 of the output format specifications form
- As a resulting indicator in columns 54 through 59 of the SETOF operation on the calculation specifications form

#### NOTES:

1. *IBM System/34 has two other function keys and so has two other indicators, KX and KY. Thus, errors may occur when IBM System/34 programs are converted to OS/3. For compatibility with BC/7, indicators KA, KB, and KC may be referenced as F1, F2, F3, respectively.*
2. *In RPG II IMS action programs, function key indicators (KA-KN and KP-KW) are not set when function keys are pressed. IMS may be configured to initiate transactions when function keys are pressed.*

### 12.3. INDICATORS NOT DEFINED ON THE RPG II SPECIFICATIONS FORMS

There are two types of indicators that are not defined on the RPG II specifications forms. They are the external indicators that are defined in the job control language and the internal indicators that are defined by the RPG II program.

### 12.3.1. External Indicators

There are eight external indicators: U1 through U8 that correspond to UPSI bits 0 through 7, respectively. These indicators are defined by a SET UPSI job control statement. They are set on or off by this job control statement prior to the execution of the program.

The format of the SET UPSI job control statement is as follows:

```
//ΔSETΔUPSI , u1u2u3u4u5u6u7u8
```

where:

u<sub>1</sub> . . . u<sub>8</sub>

Represents the external indicators U1 through U8.

- 1 Set external indicator on.
- 0 Set external indicator off.
- x The current setting of the external indicator is not changed.

For example, assume that external indicators U1, U4, and U6 are to be set on; U2 and U5 are to be set off; and the current setting of U3, U7, and U8 are not to be changed. In this case, the SET UPSI job control statement would be:

```
//ΔSETΔUPSI , 10X101XX
```

➔ To control external indicators with an RPG program, use the SETON and SETOF operations.

At the end of the job step, the U1 to U8 indicators are copied to the UPSI byte in the job preamble.

### 12.3.2. Internal Indicators

There are five internal indicators: 1P (first page), H0 (halt zero), L0 (control level zero), LR (last record), and MR (matching record). Each indicator represents a condition that results from the RPG II program processing logic; that is, if the condition that the indicator represents occurs during the execution of a program, the RPG II program processing logic will set the indicator on.

#### ■ 1P Indicator

The 1P indicator is set on at the beginning of program execution. It is used to condition the printing of heading or detail lines on the first page of your report. It is set off immediately before the first record is read.

- HO Indicator

The HO indicator is set on when any one of several error conditions occurs during the execution of your program. These error conditions fall into two categories: error conditions which, when they occur, allow you the option of setting off the HO indicator so that your program can continue; and those that do not provide this option. In either case, an error code is placed in the first byte of the \*ERROR field. For further information on the error codes, see the system messages programmer reference manual.





If any of the following error conditions set on the HO indicator during the execution of your program and you wish to continue processing in spite of the error condition, you can do this by writing a routine on the calculation specifications form that tests the \*ERROR field for the presence of the particular code for the error condition and sets the HO indicator off if the error code is present. We'll cover this type of routine in our discussion of the \*ERROR field in Appendix C.

- Record sequence error
- Undefined record type
- Collating sequence error
- Trying to add a duplicate record to an indexed sequential file
- No resulting indicators specified and no-record-found condition occurs during a CHAIN operation
- The overflow area for an indexed sequential file is full.
- No record found condition when C1 through C9 is used for chaining

If any of the following error conditions set on the HO indicator during the execution of your program, the program will terminate because you do not have the option to continue processing by setting the HO indicator off. In these cases, the error cannot be bypassed.

- At least one input and one output file has not been opened.
- Invalid array index
- Sequence checking is specified for a table and an element is out of sequence.
- An attempt is made to find the square root of a negative number.
- The writing of a record from an update file is attempted before it is read.
- Any error condition for indexed sequential or direct files other than full overflow area, duplicate records, and no record found

■ LO indicator

The LO indicator is used to condition total calculations when a control break has not occurred. The LO indicator is always on and cannot be set off.

#### ■ LR indicator

The LR (last record) indicator is set on by the RPG II program processing logic before total time when the last record has been processed. This means that any output written when the last record is processed should be specified as total time output. Exception output may be performed when LR is on by specifying LR in columns 7 and 8 of the calculations specifications for the EXCPT operation. The last record is processed when all records from the files for which end-of-file has been specified in column 17 of the file description specifications form have been processed. If end-of-file has not been specified for any files, all input files must be end-of-file before the last record is processed. When the LR indicator is set on, all control level indicators (L1 through L9) that are defined are also set on. If the LR indicator is used as a conditioning indicator or with the SETON operation, the L1 through L9 indicators are not set on at that time.

If the LR indicator was set on during detail calculations, the L1 through L9 indicators are set on before the next input record is read. The total time calculations and output are then performed and the program is terminated.

If the LR indicator is set on during total calculations, the program is terminated after total output time.

You cannot set off the LR indicator.

#### ■ MR indicator

The MR (matching record) indicator is associated with the M1 through M9 matching field entries in columns 61 and 62 of the input format specifications form. It is set on when all the matching fields in a record of a secondary file match all the matching fields in the primary file. The MR indicator remains on during the complete processing cycle of both the primary and secondary records. The MR indicator is set off when all total calculations, output, and overflow for the records have been executed.

### 12.4. USING THE INDICATORS

The indicators are used to condition the execution of your program. They can be used to specify:

- file conditioning, if a file is to be used during the execution of a program;
- control level, when calculations and output are to be performed;
- field record relation, when a field in a record is available for input;
- calculation conditioning, the conditions that must be present before a calculation can be performed; and
- output conditioning, the conditions that must be present before a complete record or field within a record can be written.

### 12.4.1. File Conditioning

The file condition indicators are specified only in columns 71 and 72 of the file description specifications form. The only entries that are permitted are the U1 through U8 external indicators.

If an external indicator is specified in these columns, the associated file cannot be accessed unless the specified indicator was previously set on prior to program execution by a SET UPSI job control statement (12.3.1). If the external indicator is not on, the file is considered to be at end-of-file and, therefore, cannot be accessed.

A file condition indicator may be specified for input, output, combined, update, table output, and record address files. It cannot be specified for input table or array files, chained files, or a file that is being accessed by a record address file.

### 12.4.2. Control Level

The control level indicators are specified in columns 7 and 8 of the calculation specifications form. The only entries that are permitted are L0, L1 through L9, and LR. These indicators are used to control when calculations are to be performed. If the entry is left blank, the calculation is performed at detail time.

If L0 is specified, the calculation is performed every program cycle at total time.

If L1 through L9 is specified, the calculation is performed only when the specified indicator is on. When the specified indicator is set on by a control break (columns 59 and 60 on the input format specifications form), all lower level indicators are also set on.

If LR is specified, the calculation will be performed during the last total time.

### 12.4.3. Field Record Relation

The field record relation indicators are specified in columns 63 and 64 of the input format specifications form. The entries that are permitted are 01 through 99, L1 through L9, MR, and U1 through U8.

An entry in columns 63 and 64 is used when there is an OR relationship between the various types of input records that are to be processed. The OR relationship is used to identify those types of records in which most of the fields are in the same position within the records. The entries that you make in columns 63 and 64 specify the fields that are associated with each record type; that is, the field that is defined on a line is only available for input when the indicator you specified in columns 63 and 64 is on or if these columns are left blank. In the latter case, the field is common to all record types. When field record relation is used, the following points should also be borne in mind:

- Main storage can be conserved by grouping together, on consecutive lines, all field descriptions that do not use field record relation and by grouping together all field descriptions that use the same field record relation indicator.

- All fields that are used for control level and chaining and matching fields that are not used with field record relation must be listed on the input format specifications form before those control level and matching and chaining fields that use field record relation.
- L1 through L9, U1 through U8, and MR cannot be used as field record relation indicators for matching and control fields. In these cases, only O1 through 99 and H1 through H9 are permitted.
- When more than one matching field is specified and some of the matching fields do not use field record relation and the others do, all of the matching fields must be specified first without field record relation and those fields that use field record relation must be specified in groups by indicator. For example, assume that M1, M2, and M3 are the matching fields; M1 uses O1 and O2 as field record relation indicators; M2 uses O1 as field record relation indicator; and M3 does not use field record relation. In this case, the matching fields would be listed on the input format specifications form in the following order:

Matching Field (Columns 61 and 62)	Field Record Relation (Columns 63 and 64)
M1	Blank
M2	Blank
M3	Blank
M1	O1
M2	O1
M1	O2

#### 12.4.4. Calculation Conditioning

The conditions that must be present before a calculation can be performed are specified by placing an indicator in columns 10 and 11, 13 and 14, and 16 and 17 of the calculation specifications form. The entries that are permitted are O1 through 99, H0 through H9, L1 through L9, LR, MR, OA through OG, OV, KA through KN, KP through KW, and U1 through U8.

The entries that you make in these fields are those conditions that you have determined must be present before the calculation on this line can be performed. The calculation will be performed only when the specified indicators are on. Up to three indicators may be specified on a line. If an indicator must not be on (the condition must not be present), this can be specified by placing N before the indicator (in column 9, 12, or 15).

When more than three conditions are required to condition an operation, the first three indicators are placed on the first line and the additional indicators (4 through n) are placed on successive lines (3 per line) until all conditions have been specified. In these cases (an AND relationship), each successive line (after the first) must have AN specified in columns 7 and 8. Columns 18 through 59 must be blank on the first line and on all successive AN lines except the last. The last line must contain the operation to be performed. A maximum of seven AN lines may be used for one operation.



When the indicators are used to condition the writing of a complete record, columns 32 through 70 must be blank. The complete record will be written only when the specified indicators are on. Up to three indicators may be specified on a line. If an indicator must not be on (the condition must not be present), this can be specified by placing N before the indicator (in column 23, 26, or 29).

If more than three conditions are required to condition the writing of a record, the first three indicators are placed on the first line and the additional indicators (4 through n) are placed on successive lines until all conditions have been specified. In these cases (an AND relationship), each successive line after the first must have AND specified in columns 14 through 16 and columns 32 through 70 must be blank.

When the presence of any one of various sets of conditions will permit a record to be written, the first set of conditions (1 through 3 indicators) is specified on the first line and each additional set on a successive line. In these cases (an OR relationship), each successive line after the first must have OR specified in columns 14 and 15 and must be blank in columns 32 through 70.

The conditions that must be present before a field within a record can be written are specified by placing from one to three indicators in columns 23 through 31 on the line that defines the field. The field will be written only when the specified indicators are on. AND/OR lines cannot be used with fields within a record; consequently, three indicators are the maximum that can be specified for any one field.

After it has been determined that a record is to be written (all required conditions are present), the fields that are defined for the record in columns 32 through 37 (commencing on the first line after the last line that specifies the record conditions) that do not have any entries in columns 23 through 31 and those whose specified conditions are present are written.

Figure 12-7 shows some examples of how you specify output conditioning for records and fields.

The detail record on line 010 is written when indicators 01 and 03 are on. The record always contains the field DATA1; the field DATA2 is written only when indicator 05 is on. An AND relationship is shown on lines 040 through 060. The detail record on line 040 is written when indicators 01, 03, 05, and 07 are on and the field DATA3 is always included in the record.









Table 12-3. Setting Indicators On and Off (Part 1 of 2)

Indicator Type	When Set On	When Set Off
Record identification indicator	When the specified record is read and prior to the execution of total calculations	After the current processing cycle is completed and before the next record is read during the next cycle
Field indicator	When the condition the indicator represents is present in the specified field	Prior to the next time the field is to be tested for the condition
Control level indicator	When the value in the specified control field changes. (All lower level indicators are also set on.)	When the following detail cycle is completed
Resulting indicator	When the calculation is performed and the condition that the indicator represents is present	The next time the calculation is performed and the condition that the indicator represents is not present
Overflow indicator	When a line is printed on the form on or past the forms overflow channel or if the form is spaced past that point	After the following heading and detail lines are printed
U1-U8 external indicators	By a //△SET△UPSI job control statement prior to the execution of your program or by a SETON operation within your program or when used as a resulting indicator in calculations	By a SETOF operation within your program
1P (first page) internal indicator	At the beginning of program execution	Before the first input record is read
H0-H9 internal indicators	By an error condition (H0) or by a SETON operation within your program	By a SETOF operation within your program
L0 internal indicator	Always on	Never









The statements shown in Figure 12-12 are the same as those in Figure 12-11 except that line 010 must be specified with columns 9 through 17 left blank or specified with other indicators. This is necessary so that these statements will be executed during a regular program cycle rather than only at the beginning of program execution.

If you specify the unsolicited inquiry request subroutine, the message

RPG038 JOB IS NOW ABLE TO ACCEPT UNSOLICITED KEY-IN 'IR'

will be displayed on the system console at the beginning of program execution to inform you that your program will accept unsolicited IR type-ins. The program will then halt. To continue, type in

GO△your program name

From this point on, the program will test during each cycle to see if an unsolicited inquiry request has been made. If so, the specified indicator is set on, the conditioned calculation or output operations are performed, and the inquiry request is reset. If not, the indicator is set and the program continues.

At this point, the system you use dictates the method of data entry.

If you want to make an unsolicited inquiry request and are using an OS/3 system at level 8.1 or earlier, type in: ←

▶ job number (supplied by system) 0 IR ←

Thus, you enter:

▶jj0△IR ←

where:

▶ Is the SOE. ←

jj Is the job number (supplied by system).

0 Is the message identification. It identifies the message as being unsolicited.

△ Is a space.

IR Represents an inquiry request.

For example, ▶10△IR is an unsolicited inquiry request for job 1, and ▶140△IR is an ←  
unsolicited inquiry request for job 14.

→ If you want to make an unsolicited inquiry request and are using an OS/3 system at level 8.2 or later, type in:

→ ▶UNS job name IR

Thus, you enter:

▶UNS△XXXXXX△IR

where:

▶ Is the SOE.

UNS Specifies an unsolicited message for a user job.

△ Is a space.

XXXXXX Is the job name.

IR Represents an inquiry request.

Note that because an inquiry request is reset after the IR type-in is made, you must make a new type-in each time you want to have the associated indicator set on and the conditioned calculation or output operations performed.

### 12.6.3. Incorrect Type-Ins

If you have allocated a PF key or the unsolicited inquiry request subroutine to your program and you make an incorrect type-in when you attempt to activate either one, the message

RPG040 LAST KEY INVALID OR INACTIVE. VALID KEYS = IR, PF1-PF9

will be displayed on the system console. To continue, repeat the appropriate type-in with the correct information.



## 13. File Processing Methods

### 13.1. GENERAL

RPG II allows you to process files in several ways. The method that you choose for your application is determined by how your files are organized and how records can be retrieved from a file that is organized in a particular manner.

In certain cases, this may require you to convert your data files from one format to another before you can use them.

If you wanted to use a file processing method that requires an indexed sequential file and the file is a sequential file, you would have to convert it to an indexed sequential file before you could use it. You could do this conversion by writing an RPG II program or by using the file processing utility described in the data utilities user guide. Table 13-1 lists the file organization, how records are retrieved, and the processing methods that can be used.

Table 13-1. File Organization, Record Retrieval, and Processing Methods (Part 1 of 2)

File Organization	Record Retrieval	Processing Methods
Sequential	Sequentially (consecutively)	<ol style="list-style-type: none"> <li>1. Sequential</li> <li>2. Matching records</li> </ol>
	Randomly	Using a tag (ADDROUT) file to supply record addresses
Indexed	Sequentially (consecutively)	<ol style="list-style-type: none"> <li>1. Sequentially</li> <li>2. Matching records</li> </ol>
	Sequentially (consecutively) between limits	<ol style="list-style-type: none"> <li>1. Using a record address file to provide the upper and lower limits of processing</li> <li>2. Using SETLL operation</li> </ol>
	Randomly	<ol style="list-style-type: none"> <li>1. Using a record address file to supply the keys of the records to be processed</li> <li>2. Chaining</li> </ol>

Table 13—1. File Organization, Record Retrieval, and Processing Methods (Part 2 of 2)

File Organization	Record Retrieval	Processing Methods
Direct	Randomly	<ol style="list-style-type: none"> <li>1. Using a chaining file to supply relative record numbers</li> <li>2. Using a tag (ADDROUT) file</li> </ol>
	Sequentially (consecutively)	<ol style="list-style-type: none"> <li>1. Sequential</li> <li>2. Matching records</li> </ol>

## 13.2. SEQUENTIAL PROCESSING

Sequential processing consists of processing records in the order in which they appear on the file until the file is exhausted or the program terminates; that is, the first record on the file is processed, the second record is then processed, and so on.

## 13.3. PROCESSING WITH MATCHING RECORDS

Processing with matching records allows you to sequentially process multiple input files as one large sequential file. The sequence in which the records on the individual files are selected for processing is controlled by specifying matching fields for each of the files involved. The fields are compared and the values they contain govern which record is selected.

### 13.3.1. Primary File

When multiple input files are used, one file must be specified as the primary file by placing a P in column 16 on the file description specifications form, and all of the other input files must be specified as secondary or chained files by placing an S or C in column 16. The primary file is the controlling file in a program.

When you process with matching records, the primary file must have a complete set of matching fields specified for it. The primary file will be processed first when its matching fields match those of one or more of the secondary files.

### 13.3.2. Specifying Matching Fields

Matching fields are specified by placing M1 through M9 in columns 61 and 62 on the input format specifications form on the line that specifies the field for a file. The matching fields need not be specified in any order; that is, M9 can be specified before M3, and so on. The field with the highest number will be the first one to be compared, the second highest will be the second, and so on. Figure 13-1 shows a typical example of how matching fields are specified.



3. If the files are in ascending order, the record with the lowest value in its matching fields is processed first. If the files are in descending order, the record with the highest value in its matching fields is processed first.
4. When a record from the primary file matches one or more secondary records, the primary record is processed before the secondary records.
5. When two or more secondary records match, they are processed in the order that the secondary files are specified on the file description specifications form.

Figure 13-2 shows an example of how records are selected in accordance with the rules. In this example, three files are used that contain records having 2-character matching fields and a record sequence number in record position 80. The primary file is named PRIME, the first secondary file is named ONE, and the second secondary file is named TWO. If a matching field is blank, the record has no matching field. The contents of each file and the step-by-step selection process for the first 10 records are shown. The record that is selected at each step is circled.

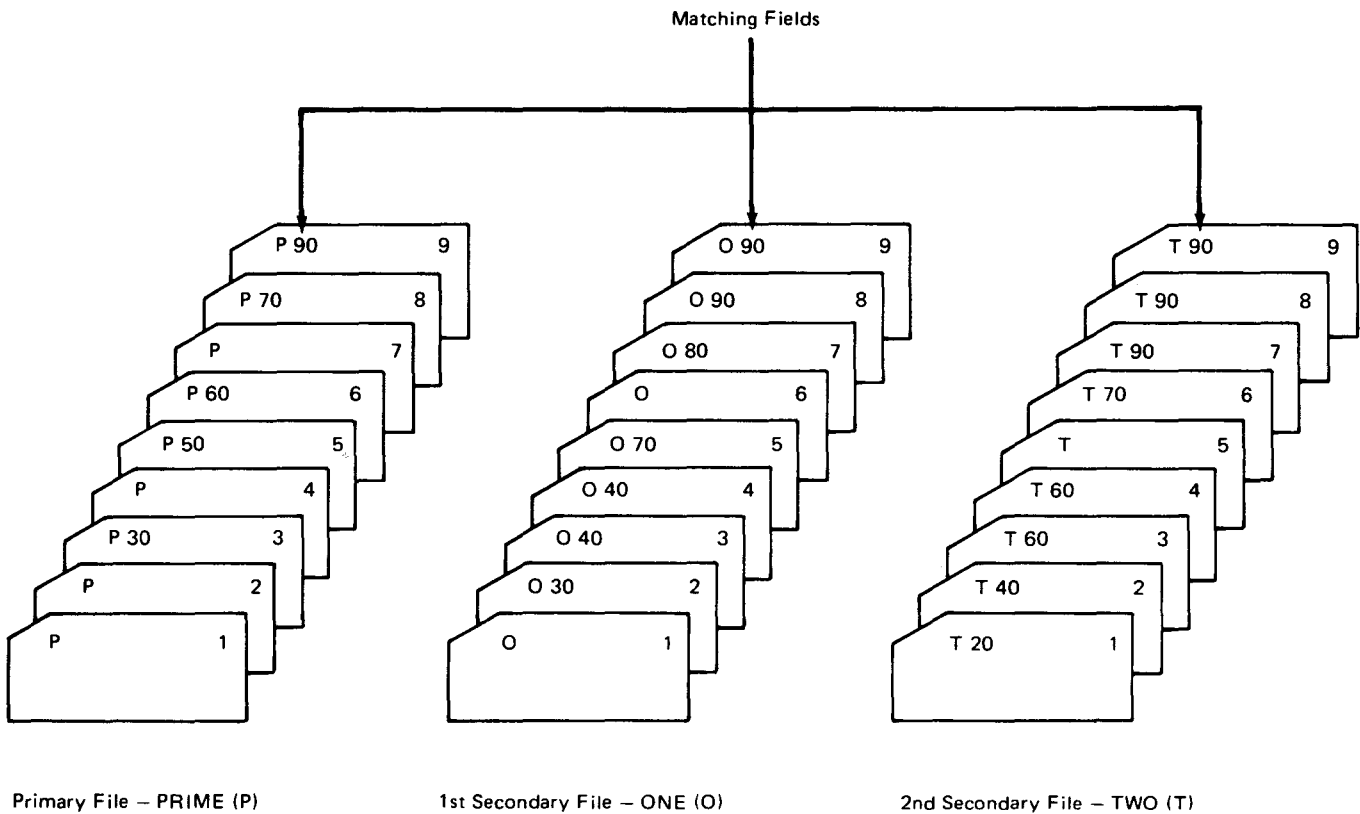


Figure 13-2. Matching Record Selection (Part 1 of 3)

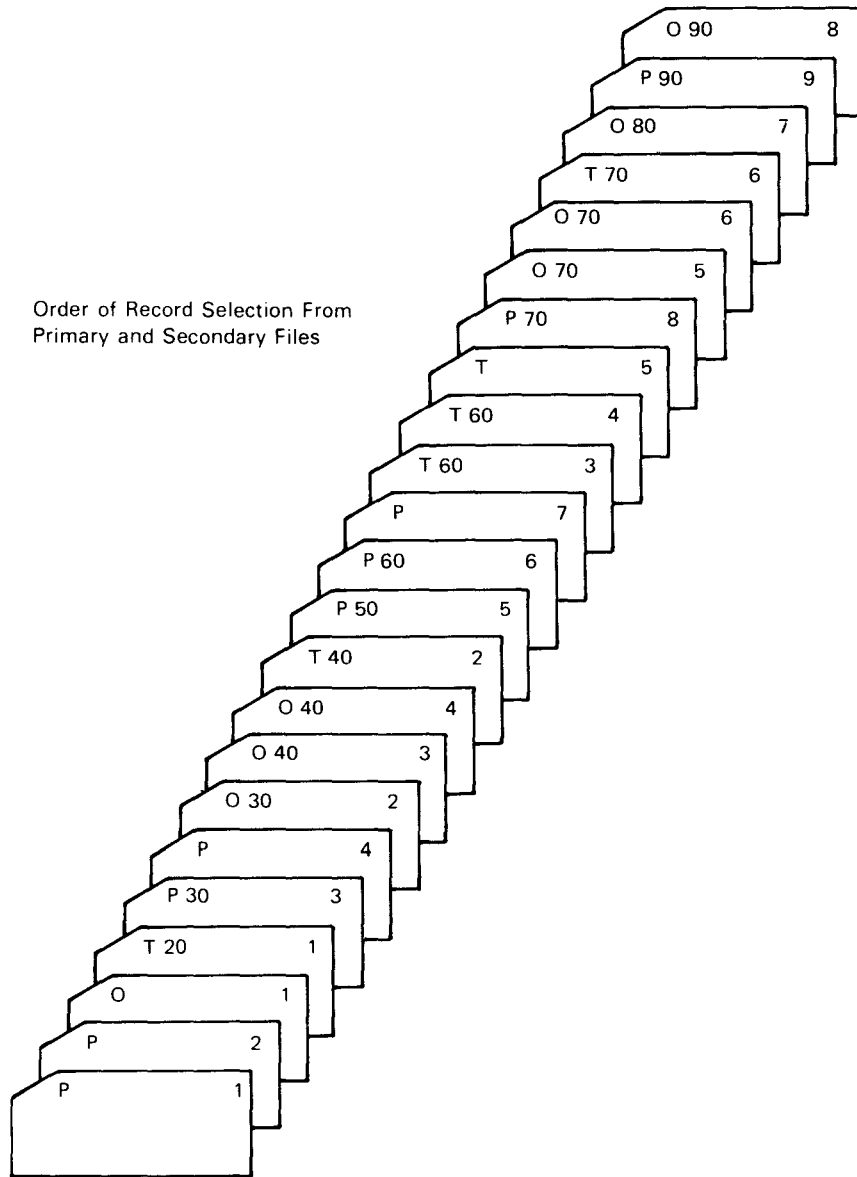


Figure 13—2. Matching Record Selection (Part 2 of 3)

## Record Selection Steps

	File <u>P</u>	File <u>O</u>	File <u>T</u>	
①	P 1	O 1	T 20 1	The first record from each file is read. The P and O records have no matching field so they will be processed before the T record. The P record is from the primary file so it will be selected first.
②	P 2	O 1	T 20 1	The next P record is read because the last record selected was from the P file. It is selected because it has no matching fields and is from the primary file.
③	P 30 3	O 1	T 20 1	The next P record is read. The O record is selected because it has no matching field.
④	P 30 3	O 30 2	T 20 1	The next O record is read. The T record is selected because the value in its matching field is lower than that of the other two.
⑤	P 30 3	O 30 2	T 40 2	The next T record is read. The P and O records have a lower value in their matching fields and these values are equal so the MR indicator is set on. The P record is selected first; it is from the primary file.
⑥	P 4	O 30 2	T 40 2	The next P record is read. It is selected before the O record because it has no matching fields.
⑦	P 50 5	O 30 2	T 40 2	The next P record is read. The O record is selected because its matching field is lowest.
⑧	P 50 5	O 40 3	T 40 2	The next O record is read. The O and T records have equal values in their matching fields, however, the O record is selected for processing first because it is from the first secondary file.
⑨	P 50 5	O 40 4	T 40 2	The next O record is read. It is selected because it has the same value in its matching field that the previously selected O record had.
⑩	P 50 5	O 70 5	T 40 2	The next O record is read. The T record is selected because it has the lowest value in its matching fields.

Figure 13-2. Matching Record Selection (Part 3 of 3)

### 13.3.4. Matching Record Indicator (MR)

The MR indicator is associated with the M1 through M9 matching fields. It is used to condition operations specified on the calculation specifications form and the output format specifications form.

The MR indicator is set on when the matching fields of a secondary file record match the matching fields of a primary file record. It is set on before detail processing for the first record of the matched primary file and remains on during processing of all succeeding primary and secondary file records that contain the same matching field.

The MR indicator is set off when all total calculations and output are completed for the last record that contains the same matching field.

## 13.4. PROCESSING BY CHAINING

Chaining allows you to randomly retrieve selected records from an indexed sequential or direct file by specifying a field (the chaining field) which is used to locate a record in that file (the chained file). Chaining can be accomplished by using the CHAIN operation on the calculation specifications form or specifying the chaining field by C1 through C9 on the input format specifications form.

The use of the CHAIN operation allows chaining to be:

- performed with a chaining field whose value is calculated or read from an input file;
- performed at either detail or total calculation time;
- performed more than once by using one or more chaining fields and calculations to chain to more than one file;
- conditioned by the results of a test or calculation; and
- performed in an internal subroutine.

C1 through C9 chaining differs from the CHAIN operation in that it is not as flexible a method. The C1 through C9 chaining indicators apply only to input fields, and they cause chaining to occur only before detail time. In addition, C1 through C9 chaining requires additional entries on the file extension specifications form.

If you intend to use chaining, it is suggested that you use the CHAIN operation wherever possible.

When you use chaining you must observe the following rules:

1. The file specified as a factor 2 in a CHAIN operation cannot have SPECIAL specified as the device on which it is contained and it cannot have matching fields, control level fields, C1 through C9 fields, look ahead fields, or numeric sequence entries.
2. The chaining field must have the same length as the keys in the chained file. If the keys in the chained file are alphanumeric, the chaining field may be either alphanumeric or numeric. If the keys in the chained file are numeric, the chaining field must be numeric. When the keys in the chained file are in packed numeric format, the chaining field must be either a packed numeric field of the same length as the key field or a numeric field of the same length as the unpacked length of the keys in the chained file.
3. The chaining field must be a numeric field with zero decimal places when chaining to a direct file.

### 13.4.1. Chaining Indexed Sequential Files

Every record in an indexed sequential file has a key which can be used to retrieve the record. These keys are used in chaining to retrieve records from an indexed sequential file by specifying them as the chaining fields. The starting location of the key field is entered in columns 35 through 38 on the file description specifications form.

For example, assume that a master customer file is to be updated with information that is contained on a transaction file and a report of deleted customers is to be printed. The transaction file contains the customer's identification and the amount he has paid on his outstanding balance to date. The master customer file contains all information about the individual customers. The customer records on the master file are to be updated by adjusting those customer balances that the transactions on the transaction file apply to. The common denominator between the transaction file and the master customer file is the customer identification. Since this is the case, chaining can be used; that is, the transaction file can be used as the chaining file by specifying the customer identification field in that file as the chaining field and specifying the master customer file as the chained file.

Figure 13-3 shows the required entries on the various specifications forms for this example if the CHAIN operation is used, and Figure 13-4 shows the required entries if C1 through C9 is used. In both cases, the transaction file is a card file named TRANS, the master customer file is an indexed sequential file named MASTER, the printer file is named PRINT, and IDNO is used as the chaining field.

Figure 13-4 shows the required entries on the various specifications forms for this example if C1 through C9 is used for chaining. As you can see, an additional entry is required on the file description specifications form (an E in column 39 to indicate that additional information is needed on the file extension specifications form) and on the input format specifications form (C1 in columns 61 and 62 to identify IDNO as the chaining field). A file extension specifications form is also required to relate the chaining field on the transaction file (TRANS) to the master file (MASTER). The entries on the calculations specifications form are also different; that is, one specification line rather than two is used.









OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW	SPACE				SKIP			OUTPUT INDICATORS									FIELD NAME	DATA FORMAT P/B/L/R	END POSITION IN OUTPUT RECORD	CONSTANT OR EDIT WORD																	
				TYPE H/D/T/E	A	N	D	BEFORE	AFTER	BEFORE	AFTER	N-NOT	N-NOT	N-NOT	1	2	3	4	5					6	7	8	9	0												
1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	
	0	1	0	MASTIER																																				
	0	2	0																																					
	0	3	0	PRINT																																				
	0	4	0																																					
	0	5	0																																					
	0	6	0																																					
	0	7	0																																					
	0	8	0																																					
	0	9	0																																					
	1	0	0																																					
	1	1	0																																					
	1	2	0																																					
	1	3	0																																					
	1	4	0																																					
	1	5	0																																					

Figure 13-4. Example of Specifying Chaining Using C1 through C9 (Part 3 of 3)

### 13.4.2. Chaining Direct Files

Each record in a direct file has a position on the file relative to the beginning of the file. These positions are called the relative record numbers. They are used in chaining to retrieve records from a direct file by specifying them as chaining fields.

When you want to retrieve records randomly from a direct file you must create a file that contains the relative record numbers of the record you want to retrieve. Then you use this file to chain to the direct file.

Figure 13-5 shows you an example of how to randomly retrieve records from a direct file. In this example, assume that you want to retrieve selected records from a direct file and print them. The file containing the records you want to print (the direct file) is named **DIRECT**, the file that contains the relative record numbers is named **RECSEL**, and the printer file is named **PRELR**.

As you can see, **RECSEL** is defined on the file description specifications form as the input primary file and **DIRECT** as a direct input chained file that is processed randomly by relative record numbers. **RRNO** is defined on the input format specifications form as a numeric field with no decimal places, and the records in the file **DIRECT** are defined as an 80-character field.

The **CHAIN** operation is used on the calculation specifications form with **RRNO** (factor 1) to retrieve the selected records from the file **DIRECT** (factor 2) and each record is subsequently printed by the entries on the output format specifications form.

### 13.4.3. Creating a Direct File Using the CHAIN Operation

If you want to create a direct file, you load records from an input file onto a disk file by relative record numbers. You do this by first defining the direct file to be created as a chained output disk file on the file description specifications form. You then select a numeric field (with no decimal places) in the input record that is to be used to supply the relative record numbers. The name of this field is then used as factor 1 in a **CHAIN** operation that has the name of the output file specified as factor 2. When processing commences, the disk space required for the direct file will be filled with blanks. Then, during calculations, the specified field in the input record will be used as the relative record number for output to the corresponding relative record positions on the disk file. At output time, the entire record will be written on the disk file in the positions corresponding to the relative record number. This process is repeated until all records on the input file have been written on the disk file. When this occurs (end-of-file on the input file), any relative record position on the disk file that does not have a record in it will contain blanks.

Figure 13-6 is an example of the required entries on the various specifications forms for creating a direct file using the **CHAIN** operation. In this example, employee records are contained on a card file named **EMPFIL** and a direct file named **PERSNL** is to be created. The field **EMPNO** is used to supply the relative record numbers.









## 13.5. RECORD ADDRESS FILE PROCESSING

Record address file processing allows you to process an indexed sequential file randomly or between specified limits.

### 13.5.1. Processing an Indexed Sequential File Randomly

When a record address file is used to process an indexed sequential file randomly, the record address file must contain the keys of the records that are to be retrieved from the indexed sequential file. When your program is executed, a record is read from the record address file, the first key in that record is used to retrieve a record from the indexed sequential file. The record from the indexed sequential file is then processed. The next key in the record address file selects the next record to be processed. This continues until the end of the record address file record or a blank field is encountered. When this occurs a new record address file record is read. When the record address file is exhausted (end-of-file) the program terminates.

The following rules apply to a record address file when it is used to randomly process an indexed sequential file:

1. The record keys in the record address file must begin in position 1 of the record and each record key must be contiguous (no intervening blank spaces).
2. The length of the key fields in the record address must be the same in all records.
3. The number of key fields in a record in the record address file may vary; however, a record must contain at least one key field. A blank field that is equal in length to a key will cause the next record in the record address file to be read.
4. The length and format of the key field entries in the record address file must be the same as the key of the indexed sequential file that is being processed.

Figure 13-7 shows the required entries to process an indexed sequential file randomly by using a record address file. Only the entries that will retrieve the records from an indexed sequential file are shown. Additional entries on the input format, calculation, and output format specifications form would be required for a complete program. In this example, the indexed sequential file is named INSEQ, the starting key field location is location 3, and the record address file is named RECAD. The file extension form is required to relate the record address file to the indexed sequential file.







### 13.6. PROCESSING WITH AN ADDRUT FILE

By using an ADDRUT (address output) file, you can process a sequential file in a sequence other than the normal sequence; that is, instead of processing record 1, then record 2, and so on, you can process the records in whatever order you wish. This can be accomplished by creating an ADDRUT (TAG) file that contains the addresses of the records you want to process. The ADDRUT file is created by sorting the file that contains the records you want to process by using the SPERRY UNIVAC OS/3 Sort/Merge with the ADDRUT option. For further information, see the sort/merge user guide. The ADDRUT file is then used to retrieve the records from the sequential file. This feature of RPG II allows you to retrieve records from a given file in a variety of ways by producing several ADDRUT files and then using each ADDRUT file to retrieve records from the original file in a different sequence.

Figure 13-9 shows the required entries to process a sequential file randomly using an ADDRUT file. Only the entries that will retrieve the records from the sequential file are shown. Additional entries on the input format, calculation, and output format specifications form would be required for a complete program. In this example, the ADDRUT file is named TAGS and the file to be processed is named ALLSEQ. On the file description specifications form, TAGS is specified as an input record address file (I in column 15 and R in column 16) and also an ADDRUT file (T in column 32). The E in column 39 indicates that further information is required on the file extension specifications form. ALLSEQ is specified as the primary input file (I in column 15 and P in column 16). Because ALLSEQ is being processed by an ADDRUT file, an R must appear in column 28, and an I in column 31. The R indicates random processing, and the I indicates record identification is used to retrieve records. The entries on the file extension specifications form are used to relate the ADDRUT file (TAGS) to the sequential file (ALLSEQ).

#### FILE DESCRIPTION SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	FILE NAME	FILE TYPE		FILE PROCESSING MODE										EXTENSION OR LINE COUNTER CODE		LABELS		NUMBER OF BYTES IN MAIN STORAGE TO BE RESERVED FOR INDEX													
				FILE DESIGNATION	END OF FILE	KEY OR RECORD ADDRESS FIELD LENGTH	SEQUENCE	FILE ORGANIZATION	FILE ADDRESS TYPE	FILE ORGANIZATION	OVERFLOW INDICATOR	KEY FIELD STARTING LOCATION	DEVICE	NOT USED	NAME OF LABEL EXIT OR NAME OF USER DEVICE ROUTINE	CONTINUATION LINES																	
1	2	3	5	7	13	14	15	16	17	18	19	20	23	24	27	28	29	30	31	32	33	34	35	38	39	40	46	47	52	53	54	59	60
	01	F	TAGS	I	R	F	1100	10	10	T																E	DISC			S			
	02	F	ALLSEQ	I	P	F				R	I																DISC			S			
	03	F																															
	04	F																															

#### FILE EXTENSION SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	RECORD SEQUENCE OF CHAINING FILE		TO FILE NAME	TABLE OR ARRAY NAME	NUMBER OF ENTRIES PER RECORD	NUMBER OF ENTRIES PER TABLE OR ARRAY	LENGTH OF ENTRY	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	AID SEQUENCE	ALTERNATING FOR											
			NUMBER OF THE CHAINING FIELD	FROM FILE NAME									TABLE OR ARRAY NAME	LENGTH OF ENTR										
1	2	3	7	8	9	10	11	18	19	26	27	32	33	35	36	39	40	42	43	44	45	46	51	52
	01	E			TAGS	ALLSEQ																		

Figure 13-9. Example of Processing with an ADDRUT File

## 13.7. DEMAND FILES AND READ OPERATION PROCESSING

Demand files and the READ operation are used to read a record from a file other than the primary or secondary files during calculation time. A record is retrieved during calculation time rather than waiting for the next cycle.

### 13.7.1. Demand File

A demand file can be an input, update, or combined file. It can be processed sequentially, or it can be an indexed sequential file that is processed randomly by a record address file or sequentially between limits by using a record address file or the SETLL operation.

### 13.7.2. READ Operation

Demand files can only be processed by the READ operation. When the READ operation is executed, the record identifying indicator for the record type is set on and the fields are extracted before the next calculation.

### 13.7.3. Using Demand Files and the READ Operation

When demand files and the READ operation are used, the following rules must be observed:

1. When a demand file is used, a D must be placed in column 16 on the line on the file description specifications form where the demand file is specified.
2. Columns 17, 18, 33, and 39 on the file description specifications form must be blank for a demand file.
3. Column 32 on the file description specifications form must be blank or contain X or I when a demand file is used.
4. A demand file is always referenced on the input format and calculation specifications form. If the demand file is an update or combined file, it may be referenced on the output format specifications form.
5. Columns 33 through 42 (factor 2) on the line on the calculation specifications form where the READ operation is specified must contain the name of the demand file. This name must be the same name that was specified for the demand file on the file description specifications form.
6. Columns 58 and 59 on the calculation specifications form may contain any indicator other than HO. The indicator that is specified is set on when the demand file is exhausted (end-of-file). If an indicator is not specified, HO is set on when the demand file is exhausted. Columns 54 through 57 must be blank.
7. A READ operation on the calculation specifications form can be conditioned by control level indicators in columns 7 and 8 and/or other indicators in columns 9 through 17.

8. Look-ahead fields (\*\* in columns 19 and 20) cannot be specified for a demand file on the input format specifications form.
9. Columns 59 and 60 and 61 and 62 cannot be used to specify control levels or matching fields for demand files on the input format specifications form.

### 13.7.3.1. Processing a Demand File

Figure 13-10 shows the required entries on the various specifications forms for a program that processes a demand file with the READ operation. In this example, there are four files specified on the file description specifications form: a combined file named REPNAM that contains employee names, an update/demand file named EMPNO that contains employee number, an output file named CURNAM that is to contain the employee names and numbers, and a display file named DISPLAY. When the operations specified on the calculation specifications form are executed, the READ operation reads a record from the demand file EMPNO each time a record is read from the file REPNAM. If the record read from the demand file (EMPNO) contains a code in record position 8 (indicator 25 is off), the program transfers control to REIT and reads another record from the demand file EMPNO. This process is repeated until a record is found that does not contain a code in record position 8 (indicator 25 is on). A Z is then moved into the field named CODE in the demand file EMPNO. When the demand file EMPNO is exhausted (end-of-file), indicator 12 is set on, which causes a line to be executed and causes the end-of-job message to be displayed.

### CONTROL CARD SPECIFICATIONS

PAGE NO.	LINE NO.	COMPILATION MODE										INVERTED PRINT		ALTERNAT: COLLATING SEQUENCE	
		FORM TYPE	7 23/4 OR BLANK	8 9 10 OR BLANK	11 OR BLANK	12 OR BLANK	13 OR BLANK	14 OR BLANK	15 OR BLANK	16 17	18 19 20	21 22	25	26 27	30
01	0	H													

SUBROUTINE OR ACTION PROGRAM			PROGRAM IDENTIFICATION		
69	70	73	74	75	80
					MAR 6

### FILE DESCRIPTION SPECIFICATIONS

PAGE NO.	LINE NO.	FILE NAME	FILE TYPE		FILE PROCESSING MODE										EXTENSION OR LINE COUNTER CODE	LABELS	NL OF IN ST TO BE I IF	
			FILE DESIGNATION	SEQUENCE	KEY OR RECORD ADDRESS FIELD LENGTH	RECORD ADDRESS TYPE	FILE ORGANIZATION	OVERFLOW INDICATOR	KEY FIELD STARTING LOCATION	OPTION								
01	0	REPNAM	CP	F														
02	0	EMPNO	UD	F	1160	80												
03	0	CURNAM	D	F	1160	80												
04	0	DISPLAY	D	F		10												
05	0	PRINT	D	F	132	132												

Figure 13-10. Example of Processing a Demand File with the READ Operation (Part 1 of 2)





### 13.7.3.2. Processing a Demand File Using the SETLL Operation

A demand file may also be processed between limits by using the SETLL operation. The SETLL operation allows you to establish any record as the next sequential record for an indexed sequential file during calculations. This enables you to initiate or reset the sequential processing at any point within the file independent of the previous position. Consequently, a SETLL operation may be used with a READ operation to retrieve any desired record. Subsequent READ operations will retrieve successive records sequentially until the end of file is reached or another SETLL operation establishes a new next record. If a SETLL operation has not been performed before a READ operation, the first READ operation will retrieve the first record of the file. A SETLL operation may be specified after end-of-file to reset the processing sequence to start at the record indicated by the SETLL operation. The SETLL operation differs from other limits processing in that, each time it is encountered, a new lower limit for subsequent processing is established. Limits processing with a record address file requires that all records within the indicated limits must be read before a new limits record can take effect.

Figure 13-11 is an example of how to use the SETLL operation to perform limits processing.

In this example, the input file CARDINP contains the employee numbers of those employees who have requested their pay checks before their vacations begin. The demand file EMPDATA is a master file that contains the payroll data for each employee. Each time a record is read from the CARDINP file, it is used to specify the next master file record to be read from the EMPDATA file. This is done by specifying the field MANNUM as factor 1 and EMPDATA as factor 2 in the SETLL operation on the calculation specifications form. Processing continues until the CARDINP file is exhausted. Note that the field MANNUM in the file CARDINP has a length of 4, and the key field for the file EMPDATA is a packed field with a length of 2. This may seem to be in conflict with the requirement that in a SETLL operation the key in factor 1 must be the same length as the key specified for the demand file on the file description specifications form. There is no conflict in this case because a 2-character packed field contains 3 digits and a sign, which is the same as a 4-character zoned numeric field (leading or trailing sign separate).







## 13.8. LOOK-AHEAD FIELDS AND FORCE OPERATION PROCESSING

RPG II processes one record at a time. During multifile processing, look-ahead fields provide you with the ability to look at information in the next record of each file that is available for processing. Then, based on this information, calculations can be performed that will condition the execution of a FORCE operation that will override the normal multifile record selection.

### 13.8.1. Look-Ahead Fields

The record in the input area for each file is used to supply the information for the look-ahead fields that are specified on the input format specifications form. Only one record from each file is in the input area at one time; therefore, only one set of look-ahead fields can be specified for any one file.

During multiple file processing, when an input file is selected by your program, the current record is in the input area and the look-ahead fields contain data from the next record in that file. When combined or update files are selected, the current record is then in both the input area and the look-ahead fields. For files not yet selected for processing, the look-ahead fields contain data from the next record in that file.

The following rules must be observed when look-ahead fields are used:

1. Columns 15 and 16 on the input format specifications form must contain an alphabetic sequence entry (AA through ZZ).
2. \*\* must be placed in columns 19 and 20.
3. The format, location, and field name must appear in columns 43 through 58 on the following line.
4. Because all look-ahead fields are available during the processing of a given file, the field names for the look-ahead fields must be unique. If an input field is also used as a look-ahead field, the look-ahead field must have a different name.
5. Columns 21 through 42 and 59 through 74 must be blank on a line that defines a look-ahead field.
6. A look-ahead field cannot be used as a result field on the calculation specifications form, nor can it be blanked-after on the output format specifications form.

### 13.8.2. FORCE Operation

The FORCE operation is specified on the calculation specifications form. It is used to override the normal record selection process during multifile processing; that is, the record that is selected for processing is the next record for the file for which the FORCE operation was issued rather than the record that would normally be selected.

The following rules must be observed when the FORCE operation is used:

1. Columns 18 through 27 and 43 through 59 must be blank on the line where the FORCE operation is specified on the calculation specifications form.
2. The name of the file that is being forced must appear in columns 33 through 42 (factor 2) of the calculation specifications form.
3. A FORCE operation can only be issued at detail time.
4. If a FORCE operation is issued for a file that is exhausted (end-of-file), the normal multifile logic selects the next record for processing.
5. If more than one FORCE operation is issued in a cycle, only the last one is effective.
6. The MR (matching record) indicator is always off when a record from a FORCE operation is processed.

### 13.8.3. Using Look-Ahead Fields and the FORCE Operation

The following subsections illustrate examples of how to use look-ahead fields and the FORCE operation.

#### 13.8.3.1. Controlling Record Selection during Multifile Processing

Figure 13-12 shows the required entries on the input format specifications and calculation specifications forms for a program that uses look-ahead fields and the FORCE operation to select the next record for processing. Only the entries for selecting the next record are shown. Additional entries on the file description, calculation, and output format specifications forms would be required for a complete program. In this example, there are three files to be processed: the primary file named PRIME, the first secondary file named FIRSEC, and the second secondary file named SECSEC. On the input format specifications form, each file is described as having two types of records. The fields in each type of record have the same field name. This is permitted because only one record is available for processing during any cycle. The look-ahead fields are specified once for each file and they reference all records within that file. The look-ahead fields have different names because they are available during the processing of any record type. Each record in the input files has a consecutive sequence number in the look-ahead field. This number is used in conjunction with the compare operations on the calculation specifications form to select a record for processing from the file with the lowest sequence number. On the calculation specifications form, indicators 17, 18, and 20 are used to condition the FORCE operations and indicator 25 is used to set H1 on when duplicate sequence numbers exist.

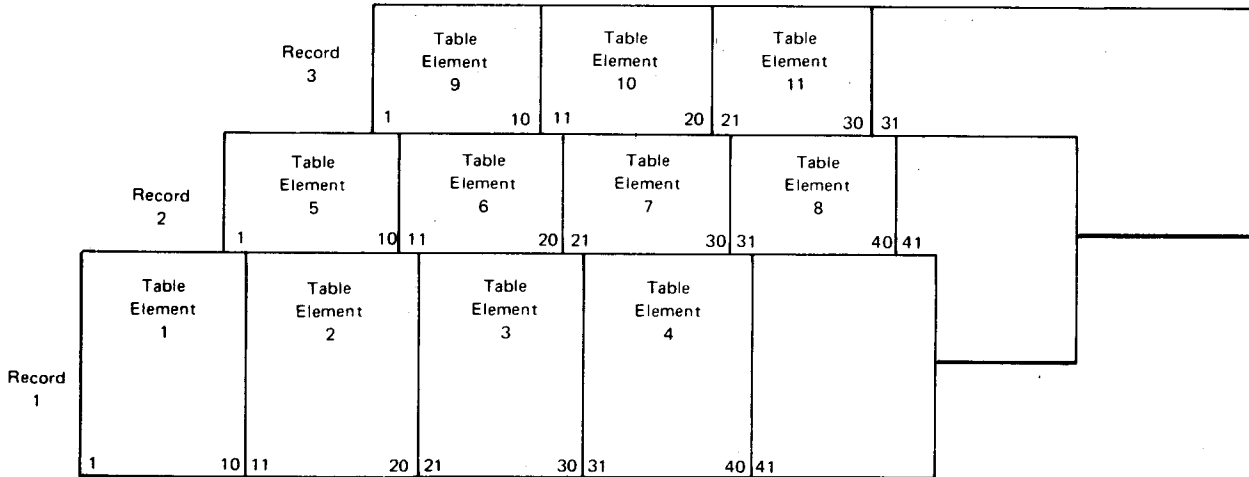




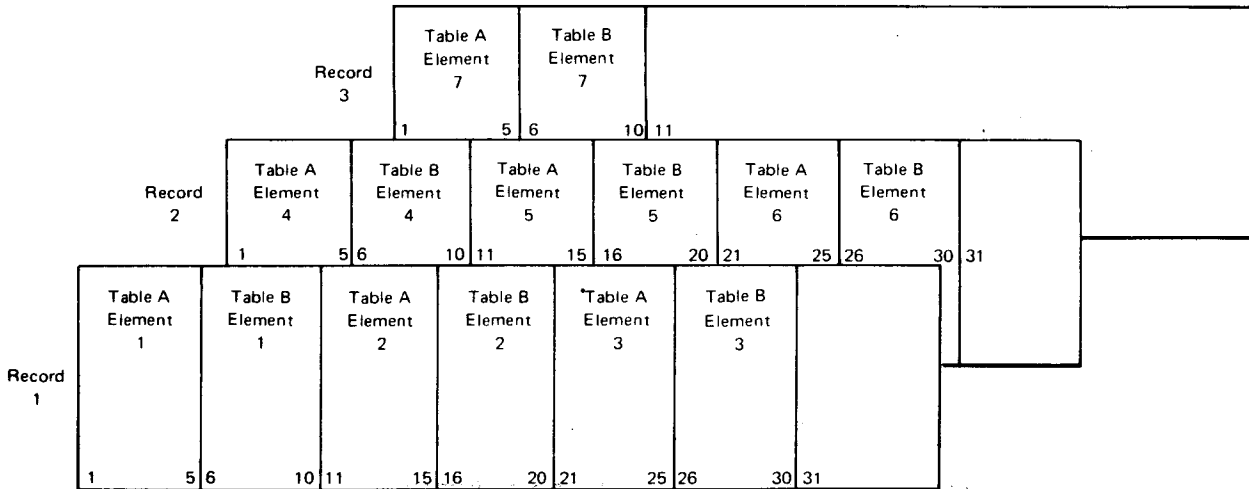








a. Solitary table file



b. Alternating table file

Figure 13-14. Table File Formats

When you use tables in your program, they must conform to the following rules:

1. The first table element in each input record must begin in record position 1, unless the // PARAM COL=7 statement is used. Then the first element begins in record position 7 and the \*\*blank also are in positions 7 through 9.
2. All elements in a table must be the same length.
3. Table elements must be contiguous within each input record. This applies regardless of whether the table file is in solitary or alternating format. The first table element must begin in record position 1, the next element must start in the record position immediately following the end position of the first, and so on.

4. A table element must be contained completely within a record. It cannot be split; that is, part of a table element cannot be in one record and the remainder in the next record.
5. Each input record except the last must contain the same number of table elements. The last input record may contain fewer elements.
6. If a table file is in alternating format, each record in the file must begin (starting in record position 1) with an element from the first table, and it must end with an element from the second table.
7. If you specify sequence checking, the last element for a table in ascending sequence must be the highest in the sequence; for a table in descending sequence, the last element must be the lowest in sequence.
8. Compile time table may be up to 128 characters per record depending on the input device.

### 13.9.2. Table Definition

When you use a table in your program, you must define it by specifying the following information:

- A unique name for it
- When it is to be loaded into main storage
- The number of elements in each record
- The total number of elements in the table
- The length of each element
- The data formats of the elements
- The number of decimal positions in each element (if the data is numeric)
- Whether the file in which the table is contained is in solitary or alternating format

You define all tables on the file extension specifications form. If a single table is involved (solitary format), you define it in columns 27 through 45. If two tables are involved (alternating format), you define the second table in columns 46 through 57.

You can load a table at either compilation time or execution time. The method that you pick governs whether you will need to make additional entries on the file description specifications form.

If you load a table at compilation time, you have to make entries only on the file extension specifications form, because you include the table in the input deck for your source program where it is compiled along with your program and becomes a permanent part of it. In this case, each table that you want to load must be preceded by a record that contains \*\*blank in positions 1 through 3 or in positions 7 through 9 if // PARAM COL=7 is specified. This record, plus the table records, must be placed in the deck following the last source program statement or the last file translation or alternate collating sequence statement in the order in which you specified the tables on the file extension specifications form. A record with /\* in positions 1 and 2 must follow the last table used.

If you want to load a table at execution time, you have to make entries on both the file description specifications form and the file extension specifications form. In this case, entries are required on the file description specifications form because the table is not part of your program. Instead, it is considered to be like any other input data used by your program. Therefore, you must use the file description specifications form to describe the file it is contained on. When you describe an input table file on the file description specifications form, you must specify the file as an input table file that has fixed length records and is a sequential file. You do this by placing an I in column 15, a T in column 16, and an F in column 19, and by leaving column 32 blank. When you do this, the table file will be read into main storage at execution time from the specified card, tape, or disk device.

If your table files are on punched cards and you want to load them at execution time via the control stream reader (CTLRDR), you can do this by placing a record that contains /\$ in positions 1 and 2 before the table records and a record that contains /\* after the table records in each table file. You then place these files in the program execution job control stream immediately after the // EXEC statement in the order in which you specified them on the file extension specifications form. When your program is executed, these files will be read into main storage from the control stream reader. If data is also to be read from the control reader, it should follow the last table.

#### **13.9.2.1. File Description and File Extension Specifications Form Entries for Table Definition**

The required entries for table definition on the file description and file extension specifications forms are summarized in Table 13-2 and Table 13-3, respectively. These tables show you the entries you must make when you use tables in your program.

Table 13—2. Required File Description Entries for Table Definition

Field Name/Column	Entry
File name 7 through 13	If the table is to be loaded at execution time or automatically written on an output file at end-of-job, enter the name of the table file or the output file.
File type 15	If the table is to be loaded at execution time, enter an I in this field. If the table is to be written on an output file at end-of-job, enter an O in this field.
File designation 16	If the table is to be loaded at execution time, enter a T in this field. If the table is to be written on an output file at end-of-job, leave this field blank.
File format 19	Enter an F in this field. This entry is required because all table input and output files must be in fixed format.
Block length 20 through 23	Enter the block length if the table input or output file is blocked. This entry must be a multiple of the record length.
Record length 24 through 27	Enter the record length in this field.
File organization 32	Leave this field blank for both table input and output files. This is required because table files must be sequential files.
Extension or line counter code 39	If the table is to be loaded at execution time, enter an E in this field. This entry is required because you must supply additional information on the file extension specifications form.
Device 40 through 46	Enter the name of the device that the table file is to be read from at execution time or written on at end-of-job.  If the table is on punched cards and you want to load it at execution time via the job control stream reader, enter CTLRDR in this field.

Table 13-3. Required File Extension Entries for Table Definition

Field Name/Column	Entry
From file name 11 through 18	<p>If the table is to be loaded at execution time, enter the file name that is specified in columns 7 through 13 on the file description specifications form.</p> <p>Leave this field blank if the table is to be loaded at compilation time.</p>
To file name 19 through 26	<p>If the table is to be written on an output file at end-of-job, enter the name of the output file that is specified in columns 7 through 13 on the file description specifications form.</p> <p>Leave this field blank if the table is not to be written on an output file at end-of-job.</p>
Table or array name 27 through 32	<p>Enter the name of the table. The table name must be in the form TABaaa, where aaa is from one to three alphanumeric characters.</p>
Number of entries per record 33 through 35	<p>Enter the number of elements per record in the table.</p>
Number of entries per table or array 36 through 39	<p>Enter the total number of elements contained in the table.</p>
Length of entry 40 through 42	<p>Enter the length of each element in the table. The maximum length for a binary element is 9. The maximum length for a numeric element is 15. The maximum length for an alphanumeric element is 256.</p> <p>If L or R is specified in column 43, the length that is specified must include the preceding or following sign.</p>
Data format 43	<p>If the table is to be loaded at execution time and the data in the table elements is in binary format, enter a B. If the data is in packed decimal format, enter a P.</p> <p>If the data in the table elements is preceded by a sign, enter an L. If the data is followed by a sign, enter an R.</p>
Decimal positions 44	<p>If the data in the table elements is numeric, enter the number of decimal positions (0 through 9).</p> <p>Leave this field blank if the data in the table elements is not numeric.</p>
Sequence 45	<p>If the table elements are in ascending sequence, enter an A. If they are in descending sequence, enter a D. An entry is required if you use a LOKUP operation and you specify a high or low resulting indicator.</p>
Table or array name 46 through 51	<p>If the table file is in alternating format, enter the name of the second table. The table name must be in the form TABaaa, where aaa is from one to three alphanumeric characters.</p>
Length of entry 52 through 54	<p>Used with the second table when the table file is in alternating format. The entries are the same as those for columns 40 through 45.</p>
Data format 55	
Decimal positions 56	
Sequence 57	



Lines 020 and 030 show that LSTPAR and CURCHG are also input table files. They differ from PAYINF in that they are read from different types of devices, that is, LSTPAR is read from disk (DISC in columns 40 through 46) and CURCHG is read from tape (TAPE in columns 40 through 46).

Line 040 shows that FINCHG is an output file (O in column 15). The file is a sequential file in fixed unblocked format (F in column 19; columns 20 through 23 are blank; column 32 is blank), the length of each record is 80 characters (80 in columns 24 through 27), and the file is written on tape (TAPE in columns 40 through 46). This file is used to store a table file at end-of-job.

#### ■ File Extension Specifications Form

Line 010 shows that PAYINF is a table file that is loaded at execution time (PAYINF in columns 11 through 18). The file contains two tables in alternating format. The first table is named TABCLS (TABCLS in columns 27 through 32) and the second table is named TABRTE (TABRTE in columns 46 through 51). There are four table elements from each table in each record (4 in columns 33 through 35). Each table consists of 160 elements (160 in columns 36 through 39). The length of each element in the first table (TABCLS) is 10 characters and the elements are in ascending order (10 in columns 40 through 42 and A in column 45). The length of each element in the second table (TABRTE) is 10 digits with no decimal positions (10 in columns 52 through 54 and O in column 56).

Line 020 shows that LSTPAR is a table file that is loaded at execution time (LSTPAR in columns 11 through 18). The file contains a single table named TABPNO (TABPNO in columns 27 through 32). There are eight elements in each record (8 in columns 33 through 35), the table contains 160 elements (160 in columns 36 through 39), and the length of each element is 10 characters (10 in columns 40 through 42).

Line 030 shows that CURCHG is a table file that is loaded at execution time (CURCHG in columns 11 through 18). The file contains two tables in alternating format. The first table is named TABCUS (TABCUS in columns 27 through 32) and the second table is named TABDIS (TABDIS in columns 46 through 51). There are eight table elements from each table in each record (8 in columns 33 through 35). Each table consists of 200 elements (200 in columns 36 through 39). The length of each element in the first table (TABCUS) is five characters (5 in columns 40 through 42). The length of each element in the second table (TABDIS) is five digits, and each element contains three decimal positions (5 in columns 52 through 54 and 3 in column 56). At end-of-job, the tables are written on the output file FINCHG (FINCHG in columns 19 through 26).

Line 040 shows that TABSSN and TABTAX are two tables in alternating format that are loaded at compilation time (columns 11 through 18 are blank; TABSSN is in columns 27 through 32; and TABTAX is in columns 46 through 51). There are five table elements from each table in each record (5 in columns 33 through 35). Each table consists of 60 elements (60 in columns 36 through 39). The length of each element in the first table (TABSSN) is eight characters (8 in columns 40 through 42). The length of each element in the second table (TABTAX) is eight digits with four decimal places (8 in columns 52 through 54 and 4 in column 56).



Line 050 shows that TABCST is a single table that is loaded at compilation time (columns 11 through 18 are blank; TABCST in columns 27 through 32). There are eight table elements in each record (8 in columns 33 through 35). The table consists of 160 elements (160 in columns 36 through 39). The length of each element is 10 digits with 2 decimal places (10 in columns 40 through 42 and 2 in column 44).

### 13.9.3. Using Tables

You access specific information in a table by using the LOKUP operation. This process consists of specifying a search value (called the argument), specifying the search conditions (search the table for a value equal to, next greater than, next less than, greater than or equal to, or less than or equal to the argument), and then searching the table to see whether one of its elements contains a value that meets the search conditions you specified. If a value that meets the search conditions is present, this can be used to set indicators that condition subsequent operations or to cause a data element (called the function) to be selected from a related table. In the latter case, the function that is selected from the related table will be the one that is in the same relative position as the element that met the search conditions on the table that was searched; that is, if the third element meets the search conditions, the third function in the related table will be selected. If no corresponding element exists (the related table is shorter), the related table function is unchanged from its previous value.

The LOKUP operation is specified on the calculation specifications form. The argument is specified in the factor 1 field (columns 18 through 27), LOKUP is specified in the operation field (columns 28 through 32), the name of the table to be searched is specified in the factor 2 field (columns 33 through 42), the name of the related table (if applicable) is specified in the result field (columns 43 through 48), and the search conditions are specified by putting one or two indicators in the resulting indicator fields (high columns 54 and 55, low columns 56 and 57, and equal columns 58 and 59).

If you use a table name in any other operation, it references the last element that was found in a previous LOKUP operation or the first element if no LOKUP operation was performed.

#### 13.9.3.1. Using the LOKUP Operation

Figure 13-16 shows you the actual contents of five tables (TABLE1, TABLE2, and TABLE5 in solitary format; TABLE3 and TABLE4 in alternating format) along with the entries required to define these tables on the file extension specifications form. This is followed by a series of entries on the calculation specifications form that show you several ways you can use the LOKUP operation with these tables and a chart that shows what the result of the operation is in each case.

**TABLE CONTENTS**

	Element 1	Element 2	Element 3	Element 4	Element 5
TABLE1	06	11	13	35	89
TABLE2	40.38	37.47	82.50	64.00	50.51
TABLE3	TTT	KKK	JJJ	EEE	BBB
TABLE4	6	7	2	1	4
TABLE5	FFF	HHH	AAA		

Figure 13-16. LOKUP Operation Examples (Part 1 of 3)



## OPERATION RESULTS

Line Number	Was Element Found?	Indicator Set on	Element That Satisfied Search Conditions	Element Selected From Related Table
010	Yes	10	13	82.50
020	Yes	11	11	37.47
030	Yes	12	35	64.00
040	No			
050	Yes	16	13	
060	Yes	14	13	
070	Yes	18	89	
080	Yes	21	13	JJJ
090	Yes	19	35	EEE
100	No			
110	Yes	24	KKK	
120	Yes	23	KKK	7
130	Yes	28	13	AAA
140	Yes	27	13	AAA
150	Yes	26	JJJ	2
160	No			
170	Yes	29	06	FFF
180	Yes	31	35	Unchanged from previous value
190	Yes	32	89	

Figure 13-16. LOKUP Operation Examples (Part 3 of 3)

**13.9.3.2. Using Data from Table LOKUP**

A typical example of using data from tables is a payroll application in which you have hourly employees whose pay rates are determined by their job classification. In this case, you have to determine what each employee's pay rate is, based upon his job classification, and then multiply the number of hours he worked by the pay rate to calculate his weekly pay.

You can handle this application by including a job classification field on each employee's time card and by using a job classification table and a related pay rate table. Then, you can process each time card and use the job classification as a search argument to search the job classification table for the particular job classification. When you find the job classification, the applicable pay rate will be selected from the related pay rate table and you can use it to calculate the employee's weekly pay.

Figure 13-17 shows the format of the two tables and the required entries on the specifications forms for this application.

Job Classification (TABCLS)	Pay Rate (TABPAY)
01	225
02	290
03	320
04	360
05	400
...	
97	1732
98	1850
99	1976
100	2000

**FILE DESCRIPTION SPECIFICATIONS**

PAGE NO	FORM TYPE	LINE NO	FILE NAME	FILE TYPE				FILE PROCESSING MODE										EXTENSION OR LINE COUNTER CODE		LABELS													
				NOT USED	END OF FILE	SEQUENCE	FILE FORMAT	KEY OR RECORD ADDRESS FIELD LENGTH	RECORD ADDRESS TYPE	FILE ORGANIZATION	OVERFLOW INDICATOR	KEY FIELD STARTING LOCATION	DEVICE	NOT USED	NAME OF LABEL EXIT OR NAME OF USER DEVICE ROUTINE	CONTINUAT																	
1	2	3	5	6	7	13	14	15	16	17	18	19	20	23	24	27	28	29	30	31	32	33	34	35	38	39	40	46	47	52	53	54	59
	F	01	TICARDS				I	P	E	F																							
	F	02	PAYFILE				I	T	F																								

**FILE EXTENSION SPECIFICATIONS**

PAGE NO	FORM TYPE	LINE NO	RECORD SEQUENCE OF CHAINING FILE										TO FILE NAME	TABLE OR ARRAY NAME	NUMBER OF ENTRIES PER RECORD	NUMBER OF ENTRIES PER TABLE OR ARRAY	LENGTH OF ENTRY	ALTERNATING FORMAT															
			01-99 or AA-ZZ	C1-C9	NUMBER OF THE CHAINING FIELD	FROM FILE NAME	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	A/D SEQUENCE	TABLE OR ARRAY NAME	LENGTH OF ENTRY	P/B/L/R DATA FORMAT						DECIMAL POSITIONS	A/D SEQUENCE														
1	2	3	5	6	7	8	9	10	11	18	19	26	27	32	33	35	36	39	40	42	43	44	45	46	51	52	54	55	56	57	58		
	E	01																															
	E	02																															

Figure 13-17. Computing a Payroll with Tables (Part 1 of 2)



### 13.9.3.3. Modifying a Table during Program Execution

You can temporarily change a table during the execution of your program. This can be done by using the table as a result field in an arithmetic or move operation. This causes the appropriate element in the table to be modified for this execution of your program. This change only applies while the program is being executed. It does not permanently change the table. If you want a temporary change to become permanent you can do this by writing the table out at end-of-job.

Figure 13-18 shows an example of how to make a temporary change to tables in alternating format during program execution. In this case, there are two tables called TABCLS and TABPAY. All elements in TABCLS that contain 50 are to be changed to 75, and the related elements in TABPAY are to be changed to 900.

### CALCULATION SPECIFICATIONS

PAGE NO	FORM TYPE	LINE NO	CONDITIONS														CALCULATION										RESULTING INDICATORS										
			INDICATORS														FACTOR 1	OPERATION	FACTOR 2	RESULT FIELD			ARITHMETIC														
			CONTROL LEVEL		AND		AND		NAME	FIELD LENGTH	DECIMAL POSITIONS	HALF ADJUST	COMPARE																								
			LD	LR	N	NOT	N	NOT					1 > 2	1 < 2	1 = 2																						
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	27	28	32	33	42	43	48	49	51	52	53	54	55	56	57	58	59			
	C																50		LOOKUP	TABCLS			TABPAY													20	
	C																	MOVIE	75			TABCLS															
	C																	MOVIE	900			TABPAY															

Figure 13-18. Modifying Table Data during Program Execution

Line 010 causes the constant 50 to be used as the search argument for searching the table TABCLS. If the value is present in an element, indicator 20 is set on and that element and the related element in TABPAY are available for change. Lines 020 and 030 are executed, 75 moves into the TABCLS element, and 900 moves into the TABPAY element.

If you want these changes to be permanent, you can do this by defining an output file on the file description specifications form and then specifying this output file on the file extension specifications form as the file that TABCLS and TABPAY are to be written on at end-of-job.

### 13.9.3.4. Adding Elements to an Existing Table

There are times when you have a table that you know you will have to add to at some later time. You can avoid having to recreate this table when you want to add new data by initially creating this table as a "short" table; that is, you create a table that contains the data elements that you presently have, and you provide space in the table in which to add new data. You do this by initially specifying, on the file extension specifications form, the number of elements you expect to have in the table rather than the number you presently have. When you do this, you will provide space for new data because, when the table is loaded, the unused parts will be filled with zeros if the table is numeric or with blanks if it is alphanumeric.

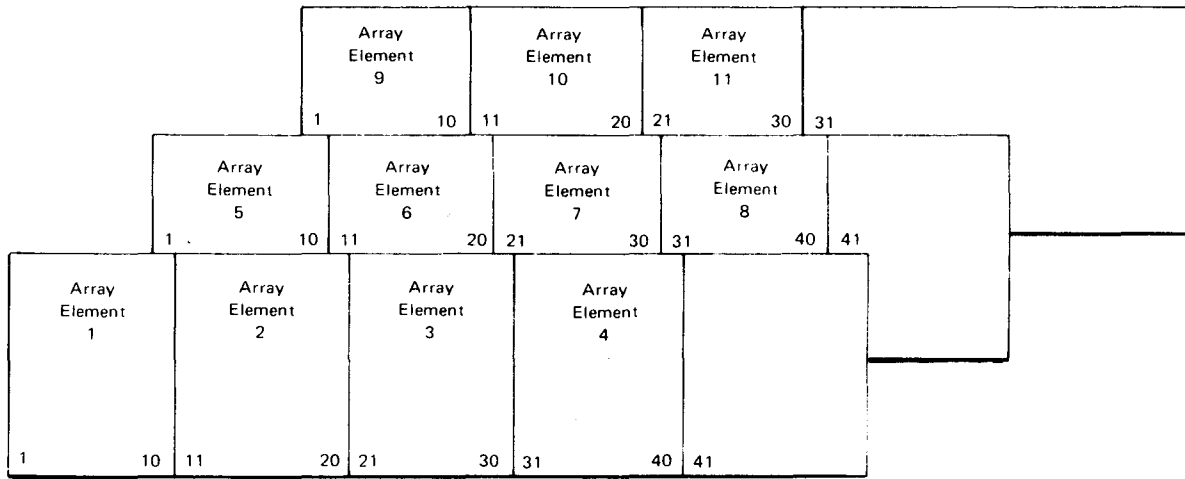


By using arrays, you can substantially reduce the number of statements required in your program when you have calculations that are repeated for fields that have the same characteristics or when you want to check the contents of an array element for one or more specific characteristics.

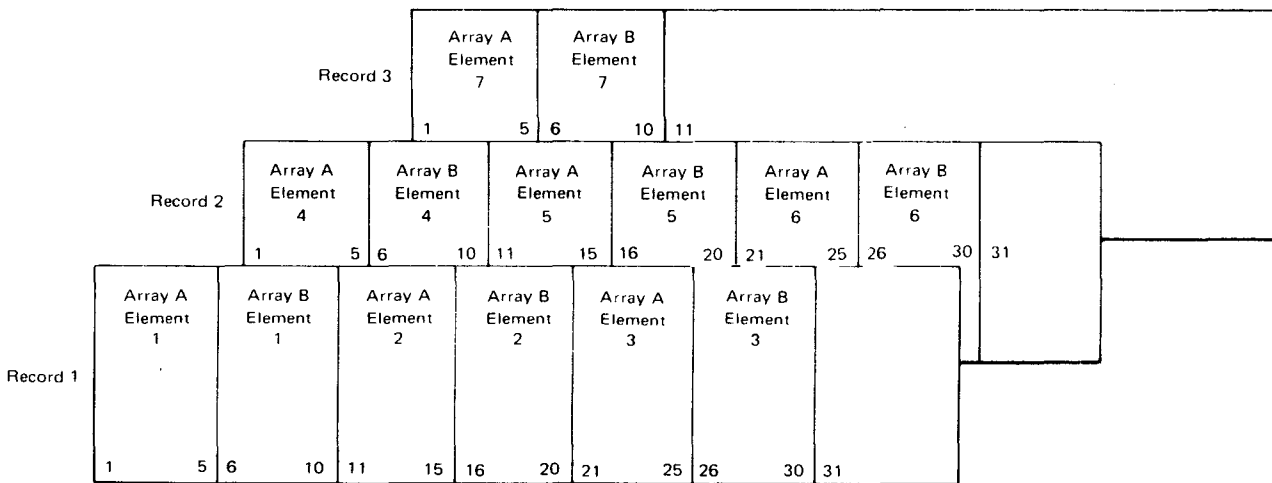
**13.10.1. Array Formats**

Each unit of information in an array is called an element. Each element in an array is identical to the other elements in that it contains the same type of information, is the same length, and has the same number of decimal positions if the array contains numeric data.

An array file can consist of elements from a single array or elements from two arrays. The former is called a solitary array file and the latter an alternating array file. Figure 13-20 shows the format of the two types of array files.



a. Solitary array file



b. Alternating array file

Figure 13-20. Array File Formats



When you use arrays in your program, they must conform to the following rules.

1. The first array element in each input record must begin in record position 1.
2. All elements in an array must the same length.
3. Array elements must be contiguous within each input record. This applies regardless of whether the array file is in solitary or alternating format. The first array element must begin in record position 1, the next element must start in the record position immediately following the end position of the first, and so on.
4. An array element must be contained completely within a record. It cannot be split, that is, part of an array element cannot be in one record and the remainder in the next record.
5. Each input record except the last must contain the same number of array elements. The last input record may contain fewer elements.
6. If an array file is in alternating format, each record in the file must begin (starting in record position 1) with an element from the first array and it must end with an element from the second array.
7. If you specify sequence checking, the last element for an array in ascending sequence must be the highest in the sequence and, for an array in descending sequence, the last element must be the lowest in sequence.
8. An RPG II program produces unpredictable results if it uses an array element that is indexed by a literal number and when the product of this number multiplied by the array element length exceeds 4095.

### 13.10.2. Array Definition

When you use an array in your program, you must define it by specifying the following information:

- A unique name for it
- When it is to be loaded into main storage
- The number of elements in each record
- The total number of elements in the array
- The length of each element
- The data format of the elements
- The number of decimal positions in each element (if the data is numeric)
- Whether the file in which the array is contained is in solitary or alternating format

You define all arrays on the file extension specifications form. If a single array is involved (solitary format), you define it in columns 27 through 45. If two arrays are involved (alternating format), you define the second array in columns 46 through 57. You can load an array at compilation time, during execution, or by using the input format or calculation specifications form. The method that you pick governs whether you need to make additional entries on the file description specifications form.

If you load an array at compilation time, you include the array in the input deck for your source program, where it is compiled along with your program and becomes a permanent part of it. In this case, each array that you want to load must be preceded by a record that contains \*\*blank in positions 1 through 3. This record, plus the array records, must be placed in the deck following the last source program statement or the last file translation or alternate collating sequence statement in the order you specified the arrays on the file extension specifications form. A record with /\* in positions 1 and 2 must follow the last array record.

If you want to load an array at execution time, you have to make entries on both the file description specifications form and the file extension specifications form. In this case, entries are required on the file description specifications form because the array is not part of your program. Instead, it is considered like any other input data used by your program and, as such, you must use the file description specifications form to describe the file on which it is contained. When you describe an input array file on the file description specifications form, you must specify the file as an input array file that has fixed-length records and is a sequential file. You do this by placing an I in column 15, a T in column 16, and an F in column 19 and by leaving column 32 blank. When you do this, the array file will be read into main storage at execution time from the specified card, tape, or disk device.

If your array files are on punched cards and you want to load them at execution time via the control stream reader (CTLRDR), you can do this by placing a record that contains /\$ in positions 1 and 2 before the array records and a record that contains /\* after the array records in each array file. You then place these files in the program execution job control stream immediately after the // EXEC statement in the order you specified them on the file description specifications form. When your program is executed, these files are read into main storage from the control stream reader.

If you load an array (in solitary format only) by using the input format or calculation specifications form, you define the array on the file extension specifications form as you would any other array that is not being loaded at execution time, with the exception that columns 19 through 26 and 33 through 35 must be blank.

#### **13.10.2.1. File Description and File Extension Specifications Form Entries for Array Definition**

The required entries for array definition on the file description and file extension specifications form are summarized in Table 13-4 and 13-5, respectively. These tables show you the entries you must make when you use arrays in your program.

Table 13—4. Required File Description Entries for Array Definition

Field Name/Column	Entry
File name 7 through 13	If the array is to be loaded at execution time or automatically written on an output file at end-of-job, enter the name of the array file or the output file.
File type 15	If the array is to be loaded at execution time, enter an I in this field.  If the array is to be automatically written on an output file at end-of-job, enter an O in this field.
File designation 16	If the array is to be loaded at execution time, enter a T in this field.  If the array is to be automatically written on an output file at end-of-job, leave this field blank.
File format 19	Enter an F in this field. This entry is required because all array input and output files must be in fixed format.
Block length 20 through 23	Enter the block length if the array input or output file is blocked. This entry must be a multiple of the record length.
Record length 24 through 27	Enter the record length in this field.
File organization 32	Leave this field blank for both array input and output files. This is required because array files must be sequential files.
Extension or line counter code 39	If the array is to be loaded at execution time, enter an E in this field. This entry is required because you must supply additional information on the file extension specifications form.
Device 40 through 46	Enter the name of the device the array file is to be read from at execution time or written on at end-of-job.  If the array file is on punched cards and you want to load it at execution time via the job control stream reader, enter CTLRDR in this field.

Table 13-5. Required File Extension Entries for Array Definition

Field Name Column	Entry
From file name 11 through 18	<p>If the array is to be loaded at execution time, enter the file name that is specified in columns 7 through 13 on the file description specifications form.</p> <p>Leave this field blank if the array is to be loaded at compilation time or if it is loaded by using the input format or calculation specifications form.</p>
To file name 19 through 26	<p>If the array is to be written on an output file at end-of-job, enter the name of the output file that is specified in columns 7 through 13 on the file description specifications form.</p> <p>Leave this field blank if the array is not to be written on an output file at end-of-job or the array is to be loaded by using the input format or calculation specifications form.</p>
Table or array name 27 through 32	<p>Enter the name of the array. The array name can consist of from one to six alphanumeric characters. The array name must not begin with TAB. If you intend to use the array name with indexes in your program, you should limit the array name to three alphanumeric characters.</p>
Number of entries per record 33 through 35	<p>Enter the number of elements per record in the array.</p> <p>Leave this field blank if the array is loaded by using the input format or calculation specifications form.</p>
Number of entries per table or array 36 through 39	<p>Enter the total number of elements contained in the array.</p>
Length of entry 40 through 42	<p>Enter the length of each element in the array. The maximum length for a binary element is 9. The maximum length for a numeric element is 15. The maximum length for an alphanumeric element is 256.</p> <p>If L or R is specified in column 43, the length that is specified must include the preceding or following sign.</p>
Data format 43	<p>If the array is to be loaded at execution time and the data in the array elements is in binary format, enter a B. If the data is in packed decimal format, enter a P.</p> <p>If the data in the array elements is preceded by a sign, enter an L. If the data is followed by a sign, enter an R.</p> <p>Leave this field blank if the data in the array elements is in alphanumeric or unpacked numeric format or if columns 33 through 35 are blank.</p>
Decimal positions 44	<p>If the data in the array elements is numeric, enter the number of decimal positions (0 through 9).</p> <p>Leave this field blank if the data in the array elements is not numeric.</p>
Sequence 45	<p>If the array elements are in ascending sequence, enter an A. If they are in descending sequence, enter a D. An entry is required if you use a LOKUP operation and you specify a high or low resulting indicator.</p>
Table or array name 46 through 51	<p>If the array file is in alternating format, enter the name of the second array. The array name can consist of one to six alphanumeric characters. The array name must not begin with TAB. If you intend to use the array name with indexes, you should limit the array name to three alphanumeric characters.</p>
Length of entry 52 through 54	<p>Used with the second array when the array file is in alternating format. The entries are the same as those for columns 40 through 45.</p>
Data format 55	
Decimal positions 56	
Sequence 57	

### 13.10.2.2. Array Definition Examples

Figure 13-21 shows some examples of how to define arrays on the file description and file extension specifications forms.

#### FILE DESCRIPTION SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	FILE NAME	FILE TYPE				FILE PROCESSING MODE						EXTENSION OR LINE COUNTER CODE		LABELS																	
				END OF FILE	SEQUENCE	FILE FORMAT	BLOCK LENGTH	RECORD LENGTH	L/R	KEY OR RECORD ADDRESS FIELD LENGTH	RECORD ADDRESS TYPE	FILE ORGANIZATION	OVERFLOW INDICATOR	KEY FIELD STARTING LOCATION	DEVICE	NOT USED	S/N/E	NAME OF LABEL EXIT OR NAME OF USER DEVICE ROUTINE															
1	2	3	5	7	13	14	15	16	17	18	19	20	23	24	27	28	29	30	31	32	33	34	35	38	39	40	46	47	52	53	54	59	
	F	010	CARAY		I	T	F																										
	F	020	DARAY		I	T	F																										\$
	F	030	EARAY		I	T	F																										\$
	F	040	ARAYOUT				F																										\$

#### FILE EXTENSION SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	RECORD SEQUENCE OF CHAINING FILE		TO FILE NAME	TABLE OR ARRAY NAME	NUMBER OF ENTRIES PER RECORD	NUMBER OF ENTRIES PER TABLE OR ARRAY	LENGTH OF ENTRY	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	A/D SEQUENCE	ALTERNATING FORMAT																				
			NUMBER OF THE CHAINING FIELD	FROM FILE NAME									TABLE OR ARRAY NAME	LENGTH OF ENTRY	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	A/D SEQUENCE																
1	2	3	5	6	7	8	9	10	11	18	19	26	27	32	33	35	36	39	40	42	43	44	45	46	51	52	54	55	56	57			
	E	010			CARAY	AR1	4	200	110	0A				AR2																			
	E	020			DARAY	AR3	8	200	15					AR4																			
	E	030			EARAY	AR5	10	160	8																								
	E	040				AR6	5	100	8																								
	E	050				AR8		120	115																								

Figure 13-21. Array Definition

#### File Description Specifications Form

Line 010 shows that CARAY is an input array file (I in column 15 and T in column 16). The file is a sequential file in fixed unblocked format (F in column 19; columns 20 through 23 and column 32 are blank); the length of each record is 80 characters (80 in columns 24 through 27); additional information about this file is contained on the file extension specifications form (E in column 39); and the file is read from the card reader (READER in columns 40 through 46). In this case, the array file is on punched cards, and it must have a record that contains /\$ in positions 1 and 2 before the array records, as well as a record that contains /\* in positions 1 and 2 after the table records. The file must be placed in your program execution job control stream immediately after the // EXEC statement.

Lines 020 and 030 show that DARAY and EARAY are also input array files. They differ from CARAY in that they are read from different types of devices; that is, DARAY is read from tape (TAPE in columns 40 through 46) and EARAY is read from disk (DISC in columns 40 through 46).

Line 040 shows that ARAYOUT is an output file (O in column 15). The file is a sequential file in fixed unblocked format (F in column 19; columns 20 through 23 and column 32 are blank); the length of each record is 80 characters (80 in columns 24 through 27); and the file is written on disk (DISC in columns 40 through 46). This file will be used to store an array file at end-of-job.

#### ■ File Extension Specifications Form

Line 010 shows that CARAY is an array file that is loaded at execution (CARAY in columns 11 through 18). The file contains two arrays in alternating format. The first array is named AR1 (AR1 in columns 27 through 32) and the second array is named AR2 (AR2 in columns 46 through 51). There are four array elements from each array in each record (4 in columns 33 through 35). Each array consists of 200 elements (200 in columns 36 through 39). The length of each element in both arrays is 10 digits with no decimal places, and the elements are in ascending order (10 in columns 40 through 42 and 52 through 54, 0 in columns 44 and 56, and A in columns 45 and 57).

Line 020 shows that DARAY is an array file that is loaded at execution time (DARAY in columns 11 through 18). The file contains two arrays in alternating format. The first array is named AR3 (AR3 in columns 27 through 32) and the second array is named AR4 (AR4 in columns 46 through 51). There are eight elements from each array in each record (8 in columns 33 through 35). Each array consists of 200 elements (200 in columns 36 through 39). The length of each element in each array is five characters (5 in columns 40 through 42 and 52 through 54). At end-of-job, the arrays are written on the output file ARAYOUT (ARAYOUT in columns 19 through 26).

Line 030 shows that EARAY is an array file that is loaded at execution time (EARAY in columns 11 through 18). The file contains a single array named AR5 (AR5 in columns 27 through 32). There are 10 elements in each record (10 in columns 33 through 35); the table contains 160 elements (160 in columns 36 through 39); and the length of each element is eight digits with three decimal places (8 in columns 40 through 42 and 3 in column 44).

Line 040 shows that AR6 and AR7 are two arrays in alternating format that are loaded at compilation time (columns 11 through 18 are blank; AR6 is in columns 27 through 32; AR7 is in columns 46 through 51). There are five array elements from each array in each record (5 in columns 33 through 35). Each array consists of 100 elements (100 in columns 36 through 39). The length of each element in each array is eight characters (8 in columns 40 through 42 and columns 52 through 54).

Line 050 shows that AR8 is an array that is loaded by using the input format or calculation specifications form (columns 11 through 18 are blank; AR8 is in columns 27 through 32; columns 33 through 35 are blank). The array consists of 120 elements (120 in columns 36 through 39). The length of each element is 15 digits with two decimal places (15 in columns 40 through 42 and 2 in column 44).

### 13.10.3. Using Arrays

You can use arrays in your program on the input format, calculation, and output format specifications. You can access the entire array or individual elements.

If you want to access the entire array, you specify only the name of the array. The array name must be the same as specified on the file extension specifications form.

If you want to access an individual array element, you do this by specifying the array name followed by a comma and an index (a constant that specifies the actual number of the data item or the name of a field with no decimal places that contains the item number). An array name followed by a comma and an index entry cannot exceed 6 characters when used on an input or output specifications form and cannot exceed 10 characters when used in factor 1 or factor 2 on the calculations specifications form. When using the six character limitation, the governing factor is the number of elements in the array.

If you have an array that contains 99 elements, you could give the array a 3-character name, such as ARY, and you would be able (within the 6-character limitation) to access any of the elements because the highest index you could specify is 99 (ARY,99). If you have an array that contains 999 elements, you could not give this array a 3-character name because you must allow for a 3-digit index. If you gave it a 3-character name (such as ARX) and you wanted to access element 100, you could not do this because the specification ARX,100 exceeds six characters. In this case, you would have to change the array name to a 2-character name, such as AX, in order to access element 100; then you could access element 100 by specifying AX,100 or any 2-character name followed by a comma and the index 100.

When you use an array with an index and the index is a constant, the same element is used each time the operation is performed. If the index is a numeric field, the element number in the field will determine which element is used when the operation is performed. In either case, the index value cannot exceed the number of elements in the array. A program exception occurs if the index value exceeds 2,147,483,647. ←

You must also remember that, when you use an array in your program, the characteristics of the array must be the same each time you access it. This means that each element in the array must contain the same number of digits or characters each time you access the array.

#### 13.10.3.1. Loading an Array with the Input Format Specifications Form Using Fixed Indexes

Figure 13-22 shows you an example of how you can load an array with the input format specifications form using fixed indexes. In this example, the array ARY is defined on the file extension specifications form as an array that is loaded by using the input format or calculation specifications form (columns 11 through 18 and 33 through 35 are blank) and that has twenty 5-character elements (20 in columns 36 through 39 and 5 in columns 40 through 42). The input format specifications form shows two methods of loading data into array ARY. The first method is shown on lines 010 and 020. It shows the case where the array elements are contained in consecutive positions within a single input record. In this case you need to specify only the consecutive positions that the array elements occupy within the input record (1 through 50) and the array name (ARY) as shown in line 020 to have the first 10 elements loaded into the array.









As you can see, on the input format specifications form, there are two record types, 07 and 08. Record type 07 contains 10 fields and record type 08 contains 3 fields. These fields are used with the operations on the calculation specifications form to create and load the elements into array ARZ.

On line 010 of the calculation specifications form, the element 1 is loaded into array ARZ by adding the contents of FA to FB. Elements 2, 3, 4, and 5 are loaded on lines 020 through 050 by moving the contents of FC, FD, FE, and FF into the array. Element 6 is loaded on line 060 by adding the contents of element 5 (ARZ,5) to FG. Elements 7 and 8 are loaded on lines 070 and 080 by moving the contents of FH and FI into the array. Element 9 is loaded on line 090 by adding 100 to FJ. Element 10 is loaded on line 100 by subtracting the contents of FA from element 9 (ARZ,9). Elements 11, 12, and 13 are loaded on lines 110 through 130 by moving the contents of FK, FL, and FM into the array. Element 14 is loaded on line 140 by adding the contents of FK and FM. Element 15 is loaded on line 150 by subtracting the contents of FL from element 14 (ARZ,14).

FILE EXTENSION SPECIFICATIONS

PAGE NO	FORM TYPE	LINE NO.	RECORD SEQUENCE OF CHAINING FILE		TO FILE NAME	TABLE OR ARRAY NAME	NUMBER OF ENTRIES PER RECORD	NUMBER OF ENTRIES PER TABLE OR ARRAY	LENGTH OF ENTRY	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	A/D SEQUENCE	ALTERNATING FORMAT																		
			01-99 or AA-ZZ	C1-C9									TABLE OR ARRAY NAME	LENGTH OF ENTRY	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	A/D SEQUENCE														
1	2	3	5	6	7	8	9	10	11	18	19	26	27	32	33	35	36	39	40	42	43	44	45	46	51	52	54	55	56	57	58
	E	010				ARZ			5																						

INPUT FORMAT SPECIFICATIONS

PAGE NO	FORM TYPE	LINE NO.	FILE NAME	RECORD IDENTIFICATION										FIELD DESCRIPTION																			
				RECORD IDENTIFICATION CODES										FIELD LOCATION																			
				SEQUENCE 01-99 or AA-ZZ		NUMBER 1 or N	OPTIONAL RECORD IDENTIFYING INDICATOR OR **	1		2		3		FROM	TO	DECIMAL POSITIONS	FIELD NAME	CONTROL LEVEL															
				A	N			POSITION	N-IND	POSITION	N-IND	POSITION	N-IND																				
14	15	16	17	18	19	20	21	24	25	26	27	28	31	32	33	34	35	38	39	40	41	42	43	44	47	48	51	52	53	58	59		
		010	CALFILE	AA		07		7	7																								
		020																								2		16				FA	
		030																								7		11				FB	
		040																								12		16				FC	
		050																								17		21				FD	
		060																								22		26				FE	
		070																								27		31				FF	
		080																								32		36				FG	
		090																								37		41				FH	
		100																								42		46				FI	
		110																								47		51				FJ	
		120				BB		08		7	7																						
		130																									2		16				FK
		140																								7		11				FL	
		150																								12		16				FM	

Figure 13-24. Loading an Array with the Calculation Specifications Form (Part 1 of 2)

### CALCULATION SPECIFICATIONS

PAGE NO	FORM TYPE	LINE NO	CONDITIONS														CALCULATION										RESULTING INDICATORS									
			INDICATORS														FACTOR 1	OPERATION	FACTOR 2	RESULT FIELD						ARITHMETIC										
			AND AND																	NAME	FIELD LENGTH	DECIMAL POSITIONS	H - HALF ADJUST	COMPARE												
			N-NOT N-NOT N-NOT																					1>2 1<2 1=2												
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	27	28	32	33	42	43	48	49	51	52	53	54	55	56	57	58	59		
		0,1,0		C					0	7								FA	ADD	FB			ARZ <sub>1,1</sub>													
		0,2,0		C					0	7									MOVE	FC			ARZ <sub>1,2</sub>													
		0,3,0		C					0	7									MOVE	FD			ARZ <sub>1,3</sub>													
		0,4,0		C					0	7									MOVE	FE			ARZ <sub>1,4</sub>													
		0,5,0		C					0	7									MOVE	FE			ARZ <sub>1,5</sub>													
		0,6,0		C					0	7							ARZ <sub>1,5</sub>	ADD	FG			ARZ <sub>1,6</sub>														
		0,7,0		C					0	7									MOVE	FH			ARZ <sub>1,7</sub>													
		0,8,0		C					0	7									MOVE	FI			ARZ <sub>1,8</sub>													
		0,9,0		C					0	7							100	ADD	FJ			ARZ <sub>1,9</sub>														
		1,0,0		C					0	7							ARZ <sub>1,9</sub>	SUB	FA			ARZ <sub>1,10</sub>														
		1,1,0		C					0	8									MOVE	FK			ARZ <sub>1,11</sub>													
		1,2,0		C					0	8									MOVE	FL			ARZ <sub>1,12</sub>													
		1,3,0		C					0	8									MOVE	FM			ARZ <sub>1,13</sub>													
		1,4,0		C					0	8							FK	ADD	FM			ARZ <sub>1,14</sub>														
		1,5,0		C					0	8							ARZ <sub>1,14</sub>	SUB	FL			ARZ <sub>1,15</sub>														

Figure 13-24. Loading an Array with the Calculation Specifications Form (Part 2 of 2)

#### 13.10.3.4. Using the LOKUP Operation with Arrays

You can search an array to determine whether it contains a particular data element. As with searching a table, this process consists of specifying a search value (called the argument), specifying the search conditions (search the array for a value equal, next greater than, next less than, greater than or equal to, or less than or equal to the argument), and then searching the array to see whether one of its elements contains a value that meets the search conditions you specified. If a value that meets the search conditions is present, this can be used to set indicators that condition subsequent operations or indicate the location in the array of the element that met the search conditions.

The LOKUP operation is specified on the calculation specifications form. The argument is specified in the factor 1 field (columns 18 through 27); LOKUP is specified in the operation field (columns 28 through 32); the name of the array to be searched is specified in the factor 2 field (columns 33 through 42); and the search conditions are specified by placing one or two indicators in the resulting indicator fields (high columns 54 and 55, low columns 56 and 57, and equal columns 58 and 59). If high or low are used, the array must be in either ascending or descending order. The result field (columns 43 through 48) is not used. The entry in factor 2 may be an array name without an index, an array name with a constant index (literal subscript), or an array name with a numeric field index (variable subscript).

If an array name without an index is specified in factor 2, the search commences at the beginning of the array and ends at the end of the array or when the search conditions you specified are met. With this type of search, you can check the resulting indicators and determine whether a specific element is present in the array or whether the search conditions have been satisfied. There is no means of determining the location of the element that satisfied the search conditions because an index is not specified with the array name. If an array name with a constant index (array name, numeric constant) is specified in factor 2, the search commences at the element whose number is the same as the numeric constant and ends at the end of the array or when the search conditions you specified are met. With this type of search, you can check the resulting indicators and determine whether a specific element is present in the array or whether the search conditions have been satisfied. There is no means of determining the location of the element that satisfied the search condition because the index is a constant and does not change. The main use of this type of search is when you know approximately where the element is that you are searching for in the array. By specifying a constant index that represents the beginning of the area the element is in, you can greatly reduce the search time.

If an array name with a numeric field index (array name, numeric field name) is specified in factor 2, the search commences at the element whose number is contained in the index field and ends at the end of the array or when the search conditions you specified are met. If the search conditions are met, the number of the element that met the search conditions is stored in the index field and the associated resulting indicator is set on. If the search conditions are not met, the index field is set to 1 and the associated indicator is set off. With this type of search, the location (number) of the element is stored in the index field. Consequently, the element that met the search conditions can be used in subsequent operations by specifying the array name and the numeric field index.

Figure 13-25 shows four examples of using the LOKUP operation with arrays. The entries on the file extension specifications form apply to all of the examples.

Line 010 on the file extension specifications form shows that CFILE contains the numeric array CLU. CLU consists of sixty 3-digit elements with no decimal places arranged in ascending sequence, and there are 15 elements in each record.

Line 020 shows that EFILE contains two arrays, FIG and WIN, in alternating format. FIG is a numeric array that contains fifty 5-digit elements with no decimal places arranged in ascending sequence. WIN is an alphanumeric array that contains fifty 4-character elements. Both FIG and WIN have 10 elements in each record.

Line 030 shows that DFILE contains two arrays, DIS and FAM, in alternating format. Both DIS and FAM are numeric arrays that contain one hundred 7-digit elements and each record contains 10 elements from each array. DIS is in ascending sequence.



The circled numbers on the calculation specifications form refer to the following examples:

■ Example 1 – Lines 010 and 020

Starting with the first element, the array CLU is searched for an element that is equal to UNOT. If an element is found, indicator 66 is set on. If an element is not found, indicator H3 is set on. These entries show you whether an element equal to UNOT is present in the array, but they will not indicate which element meets the search conditions because the array name (CLU) was specified without an index in factor 2.

■ Example 2 – Lines 040 through 090

This example shows how you can reduce the search time by limiting the search to a particular area of the array. Assume that you know that element 30 contains the value 791. By comparing MIDV with 791, you can determine whether MIDV is equal to element 30. If it is, indicator 68 is set on and the program branches around the LOKUP operations (line 050). If MIDV does not equal element 30, a LOKUP operation is specified for each half of the array. If MIDV is less than element 30, indicator 67 is set on and line 060 is executed. This search commences at the beginning of the array (CLU,1). If MIDV is not less than or equal to element 30, indicator 67 is set off and line 070 is executed. This search commences at element 31 of the array (CLU,31). If a value equal to the search argument MIDV does exist in the array CLU, indicator 68 is not set on and indicator H8 is set on (on line 080). These entries show whether an element equal to MIDV is present in the array, but they will not indicate which element meets the search conditions because the array was specified with a constant index (CLU,1 and CLU,31) and this index does not change.

■ Example 3 – Lines 110 through 150

On lines 110 and 120, the contents of the numeric fields A and B are set to 1. These are then used as numeric field indexes for the arrays in the operations on lines 130 through 150.

On line 130, CNT is the search argument. The array to be searched is DIS. The search commences with the first element of DIS (since the index field, A, has been set to 1) and searches for a value that is equal to CNT. If this value is found, A is set equal to the number of the element that met the search conditions and indicator 25 is set on. If this value is not found, indicator 25 is not set on and A contains 1 after the LOKUP operation is completed.

On line 140, the element in the FAM array that corresponds to the element DIS,A found in the previous LOKUP operation is used as the search argument. The FIG array is searched for a value equal to FAM,A. If this value is found, B is set equal to the number of the element that met the search conditions and indicator 20 is set on. This operation is performed only if the previous LOKUP operation on line 130 resulted in an equal condition.

On line 150, the data in the element in the WIN array that corresponds to the element FIG,B found in the previous LOKUP operation is moved to the THEL field. This operation is performed only if the previous LOKUP operation on line 140 resulted in an equal condition.

#### ■ Example 4 - Lines 170 through 200

In this example, elements are added to the FIG and WIN arrays. On line 170, the contents of numeric field A is set to 1. This is then used as a numeric field index for the operations on lines 180 through 200.

On line 180, 00000 is used to search the FIG array for the first vacant element. If an element equal to 00000 is found, A is set equal to the number of the element that met the search conditions and indicator 40 is set on. A now contains the number of the first vacant element in FIG and its corresponding WIN array.

On lines 190 and 200, the new element (FIGNEW) is inserted in the vacant element in the FIG array and the new element WINNEW is inserted in the corresponding vacant element in the WIN array.

#### 13.10.3.5. Using Arrays to Format Output Records

Figure 13-26 shows an example of how to use arrays to format your output records.

In this example, there are three arrays: AR1, AR2, and AR3. On the file extension specifications form, these arrays are defined as arrays that are loaded by using the input format specifications or calculation specifications form (columns 11 through 18 and columns 33 through 35 are blank). On line 010, AR1 is defined as a numeric array that has four 5-digit elements with no decimal places. On line 020, AR2 is defined as an alphanumeric array that has five 10-character elements. On line 030, AR3 is defined as a numeric array having six 4-character elements with two decimal places.

The input format specifications form shows that there are two record types: 01 and 02. Record type 01 contains array AR1 in record positions 3 through 22 and array AR3 in record positions 55 through 78. Record type 02 contains array AR2 in record positions 3 through 52 and array AR3 in record positions 55 through 78. The output format specifications form shows that array AR3 and the first element of array AR1 are to be included in an output record, as are array AR3 and an element from array AR2 (identified by the numeric field index Z). Each element in array AR3 is to be edited by the edit word 0△.△△&CR (△ represents blank), and the element from array AR1 is to be zero suppressed. To illustrate this, we will assume that the contents of the first two records are:

<u>Record Type</u>	<u>Array</u>	<u>Contents</u>
01	AR1	21345768902134576890
	AR3	02134567890132457689687L (L is interpreted as a minus 3)
02	AR2	BOB△JACE△△DICK△SMITH△BILL△ BOYD△△△JOE△ZEED△BEN△TABOR
	AR3	02134567890132457689687L (L is interpreted as a minus 3)







As you can see, by using arrays, you can substantially reduce the number of statements you must write. In Figure 13-27, arrays are not used and it takes 27 statements to produce the desired results. In Figure 13-28, where arrays are used, it takes only 15 statements.

### CALCULATION SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	CONDITIONS															CALCULATION									
			CONTROL LEVEL L0-L9/LR/SR	INDICATORS															FACTOR 1	OPERATION	FACTOR 2	RESULT FIELD					
				AND					AND					NAME	FIELD LENGTH	DECIMAL POSITIONS											
				A	Z	N-NOT	N-NOT	N-NOT	A	Z	N-NOT	N-NOT	N-NOT														
7	8	9	10	11	12	13	14	15	16	17	18	27	28	32	33	42	43	48	49	51	52	53	54	55			
		0,10	C															FIELD,1	ADD	L1TOT,1				L1TOT,1	17	Z	
		0,20	C															FIELD,2	ADD	L1TOT,2				L1TOT,2	17	Z	
		0,30	C															FIELD,3	ADD	L1TOT,3				L1TOT,3	17	Z	
		0,40	C															FIELD,4	ADD	L1TOT,4				L1TOT,4	17	Z	
		0,50	C					L1										L1TOT,1	ADD	L2TOT,1				L2TOT,1	17	Z	
		0,60	C					L1										L1TOT,2	ADD	L2TOT,2				L2TOT,2	17	Z	
		0,70	C					L1										L1TOT,3	ADD	L2TOT,3				L2TOT,3	17	Z	
		0,80	C					L1										L1TOT,4	ADD	L2TOT,4				L2TOT,4	17	Z	
		0,90	C					L2										L2TOT,1	ADD	L3TOT,1				L3TOT,1	17	Z	
		1,00	C					L2										L2TOT,2	ADD	L3TOT,2				L3TOT,2	17	Z	
		1,10	C					L2										L2TOT,3	ADD	L3TOT,3				L3TOT,3	17	Z	
		1,20	C					L2										L2TOT,4	ADD	L3TOT,4				L3TOT,4	17	Z	

Figure 13-27. Calculating Totals without Arrays (Part 1 of 2)

### OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	STACKER SELECT/ F=FETCH OVERFLOW	SPACE				SKIP		OUTPUT INDICATORS						FIELD NAME	EDIT CODES	B-BLANK AFTER	DATA FORMAT P/B/L/R	END POSITION IN OUTPUT RECORD	CODES													
				TYPE H/D/T/E	FILE NAME	A	N	D	BEFORE	AFTER	BEFORE	AFTER	N-NOT	AND	N-NOT						AND	N-NOT	NONE	CR	-									
1	2	3	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45			
	0,1	O		REPORT		T	3																											
	0,2	O																																
	0,3	O																																
	0,4	O																																
	0,5	O																																
	0,6	O								T	3																							
	0,7	O																																
	0,8	O																																
	0,9	O																																
	1,0	O																																
	1,1	O								T	3																							
	1,2	O																																
	1,3	O																																
	1,4	O																																
	1,5	O																																

Figure 13-27. Calculating Totals without Arrays (Part 2 of 2)

### FILE EXTENSION SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	RECORD SEQUENCE OF CHAINING FILE								TO FILE NAME	TABLE OR ARRAY NAME	NUMBER OF ENTRIES PER RECORD	NUMBER OF ENTRIES PER TABLE OR ARRAY	LENGTH OF ENTRY	P/B/L/R DATA FORMAT	DECIMAL POSITIONS	A/D SEQUENCE	ALTERNATING FILE								
			01 99 or AA ZZ	C1-C9	NUMBER OF THE CHAINING FIELD	FROM FILE NAME	TABLE OR ARRAY NAME	LEN	C EN'																		
1	2	3	5	6	7	8	9	10	11	18	19	26	27	32	33	35	36	39	40	42	43	44	45	46	51	52	
	0,1	O																									
	0,2	O																									
	0,3	O																									

Figure 13-28. Calculating Totals with Arrays (Part 1 of 2)







## 13.11. PROCESSING WITH ALTERNATE COLLATING SEQUENCE

A collating sequence is the order in terms of value that the characters in a data processing system's character set have in relation to one another. This means that a specific value is assigned to each character in the character set so that, when you compare one character to another, that character will be higher or lower in value than the character it was compared to.

### 13.11.1. RPG II Collating Sequence

The normal collating sequence for RPG II is the EBCDIC collating sequence shown in Table 13-6. If you examine Table 13-6 you can see that the letters A, B, and C have been assigned the hexadecimal values C1, C2, and C3 respectively; consequently, the letter B is higher in value than the letter A but lower in value than the letter C.

If the normal collating sequence does not meet your specific requirements, you can change it by defining an alternate collating sequence.

### 13.11.2. Defining an Alternate Collating Sequence

If you need to use an alternate collating sequence in your program, you indicate this by entering an S in column 26 of the control card specifications form. Then, at compilation time, you include one or more records that contain the changes to be made to the normal collating sequence in your compilation input deck.

The records that contain these changes must be in the following format:

<u>Position</u>	<u>Entry</u>
1 through 6	ALTSEQ. This indicates that this is an alternate collating sequence record.
7 and 8	Blank
9 and 10	Enter the hexadecimal value (Table 13-6) that corresponds to the character whose normal collating sequence is being changed.
11 and 12	Enter the hexadecimal value that corresponds to the character that is to replace the character specified in columns 9 and 10.
13 through 80	These positions can be used in the same manner as columns 9 and 10, and 11 and 12 to specify additional changes to the normal collating sequence. Each 4-position set defines an additional change.



Table 13-6. RPG II Collating Sequence (Part 1 of 3)

Binary Value	EBCDIC Character	Hexa- decimal Value
00000000		00
00000001		01
00000010		02
00000011		03
00000100		04
00000101		05
00000110		06
00000111		07
00001000		08
00001001		09
00001010		0A
00001011		0B
00001100		0C
00001101		0D
00001110		0E
00001111		0F
00010000		10
00010001		11
00010010		12
00010011		13
00010100		14
00010101		15
00010110		16
00010111		17
00011000		18
00011001		19
00011010		1A
00011011		1B
00011100		1C
00011101		1D
00011110		1E
00011111		1F
00100000		20
00100001		21
00100010		22
00100011		23
00100100		24
00100101		25
00100110		26
00100111		27
00101000		28
00101001		29
00101010		2A
00101011		2B
00101100		2C
00101101		2D
00101110		2E
00101111		2F
00110000		30

Binary Value	EBCDIC Character	Hexa- decimal Value
00110001		31
00110010		32
00110011		33
00110100		34
00110101		35
00110110		36
00110111		37
00111000		38
00111001		39
00111010		3A
00111011		3B
00111100		3C
00111101		3D
00111110		3E
00111111		3F
01000000	Blank or space	40
01000001		41
01000010		42
01000011		43
01000100		44
01000101		45
01000110		46
01000111		47
01001000		48
01001001		49
01001010	¢	4A
01001011	.	4B
01001100	<	4C
01001101	(	4D
01001100	+	4E
01001111		4F
01010000	-	50
01010001		51
01010010		52
01010011		53
01010100		54
01010101		55
01010110		56
01010111		57
01011000		58
01011001		59
01011010	!	5A
01011011	\$	5B
01011100	*	5C
01011101	)	5D
01011110	:	5E
01011111	^	5F
01100000	-	60
01100001	/	61

Table 13-6. RPG II Collating Sequence (Part 2 of 3)

Binary Value	EBCDIC Character	Hexa-decimal Value
01100010		62
01100011		63
01100100		64
01100101		65
01100110		66
01100111		67
01101000		68
01101001		69
01101010		6A
01101011		6B
01101100	%	6C
01101101	-	6D
01101110	>	6E
01101111	?	6F
01110000		70
01110001		71
01110010		72
01110011		73
01110100		74
01110101		75
01110110		76
01110111		77
01111000		78
01111001	'	79
01111010	:	7A
01111011	#	7B
01111100	@	7C
01111101	'	7D
01111110	=	7E
01111111	"	7F
10000000		80
10000001		81
10000010		82
10000011		83
10000100		84
10000101		85
10000110		86
10000111		87
10001000		88
10001001		89
10001010		8A
10001011		8B
10001100		8C
10001101		8D
10001110		8E
10001111		8F
10010000		90
10010001		91
10010010		92
10010011		93
10010100		94
10010101		95
10010110		96
10010111		97
10011000		98
10011001		99
10011010		9A
10011011		9B
10011100		9C

Binary Value	EBCDIC Character	Hexa-decimal Value
10011101		9D
10011110		9E
10011111		9F
10100000		A0
10100001	~	A1
10100010		A2
10100011		A3
10100100		A4
10100101		A5
10100110		A6
10100111		A7
10101000		A8
10101001		A9
10101010		AA
10101011		AB
10101100		AC
10101101		AD
10101110		AE
10101111		AF
10110000		B0
10110001		B1
10110010		B2
10110011		B3
10110100		B4
10110101		B5
10110110		B6
10110111		B7
10111000		B8
10111001		B9
10111010		BA
10111011		BB
10111100		BC
10111101		BD
10111110		BE
10111111		BF
11000000	{	C0
11000001	A	C1
11000010	B	C2
11000011	C	C3
11000100	D	C4
11000101	E	C5
11000110	F	C6
11000111	G	C7
11001000	H	C8
11001001	I	C9
11001010		CA
11001011		CB
11001100		CC
11001101		CD
11001110		CE
11001111		CF
11010000	}	D0
11010001	J	D1
11010010	K	D2
11010011	L	D3
11010100	M	D4
11010101	N	D5
11010110	O	D6
11010111	P	D7

Table 13-6. RPG II Collating Sequence (Part 3 of 3)

Binary Value	EBCDIC Character	Hexadecimal Value
11011000	Q	D8
11011001	R	D9
11011010		DA
11011011		DB
11011100		DC
11011101		DD
11011110		DE
11011111		DF
11100000	\	E0
11100001		E1
11100010	S	E2
11100011	T	E3
11100100	U	E4
11100101	V	E5
11100110	W	E6
11100111	X	E7
11101000	Y	E8
11101001	Z	E9
11101010		EA
11101011		EB
11101100		EC
11101101		ED
11101110		EE
11101111		EF
11110000	0	F0
11110001	1	F1
11110010	2	F2
11110011	3	F3
11110100	4	F4
11110101	5	F5
11110110	6	F6
11110111	7	F7
11111000	8	F8
11111001	9	F9
11111010		FA
11111011		FB
11111100		FC
11111101		FD
11111110		FE
11111111		FF

You can use the entire record to specify collating sequence changes. If you have more than one change in a record, the 4-position change sets must be contiguous. If a blank position appears between two change sets, the changes following the blank position will be ignored. You can use as many change records as needed, provided they are in the proper format.

When you include alternate collating sequence records in your compilation input deck, these records must be preceded by a record that contains **\*\*blank** in positions 1 through 3. This record, plus the alternate collating sequence records, must be placed in the deck following any file translation records. A record with **\*\*blank** in positions 1 through 3 or **/\*** in positions 1 and 2 must follow the last alternate collating sequence record.

If you define an alternate collating sequence, it will affect how sequence checking, matching field operations, and alphanumeric comparison operations are performed. It will not, however, affect numeric comparison operations, look-up operations, or table or array sequence checking; and no data will be altered.

### 13.11.3. Using an Alternate Collating Sequence

As you know, you can use an alternate collating sequence when the normal collating sequence does not meet your needs. For example, you could define an alternate collating sequence when you want two characters to be considered equal, when you want to alter the normal collating sequence by inserting a character between two existing characters, or when you want to change the position of characters in the collating sequence.

#### 13.11.3.1. Causing Characters to Be Considered Equal

Figure 13-30 shows how you can use an alternate collating sequence to cause characters to be considered equal. In this example, assume that you want to have a blank considered equal to a zero. To have these characters compare equally, they must appear in the same position in the collating sequence. If you examine Table 13-6, you will see that the hexadecimal value for blank is 40 and the hexadecimal value for zero is F0. You then must define an alternate collating sequence in which blank is replaced with zero by placing 40 (blank) in positions 9 and 10 and F0 (zero) in positions 11 and 12 in an ALTSEQ record.

1	10	20	30	40	50	60
ALTSEQ△△40F0						

Figure 13-30. Causing Characters to Be Considered Equal

If you defined this alternate collating sequence, the field △△△△ would be considered as being equal to 0000.

### 13.11.3.2. Inserting a Character between Two Existing Characters in the Collating Sequence

Figure 13-31 shows how you can use an alternate collating sequence to insert a character between two existing characters in the collating sequence. In this example, assume that you want to insert an asterisk (\*) between the characters A and B in the collating sequence. In this case, the asterisk must take the letter B's position. If you examine Table 13-6, you will see that the hexadecimal value for asterisk is 5C and the hexadecimal value for B is C2. You then must define an alternate collating sequence in which asterisk takes B's position by placing 5C in columns 9 and 10 and C2 in columns 11 and 12 in an ALTSEQ record. At this point, the asterisk has taken B's position; however, the desired change requires that each successive character, beginning with B, must take the position of the next higher character in the collating sequence. This means that B must take C's position, C must take D's position, and so on until I occupies the position normally held by the hexadecimal value CA, for which there is no printable character. This requires the following additional entries in the ALTSEQ record:

<u>Column</u>	<u>Entry</u>	<u>Explanation</u>
13 and 14, 15 and 16	C2C3	B(C2) takes C's (C3) position
17 and 18, 19 and 20	C3C4	C(C3) takes D's (C4) position
21 and 22, 23 and 24	C4C5	D(C4) takes E's (C5) position
25 and 26, 27 and 28	C5C6	E(C5) takes F's (C6) position
29 and 30, 31 and 32	C6C7	F(C6) takes G's (C7) position
33 and 34, 35 and 36	C7C8	G(C7) takes H's (C8) position
37 and 38, 39 and 40	C8C9	H(C8) takes I's (C9) position
41 and 42, 43 and 44	C9CA	I(C9) takes the unprintable character CA's position.

1	10	20	30	40	50	60
ALTSEQ△△5CC2C3C3C4C4C5C5C6C6C7C7C8C8C9C9CA						
asterisk (*)		B takes C's position,				
takes B's		C takes D's position,				
position.		and so on.				

Figure 13-31. Inserting a Character between Two Existing Characters in the Collating Sequence

If you defined this alternate collating sequence, an asterisk has a higher value than A and a lower value than B because the collating sequence is A, \*, B, C, D, E, F, G, H, I instead of A, B, C, D, E, F, G, H, I.

### 13.11.3.3. Changing the Position of Characters in the Collating Sequence

Figure 13-32 shows how you can use an alternate collating sequence to change the position of characters in the collating sequence. In this example, assume that you want to change the position of the letters A and B so that B appears before A. In this case, you must reverse the positions of these letters in the collating sequence. If you examine Table 13-6, you will see that the hexadecimal value for A is C1 and the hexadecimal value for B is C2. You then must define an alternating collating sequence in which A is replaced by B and B is replaced by A. You do this by placing C1(A) in positions 9 and 10, C2(B) in positions 11 and 12, C2(B) in positions 13 and 14, and C1(A) in positions 15 and 16 in an ALTSEQ record.

1	10	20	30	40	50	60
ALTSEQ△△C1C2C2C1						

Figure 13-32. Changing the Position of Characters in the Collating Sequence

If you defined this alternate collating sequence, A has a greater value than B because the collating sequence is B,A,C,...,X,Y,Z, instead of A,B,C,...,X,Y,Z.

## 13.12. PROCESSING WITH FILE TRANSLATION

File translation is the ability to translate any character into another character. This means that you can cause a specific character to be automatically changed to whatever other character you choose by defining a file translation table.

### 13.12.1. Defining a File Translation Table

If you need to use file translation in your program, you indicate this by entering an F in column 43 of the control card specifications form. Then, at compilation time, you include one or more records that contain the file translation table in your compilation input deck.

The format of the records that contain the file translation table depends upon whether the table is to apply to all files in your program or only to selected files.

→ If the table is to apply to one or all files in your program, the format of the translation table record is:

<u>Position</u>	<u>Entry</u>
1 through 8	Enter *FILES△△ if the translation table applies to all files in your program. Enter the file name if the translation table applies to one file in your program.
9 and 10	Enter the hexadecimal value (Table 13-6) that corresponds to the character that is to be translated on input or translated to on output.

<u>Position</u>	<u>Entry</u>
11 and 12	Enter the hexadecimal value that corresponds to the internal character that RPG II will work with. This character replaces the character in positions 9 and 10 on input and is replaced by the character in positions 9 and 10 on output.
13 through 80	These positions can be used in the same manner as columns 9 and 10, and 11 and 12, to specify additional character translations. Each 4-position set defines an additional character translation. As with positions 9 and 10, and 11 and 12, the first two positions specify the character to be translated on input or translated to on output, and the second two positions specify the internal character RPG II will work with, that is, the character that replaces the character in the first two positions on input and is replaced by the character in the first two positions on output.

You can use the entire record to specify character translations. If you have more than one character translation in a record, the 4-position translation sets must be contiguous. If a blank position appears between two translation sets, the translation sets following the blank position will be ignored. You can use as many translation table records as needed, provided they are in the proper format.

If the table is to apply to one or more selected files in your program, two types of translation table records are required. The first type specifies the files that are involved and the second specifies the characters to be translated. The format of a record that specifies which files are involved is:

<u>Position</u>	<u>Entry</u>
1 through 8	*EQUATE△
9 through 80	Enter the names of the files (from columns 7 through 13 of the file description specifications form) to be translated. If more than one file is involved, a comma must separate each file name.

You can use the entire record to specify file names that are associated with the translation table. When a file name is encountered that is not followed by a comma, it indicates that this is the last file to be associated with the translation table. A file name cannot be split between records. It must be specified completely in one record. You can use as many file specification records as needed, provided they are in the proper format.

After you have specified the files that are involved, you must then specify the character to be translated. You do this by preparing one or more records in the following format:

<u>Position</u>	<u>Entry</u>
1 through 8	*EQUATE△
9 and 10	Enter the hexadecimal value (Table 13-6) that corresponds to the character that is to be translated on input or translated to on output.
11 and 12	Enter the hexadecimal value that corresponds to the internal character RPG II will work with. This character will replace the character in positions 9 and 10 on input and be replaced by the character in positions 9 and 10 on output.
13 through 80	These positions can be used in the same manner as columns 9 and 10, and 11 and 12, to specify additional character translations. Each 4-position set defines an additional character translation. As with positions 9 and 10, and 11 and 12, the first two positions specify the character to be translated on input or translated to on output, and the second two positions specify the internal character RPG II will work with, that is, the character that replaces the first two positions on input and is replaced by the character in the first two positions on output.

You can use the entire record to specify character translations. If you have more than one character translation in a record, the 4-position translation sets must be contiguous. If a blank position appears between two translation sets, the translation sets following the blank positions will be ignored. You can use as many translation table records as needed, provided they are in the proper format.

When you include file translation table records in your compilation input deck, these records must be preceded by a record that contains \*\*blank in columns 1 through 3. This record, plus the file translation table records, must be placed in the deck immediately following the last source program statement. A record with \*\*blank in columns 1 through 3 or /\* in columns 1 and 2 must follow the last file translation table record.

If you define a file translation table, a character from each record in a file is replaced internally by the translation character whether it is an input, output, update, or combined file. This includes tables or arrays loaded at execution time.

When you have a combined or update file, the file is translated at both input and output time. If an update file is involved, each record must be written before the next record is read.

When you use a file translation table with a chaining or record address file, the translated data must be the data you intend to retrieve.



### 13.12.2. Using a File Translation Table

As you know, you can use a file translation table in your program when you need to translate specific characters into other characters during input or output. For example, you could define a file translation table if you had a file that contained coded confidential data, such as wholesale prices, and you wanted to:

- translate this data into its actual value to perform calculations with it in your program; or
- translate specific characters into other characters for more than one file but not all files in your program.

#### 13.12.2.1. Using File Translation to Translate Characters in a File

Figure 13-33 shows an example of how you can use a file translation table to translate characters to other characters in a file.

In this example, assume that the file CONFILE contains billing information in the form of the retail price for an item and the coded wholesale price. The wholesale price code consists of a 10-character alphabetic code in which letters are substituted for numbers. In this case, assume that LAMPEWICKS is the code and the substitution is:

L	A	M	P	E	W	I	C	K	S
1	2	3	4	5	6	7	8	9	0

If you want to use the wholesale price in your program calculations, you must define a translation table so that these letters are translated into numbers when they are read. The hexadecimal values for the wholesale price code letters and the numbers that they are to be translated to are:

Wholesale Code	Hexadecimal Value	Equivalent Number	Hexadecimal Value
L	D3	1	F1
A	C1	2	F2
M	D4	3	F3
P	D7	4	F4
E	C5	5	F5
W	E6	6	F6
I	C9	7	F7
C	C3	8	F8
K	D2	9	F9
S	E2	0	F0

The translation table that you supply must translate L to 1, A to 2, M to 3, and so on. You do this by placing D3(L) in positions 9 and 10, F1(1) in positions 11 and 12, C1(A) in positions 13 and 14, F2(2) in positions 15 and 16, D4(M) in positions 17 and 18, F3(3) in positions 19 and 20, and so on.

1	10	20	30	40	50	60
CONF ILE△D3F1C1F2D4F3D7F4C5F5E6F6C9F7C3F8D2F9E2F0						

Figure 13—33. Using File Translation to Translate Characters in a File

If you define this translation table, CC.KA translates as 88.92.

If you wanted this translation table to apply to all the files in your program, you could substitute \*FILES△△ for CONF ILE△ in positions 1 through 8.

### 13.12.2.2. Using File Translation for More than One but Not All Files

Figure 13—34 shows an example of how you can use a file translation table in your program when you need to translate specific characters into other characters in more than one but not all files in your program.

In this example, assume that you have four files named AFILE, BFILE, CFILE, and DFILE. When you encounter a left parenthesis, (, or a right parenthesis, ), in any of these files, they are to be changed to a left brace, {, and a right brace, }, respectively. To do this, you must first define the files that are involved by specifying a \*EQUATE record that lists these files. Then, you must define a \*EQUATE record that defines a translation table that translates ( to {, and ) to }, by placing 4D(( ) in positions 9 and 10 and CO({) in positions 11 and 12, and 5D( ) in positions 13 and 14 and DO(}) in positions 15 and 16.

1	10	20	30	40	50	60
*EQUATE△AFILE,BFILE,CFILE,DFILE						
*EQUATE△4DC05DD0						

Figure 13—34. Using File Translation to Translate Characters for More than One but Not All Files

If you define this translation table, the field (99976) translates as {99976}.

## 13.13. PROCESSING WITH WORKSTATIONS

A workstation is an interactive device with a screen for displaying dialog text and a keyboard for entering input.

One or more workstations can concurrently enter data into an executing RPG II program.

When you use the workstation, you must enter your input on screen formats created with the screen format generator. These screen formats must correspond to the input/output fields you define on the input and output specifications.

To execute a workstation program, you enter standard job control statements first. A program can contain only one workstation file, but you can assign up to 255 workstations to it using the standard // DVC ... // LFD job control sequence.

You can connect all, some, or none of the 255 workstations to the workstation file when the program is executed. If none of the workstations are connected when the program opens the workstation file, the program waits for the first workstation to be connected by the CONNECT operation and then displays the first screen. While the program is executing, you can connect other workstations and then disconnect them when you no longer need them. However, once you connect the first workstation and then disconnect it along with all other workstations you may have added, the following occurs:

- For primary files (13.3.1): the last record (LR) indicator is set on and the program terminates.
- For demand files (13.7.1): the end-of-file indicator is set on.

All input you enter at the workstation should produce output from the program so that the operator knows that the input was processed. For multivolume workstation files, the operator should always wait for the return message after the input is entered before going to system mode; otherwise, other workstations connected to the file may be delayed with their transactions.

The end-of-file function key (function key 15) disconnects your workstation from your program. ←

You must observe the following restrictions when you use a workstation file:

- You can only use one workstation file per program.
- You must specify the workstation file as a combined file.
- If you specify the workstation file as a primary file, you can't use secondary files in the program.
- If you specify the workstation file as a demand file, primary and secondary files are not required.
- You can't use control level indicators, matching fields, or look-ahead fields.
- You can't use the first page indicator (1P).
- You can't use output before you enter input because the workstation identification field used internally by RPG II doesn't contain a value until after an input transaction is completed. When you use multivolume workstation files, the output goes to the workstation that last transmitted input.

The first record generated by RPG II as input from each workstation is blank. The program recognizes this blank record and displays the initial screen format. Depending on the screen formats you generate, it is possible that other records may also be blank.

You should provide some means of identifying the source of the input data from the data itself. This is necessary because each workstation transaction could involve several input/output sequences using the same or different screen formats. Therefore, you should separate the activities of the workstations within the program.

### 13.13.1. File Description Specifications for Workstation Files

The following entries are valid for the file description specifications for a workstation file:

<u>Column</u>	<u>Entry</u>
6	Must contain F.
7 through 13	Must contain the file name.
15	Must contain a C for combined files.
16	Must contain a P for a primary file or a D for a demand file. When you specify a primary file, you can't use any secondary files in the program.
17 and 18	Must be blank.
19	Must contain an F for a fixed-length record.
20 through 23	Blank or equal to the record length.
24 through 27	Length of the longest record. It must be equal to the highest end position you specified on the input or output specification. This entry doesn't apply to the screen size.
28 through 39	Must be blank.
40 through 46	Must contain the word WORKSTN, identifying the file as a workstation file. You can use only one workstation file per program, but you can assign up to eight workstations to it.
47 through 70	Must be blank.
71 and 72	Enter an external indicator (U1 through U8) if needed.
73 and 74	Must be blank.
75 through 80	Enter a program identification if needed.

### 13.13.2. Input Format Specifications for Workstation Files

The following special considerations apply to the input format specifications for a workstation file:

<u>Column</u>	<u>Entry</u>
19 and 20	Look-ahead fields. Must be blank. You can't use look-ahead fields. (See 13.8.)
19 through 41	Must identify each different record type (or screen type) including the initial blank screen.
44 through 51	Must contain the location of the field in the input record, not the location on the screen. The location of a particular input field depends on the number and length of the fields that precede it in the input record. You must know the format of the input record created for each screen format.
59 and 60	Must be blank. You can't use control level indicators.
61 and 62	Must be blank. You can't use matching fields.

The screens created by the screen format generator must correspond to the input fields you define on the input format specifications.

### 13.13.3. Calculation Specifications for Workstation Files

The following special considerations apply to the calculations specifications for a workstation file:

<u>Column</u>	<u>Entry</u>
28 through 32	Must contain the READ operation in order for demand files to input data. You must use the READ operation before the first output occurs at the workstation. The first record read is blank.

### 13.13.4. Output Format Specifications for Workstation Files

The following special considerations apply to the output format specifications for a workstation file:

<u>Column</u>	<u>Entry (on output file identification)</u>
23 through 31	Must not contain the first page indicator (1P).
40 through 43	Blank

<u>Column</u>	<u>Entry (on first field description following an output file identification)</u>
---------------	---

23 through 31	Must be blank.
---------------	----------------

42 and 43	Enter Kn, where n is the length of the format name.
-----------	---

45 through 54	Must contain a format name. One format name is required for each output record for the workstation file. You can't use more than one format name per record. You must enclose the format name within apostrophes.
---------------	---

<u>Column</u>	<u>Entry (on other field descriptions)</u>
---------------	--

40 through 43	End position. The end position refers to the end position of the field in the output record, not to the end position of the field as it appears on the screen. The fields in the output record start in position 1 and must appear in the order expected by the screen format.
---------------	--

You can't use output before you enter input. When you use multivolume files, the output goes to the workstation that last transmitted input.

The screens created by the screen format generator must correspond to the output fields you define on the output format specification.

### 13.13.5. Input/Output Errors for Workstation Files

When an input/output error occurs with a workstation file, the \*ERROR field contains the character 4 (X'F4') and the HO indicator is set on. When using a demand file, reset the HO indicator to continue processing.

### 13.13.6. Sample Program for a Workstation File

You use the following commands at the workstation to create and compile your program:

```
LOGON user-id
EDT
.
. RPG II source program
.
@WRITE INPUT1
// JOB COMPIL
//RPG01 RPG,IN=INPUT1
/&
@WRITE $$JCS
@HALT
RV COMPIL
LOGOFF
```

In the sample program, a MIRAM file is accessed from a workstation. The MIRAM file contains a list of names and addresses and is indexed by name. You access the MIRAM file with the CHAIN operation by using the name as a key.

You enter a name at the workstation on the screen format ENTRNAME (Figure 13-35) created with the screen format generator.

```
ENTER NAME .....
```

Figure 13-35. Screen Format ENTRNAME for Sample Workstation Program

If the name you enter is in the MIRAM file, the address information is displayed with the screen format ADDRFRMT (Figure 13-36):

```
NAME:      ROBERT L. SMITH
ADDRESS:   1234 ANY STREET
           ANYTOWN, ST 12345

ENTER NAME .....
```

Figure 13-36. Screen Format ADDRFRMT for Sample Workstation Program

If the name you enter is not in the MIRAM file, the error screen format NOFIND (Figure 13-37) is displayed:

```
NAME:      TOMMY CARTRE
NOT FOUND IN FILE

ENTER NAME .....
```

Figure 13-37. Screen Format NOFIND for Sample Workstation Program

The program processes names until all workstations are disconnected from the workstation file. When this happens, the last record indicator (LR) is set on by RPG II and the program terminates.

Figure 13-38 gives a sample program for this workstation file.







The following explains the coding in Figure 13-38.

<u>Line Number</u>	<u>Explanation</u>
1	The control card specification contains the program name ADDCHK.
2	The first file description specification describes the workstation file. It is the primary file and is a combined file.
3	The second file description specification describes the MIRAM file named MAST, which contains the names and addresses.
4	The input format specification specifies that indicator 01 is the record identifying indicator for the blank record. This blank record is the first record read from each workstation.
5 and 6	The input format specification for the name you enter from the workstation.
7 through 11	The input format specification for the records from the chained MIRAM file.
12	The calculation specification for the MIRAM file CHAIN operation, using the key named NAME.
13 through 18	The output format specifications that display the screen format named ADDRFRMT, which contains the address information retrieved from the MIRAM file.
19 through 21	The output format specifications that display the error screen format named NOFIND when the name you enter is not found in the MIRAM file.
22 through 24	The output format specifications that display the input screen format named ENTRNAME.

### 13.14. KEY SPECIFICATION WITH MIRAM FILES

You can process MIRAM (CDM) files using single-key or multikey structures. Once a file is created using a particular structure, any programs that access it must use the same key structure.

There are two ways of specifying a single-key MIRAM file:

- Enter the key length (columns 29 and 30), file type (columns 31 and 32), and key location (columns 35 through 38), on the file description specifications form.
- Enter the continuation line option entry (columns 54 through 59), key location (columns 60 through 65), key length (columns 66 and 67), and the key options (columns 68 and 69) on the file description specifications continuation statement.

In both cases, the key specified is the primary key of reference.

For a multikey MIRAM file, enter the key descriptions on the file description specifications continuation statements (5.2.18). You must use the continuation statement when you create a file that has more than a single key structure, and the specification sequence of key structure must be KEY1 through KEY5 (5.2.18.2). ←

When you use the continuation statements, you must leave columns 29 and 30, and 35 through 38 blank. Columns 31 and 32 must contain an entry.

For multikey MIRAM files, the first key specified on the continuation statements is used as the primary key of reference.

Use the SETK operation (7.3.3.8.2) to select the key structure that will be used to retrieve records from the indexed file.

#### **13.14.1. Processing a Multikey MIRAM File Sequentially by Key**

A multikey MIRAM file may be processed sequentially by a key structure other than the primary key structure. This requires the use of the SETK operation. The program example in Figure 13-38A shows that the indexed MIRAM file is printed sequentially by KEY2.

Lines 2 through 4 give the definition of the file. Line 3 shows that KEY1 is 20 positions long and begins in position 1. Line 4 shows that KEY2 is 5 positions long and begins in position 21. This file is also the primary input file.

The first record read from the file is by KEY1, but this record is not printed because indicator 01 is set off at line 11 the first time as a result of the calculations. Indicator 01 is also used to control the printing (lines 13 and 14). The SETK operation on line 10 sets KEY2 as the key of reference.

Since the SETK operation sets the file to the beginning of KEY2, this operation is executed only once, the first time through the calculations.

The next record read from the file is the first record read sequentially for KEY2. This is the first record printed since indicator 01 is set off only on the first cycle.

Similarly, the remaining records are read sequentially by KEY2 and printed.



## 13.15. PROCESSING WITH DATA STRUCTURES

You use data structures to:

- Allow multiple definitions of internal areas using different data formats
- Subdivide data fields so that you can reference either the entire field or its subfields
- Group fields to make referencing easier

A data structure is an alphanumeric byte string that starts on a double-word boundary. It initially contains blanks. If you specify a compile-time array as a subfield, the array data is positioned in the data structure. Similarly, subfields that redefine an input field contain input data after an input operation. ←

Subfields that redefine a result field, however, are left as blanks. Therefore, if you define any of the subfields as numeric, they must be initialized with numeric data before they are used in any calculation operations.

You can redefine subfields in the data structure by specifying the same information in columns 44 through 51 (to/from location) of the input specifications form for another subfield.

You must specify the name of an input field or a result field that you want to redefine in the data structure. The name doesn't need to immediately precede the subfields that redefine it. When you redefine an input field, the information you specify in columns 44 through 51 (to/from location) is relative to the beginning of the data structure, not to the positions that the field occupies in the input record.

If you use a field as a data structure name or as a data structure subfield name, the reserved space for that field is in the data structure no matter where you defined the field.

You can give a subfield the same length attributes as other subfields or fields. An alphanumeric subfield can be up to 256 characters in length; a numeric subfield can be up to 15 characters in length.

If you specify an array as a subfield, the length must be the same as the amount of main storage required to store the entire array.

You must observe the following restrictions when using data structures:

- A data structure can't be more than 9999 characters in length.
- Data structure entries must be specified last on the input specifications form.
- You define the length of a data structure in two ways:
  - If you specify the data structure name as a field in an input record, the data structure length must be the same as the length of the input field. If the to-location (columns 48 through 51) you specify for a subfield is longer than the length you defined for the input field, the input field specification is invalid. ←

- If you don't specify the data structure name as a field in an input record, the data structure length is defined by the highest to-location (columns 44 through 47) you specified for a subfield.

- You can't use look-ahead fields as a data structure or a subfield.
- You can't specify packed or binary numeric fields as a subfield within a data structure. You can define the fields as packed or binary in a file. When the field is placed in the data structure, RPG II converts it to zoned decimal format and it remains as such.
- You can't specify reserved words, array elements, or table names as a subfield.
- ➔ ■ You can't subdivide numeric subfields of data structures.
- If you define an internal area more than once and both alphanumeric (unpacked) and numeric (packed) data formats are present, be careful of which operations you perform, since both formats can't coexist. The format must be alphanumeric or numeric, but not both.

### 13.15.1. Input Format Specifications for Data Structures

The following entries are valid for the input format specifications for a data structure. (All columns not defined must be blank.)

<u>Column</u>	<u>Entry</u>
6	Must contain I.
7 through 13	May contain the name of the data structure (maximum of six characters). It must meet the requirements of a field name. You reference the data structure name as: <ul style="list-style-type: none"> <li>■ A field on the input or output format specifications form</li> <li>■ An RLABL operation on the calculation specifications form</li> <li>■ A SAVDS or INFDS name in columns 60 through 65 of the file description specifications continuation line for a WORKSTN file</li> </ul>
19 through 20	Enter DS, which indicates this is a data structure.

### 13.15.2. Input Format Specifications for Subfields of a Data Structure

The following entries are valid for the input format specifications for the subfields of a data structure. You must specify these entries on the line below the data structure specification. (All columns not defined must be blank.)

↓

<u>Column</u>	<u>Entry</u>
6	Must contain I.
44 through 47	Enter the number of the record position in which the subfield begins (from-location). This number is relative to the beginning of the data structure.
48 through 51	Enter the number of the record position in which the subfield ends (to-location). This number is relative to the beginning of the data structure.
44 through 50	Enter *OPCODE, *RECORD, *SIZE, *STATUS, *MODE, *INP, or *OUT to define the reserved, self-defining subfields for the file information data structure (INFDS).
52	Enter the number of digits from 0 through 9 that are to the right of the decimal point if the subfield is numeric.  Leave this column blank if the subfield is alphanumeric.
53 through 58	Enter the subfield name. This allows the program to reference the subfields you defined in columns 44 through 50. The subfield name can: <ul style="list-style-type: none"><li>■ Be identical to an input name or a result field name</li><li>■ Appear as an RLABL operation</li><li>■ Be used in factor 1, factor 2, or as an output field</li></ul> The subfield name cannot: <ul style="list-style-type: none"><li>■ Be specified in different data structures</li><li>■ Be specified as part of another data structure</li></ul> Subfield names must meet the requirements of a field name.

↑

### 13.16. PROCESSING WITH INTERACTIVE DATA ENTRY

When you specify the CONSOLE device in columns 40 through 46 on the file description specifications form, the operator can enter input to an executing RPG II program from a workstation. Workstation prompts generated from the field name on the input format specifications form prompt the operator to enter the data. To display the prompts, the operator must enter a function key.

If the CONSOLE file is a record address file, then RPG II forms the name of the screen format by adding FM at compilation time to the program name you specified in columns 75 through 80 of the control specifications form. In other words, if you name the program ACCT, the name of the screen format is ACCTFM.

For other CONSOLE files, the record identifying indicator is added to the program name so that each format is distinct. In other words, if you name the program ACCT and it contains three record types with indicators 01, 02, and 03, the names of the screen formats are:

- ACCT01
- ACCT02
- ACCT03

If you used OR lines on the input specifications form to identify the same record, the record is associated with only one screen format.

Control information on the top line of the workstation screen allows the operator to identify the current record and specify the next record type to be prompted. The remaining lines on the workstation screen are used for the formatted record. For each field you specify, 14 characters are reserved that define the field name and its attributes. Therefore, you must limit the record length to 1518 characters. The size and number of fields in a record determine the format that is generated for the screen format. Figure 13-39 shows a sample prompt screen generated when you press function key 5.

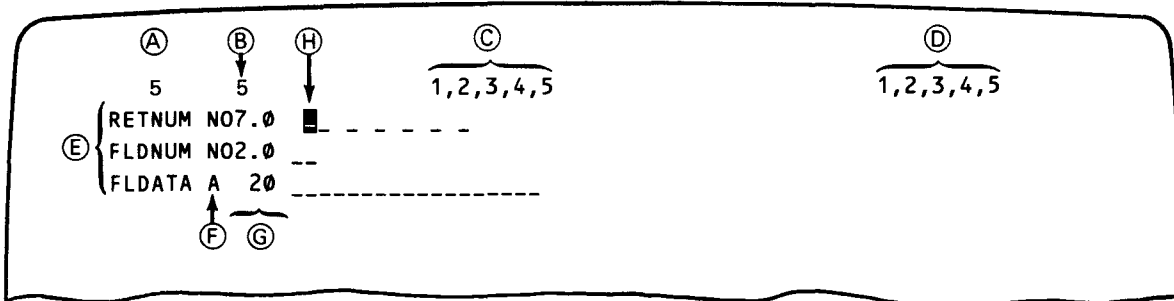


Figure 13-39. Function Key 5 Prompt Screen



Letters in the following explanation correspond to the contents of the prompt screen generated by function key 5. The codes, indicators, field names, field types, and field lengths shown in Figure 13-39 are obtained from the input description specification form.

- Ⓐ Record identification code for prompted record (columns 21 through 27, input description specification)
- Ⓑ Record identifying indicator for prompted record (columns 19 and 20, input description specification)
- Ⓒ Record identifying indicators for all record types selected before data is entered into current screen (columns 19 and 20, input description specification)
- Ⓓ Record identifying indicators for all record types selected after data is entered in current screen format
- Ⓔ Prompted field names for screen 5 (columns 53 through 58, input description specification)
- Ⓕ Type of field: alphanumeric (S) or numeric (N) – (column 52, input description specification)
- Ⓖ Field length or field length and decimal positions (columns 44 through 51, input description specification)
- Ⓗ Cursor indicates the next entry position.

RPG II generates four possible prompt screen formats:

■ One-column screen format

Results when from 1 to 23 fields are described on the input specification form for a record of the console file. Each alphanumeric field is from 1 to 66 characters in length. Each numeric field is from 1 to 15 digits in length, with a maximum of nine decimal places. The maximum record size possible is 1518 for a single-column format. (This is also the maximum allowable record size for any input record under interactive data entry.) Figure 13-40 shows a sample 1-column screen format created by the screen format generator. A 14-character prompt with the field name (from one to six characters) and field type (alphanumeric or numeric) always results.

X	X	1,2,3,4,5,6,7,8,9,10	1,2,3,4,5,6,7,8,9,10
F1	N01.0	9	
F2	N02.0	99	
F3	N03.0	999	
F4	N04.0	9999	
F5	N05.1	99999	
F6	N06.2	999999	
F7	N07.3	9999999	
F8	N08.4	99999999	
F9	N09.5	999999999	
F10	N10.6	9999999999	
F11	N11.7	99999999999	
F12	N12.8	999999999999	
F13	N13.0	9999999999999	
F14	N14.0	99999999999999	
F15	N15.9	999999999999999	
F16	A 16	XXXXXXXXXXXXXXXXXX	
F17	A 17	XXXXXXXXXXXXXXXXXX	
F18	A 18	XXXXXXXXXXXXXXXXXX	
F19	A 19	XXXXXXXXXXXXXXXXXX	
F20	A 39	XX	
F21	A 50	XX	
F22	A 66	XX	
F23MAX	A 66	XX	

Figure 13-40. One-Column Prompt Screen (23 Fields)

■ Two-column screen format

Results when from 24 to 46 fields are described. Figure 13-41 shows a sample 2-column prompt screen having 24 fields. When field lengths exceed 26 characters, the format automatically adjusts and does not allow 46 fields on the screen (Figure 13-42).

2	2	1,2,3,4,5,6,7	1,2,3,4,5,6,7		
FL1	N01.0	-	FL2	N02.0	--
FL3	N03.0	---	FL4	N04.0	----
FL5	N05.1	-----	FL6	N06.2	-----
FL7	N07.3	-----	FL8	N08.4	-----
FL9	N09.5	-----	FL10	N10.6	-----
FL11	N11.7	-----	FL12	N12.8	-----
FL13	N13.0	-----	FL14	N14.0	-----
FL15	N15.9	-----	FL16	A 16	-----
FL17	A 17	-----	FL18	A 18	-----
FL19	A 19	-----	FL20	A 26	-----
FL21	A 26	-----	FL22	A 12	-----
FL23	A 21	-----	FL24	A 26	-----

Figure 13-41. Two-Column Prompt Screen (24 Fields)

2	2	2			2		
FL1	N01.0	-			FL2	N02.0	---
FL3	N03.0	---			FL4	N04.0	-----
FL5	N05.1	-----			FL6	N06.2	-----
FL7	N07.3	-----			FL8	N08.4	-----
FL9	N09.5	-----			FL10	N10.6	-----
FL11	N11.7	-----			FL12	N12.8	-----
FL13	N13.0	-----			FL14	N14.0	-----
FL15	N15.9	-----			FL16	A 16	-----
FL17	A 17	-----			FL18	A 18	-----
FL19	A 19	-----			FL20	A 26	-----
FL21	A 33	-----					
FL22	A 50	-----					
FL23	A 06	-----			FL24	N15.9	-----
FLD25	A 11	-----			FLD26	N11.3	-----
FLD27	A 06	-----			FLD28	N06.3	-----
FLD29	A 21	-----			FLD30	A 05	-----
FLD31	A 02	---			FLD32	A 12	-----
FLD33	A 17	-----			FLD34	A 03	---
FLD35	N05.0	-----			FLD36	A 26	-----

Figure 13-42. Two-Column Prompt Screen, Self-Adjusting Fields

■ Three-column screen format

Results when from 47 to 69 fields are described. Figure 13-43 shows a sample 3-column prompt screen having 47 fields. When any field length exceeds 12 characters, the format automatically adjusts.

4	4	1,2,3,4,5,6,7			1,2,3,4,5,6,7		
FA1	A 12	-----	FALD2	N12.6	-----		
FA4	N12.4	-----	FA5	N01.0	---		
FA7	N12.4	-----	FA8	N06.2	-----		
FA10	A 12	-----	FA11	A 12	-----		
FA13	A 12	-----	FA14	N01.0	---		
FA16	N12.7	-----	FA17	A 05	---		
FA19	A 12	-----	FA20	A 11	-----		
FA22	N12.4	-----	FA23	A 06	-----		
FA25	A 12	-----	FA26	A 12	-----		
FA28	N12.5	-----	FA29	N12.3	-----		
FA31	A 12	-----	FA32	A 12	-----		
FA34	A 12	-----	FA35	A 12	-----		
FA37	A 12	-----	FA38	A 12	-----		
FA40	A 12	-----	FA41	A 12	-----		
FA43	A 11	-----	FA44	N01.0	---		
FA46	N05.0	-----	FALD47	A 12	-----		
					FA3	A 12	-----
					FA6	A 12	-----
					FA9	A 07	-----
					FA12	A 12	-----
					FA15	N12.0	-----
					FA18	N12.8	-----
					FA21	A 12	-----
					FA24	N01.0	---
					FA27	A 12	-----
					FA30	A 12	-----
					FA33	A 12	-----
					FA36	A 12	-----
					FA39	A 12	-----
					FA42	A 12	-----
					FA45	N06.2	-----

Figure 13-43. Three-Column Prompt Screen (47 Fields)

Four-column screen format

Results when from 70 to 80 fields are described. Figure 13-44 shows a sample 4-column prompt screen having 70 fields. The format automatically adjusts when any field length exceeds six characters.

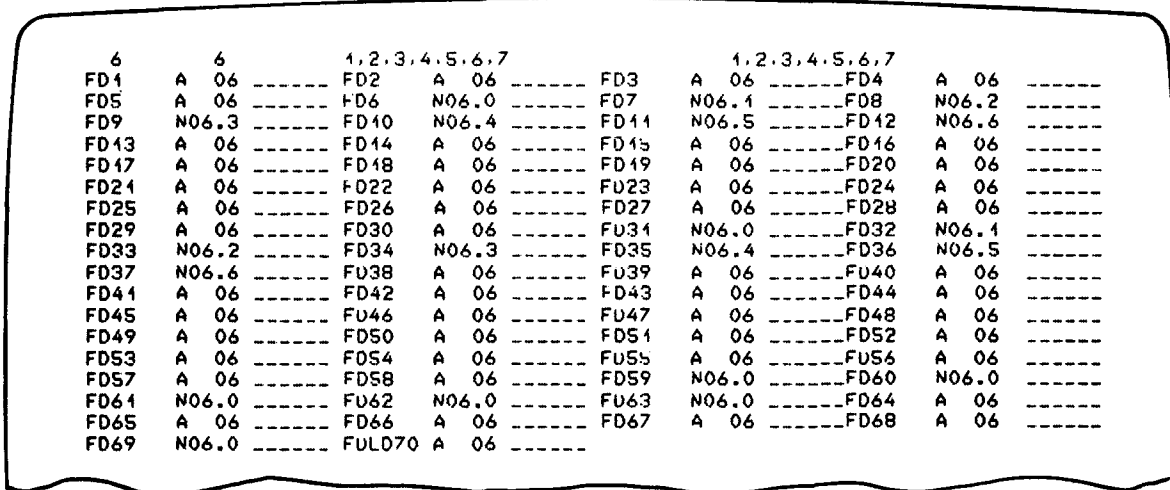


Figure 13-44. Four-Column Prompt Screen (70 Fields)

The workstation prompts are 14 positions in length and have the following format:

<u>Position</u>	<u>Entry</u>
1	Blank (This indicates the workstation field control character for the prompt.)
2 through 7	Field name
8	Blank
9	N indicates a numeric field and A indicates an alphanumeric field.
10 through 13	Field length. With numeric fields, positions 10 and 11 contain the field length, position 12 contains a decimal point, and position 13 indicates the number of decimal positions in the field. With alphanumeric fields, positions 10 and 11 contain blanks and positions 12 and 13 contain the field length.
14	Blank (This indicates the workstation control character for the input field.)

The following job control stream allows you to use interactive data entry:

```
// JOB RPGIDA
// DVC 20
// LFD PRNTR
// WORK1
// WORK2
// TEMP1
// EXEC RPGII
// PARAM CONSOLE=lfname
/$
.
. } source deck
.
/*
/&
// FIN
```

When you compile programs that use CONSOLE files, use the CONSOLE=lfname parameter on the // PARAM job control statement (18.2.3) to differentiate this type of file from a system console file. The RPG II compiler generates batch screen format (S and D) statements that are processed separately by the batch screen format (S and D) converter. When using the CONSOLE parameter in the jproc, the source deck input must be from disk.

The following conditions cause the RPG II compiler to generate error diagnostic messages:

- You use more than one CONSOLE file in a program.
- You use a workstation file and a CONSOLE file in the same program.
- You didn't enter an I (for input file) in column 15 of the file description specifications form.
- You entered a record length that exceeded 1518 characters.
- You specified an invalid record identifying indicator.
- You specified fields that overlap.

↓

### 13.16.1. File Description Specifications for Interactive Data Entry

The following entries are valid for the file description specifications for interactive data entry. The columns not described contain standard entries.

<u>Column</u>	<u>Entry</u>
15	Enter I to indicate that this is an input file.
16	Enter a P for primary file, S for secondary file, D for demand file, or R for record address file.
17	Leave this column blank if the program can end, even if all records from the file are not processed. If you leave this column blank for all files in the program, all records from every file are processed before the program ends. The operator indicates a normal end-of-file by pressing function key 15 for a CONSOLE file.  Enter E if the program can end only if all records from the file are processed.
20 through 23	Leave this column blank or enter a block length that is equal to the record length you enter in columns 24 through 27.
24 through 27	The record length is the highest to-location you specified on the input format specifications form in columns 48 through 51. You can enter a record length from 2 through 1518.  If you indicate the CONSOLE file is a record address file, you obtain the record length by multiplying the length of the record address field by 2. In this case, you can enter a record length from 2 through 58.
28	Must be blank.
29 and 30	Must be blank if you entered a P, S, or D in column 16.  Enter the length of the record key (from 1 through 29) if you entered an R in column 16.
31	This column is only used with record address files. Leave this column blank if the record address file keys are the same as the index file keys.  Enter A if the indexed file has zoned decimal keys.

↑

<u>Column</u>	<u>Entry</u>
32 through 38	Must be blank.
39	Must be blank if you entered a P, S, or D in column 16. Enter E if you entered an R in column 16.
40 through 46	Enter CONSOLE for the device name. The file can either be an input data file or a record address file. You can use only one CONSOLE file in a program.

### 13.16.2. Input Format Specifications for Interactive Data Entry

The following entries are valid for the input format specifications for interactive data entry. The columns not described contain standard entries.

<u>Column</u>	<u>Entry on file and record identification</u>
14 through 16	You must not enter AND but you can enter OR in columns 14 and 15. OR indicates a relationship between record identifying indicators or record types. When you specify OR, you must describe the same number of record identification codes as you described on the preceding line.
15 through 18	RPG II uses the entries you specify in columns 15 and 16 (sequence), column 17 (number), and column 18 (option) to determine: <ul style="list-style-type: none"> <li>■ Which record types can be used in place of the default record types. That is, the record types that can be selected before you enter data for the record type currently being displayed.</li> <li>■ Which record types are valid after you enter data for the record type currently being displayed.</li> </ul> <p>RPG II inserts the valid before-and-after record types in the top line of the screen format that is displayed for a CONSOLE file.</p>
19 and 20	Enter a record identifying indicator from 01 to 10 to define the key the operator must use to select that record type. You can't use the same indicator to define more than one record type in the input format specifications for one program.

<u>Column</u>	<u>Entry on file and record identification</u>
21 through 23	Must be blank.
24	Enter 1 for the record identification code to identify which record was keyed. This code is automatically inserted into each new record when it is prompted.
25	Must be blank.
26	Enter C.
27	Enter the digit, alphabetic character, or special character that you specified in position 1 of the input record.
28 through 34	If you want to specify only one record identification code in columns 21 through 27, leave these columns blank. If you want to specify two record identification codes, then you must also specify entries in these columns. Code these columns as you did columns 21 through 27, except enter 2 in column 31 to indicate record position 2.
35 through 74	Must be blank.

<u>Column</u>	<u>Entry on field description</u>
53 through 58	<p>Enter a field name that will be used as a prompt for this data. This name must be from 1 to 6 alphanumeric characters. If you want to enter data into a whole array for a CONSOLE file:</p> <ul style="list-style-type: none"> <li>■ define the whole array as a subfield within a field of the CONSOLE file record; or</li> <li>■ define each array element with an index and place this entry in columns 53 through 58. The index must be an integer value.</li> </ul>
59 and 60	Enter a control level indicator (L1 through L9) if this is a primary or secondary file to indicate that you want a control break when there is a change in the field's contents.
61 and 62	<p>Enter a matching field indicator (M1 through M9) if this is a primary or secondary file to indicate a match field.</p> <p>Leave these columns blank if this is not a primary or secondary file.</p>





When you describe the fields in a CONSOLE file record:

- An alphanumeric field must not be more than 66 characters in length.
- A numeric field must not be more than 15 digits in length.
- A record must not be more than 1518 characters in length.
- You can specify subfields within the fields of a CONSOLE file record. Make sure the from and to field locations (columns 44 through 51) for subfields don't overlap the from and to field locations for another field. You can't prompt subfields, but they can be assigned values from the prompted field and be used in calculations and output specifications.





## 14. Printing Techniques

### 14.1. GENERAL

If any of the output from your program is to be a printed report, RPG II reduces the effort required to produce the report by providing the following aids:

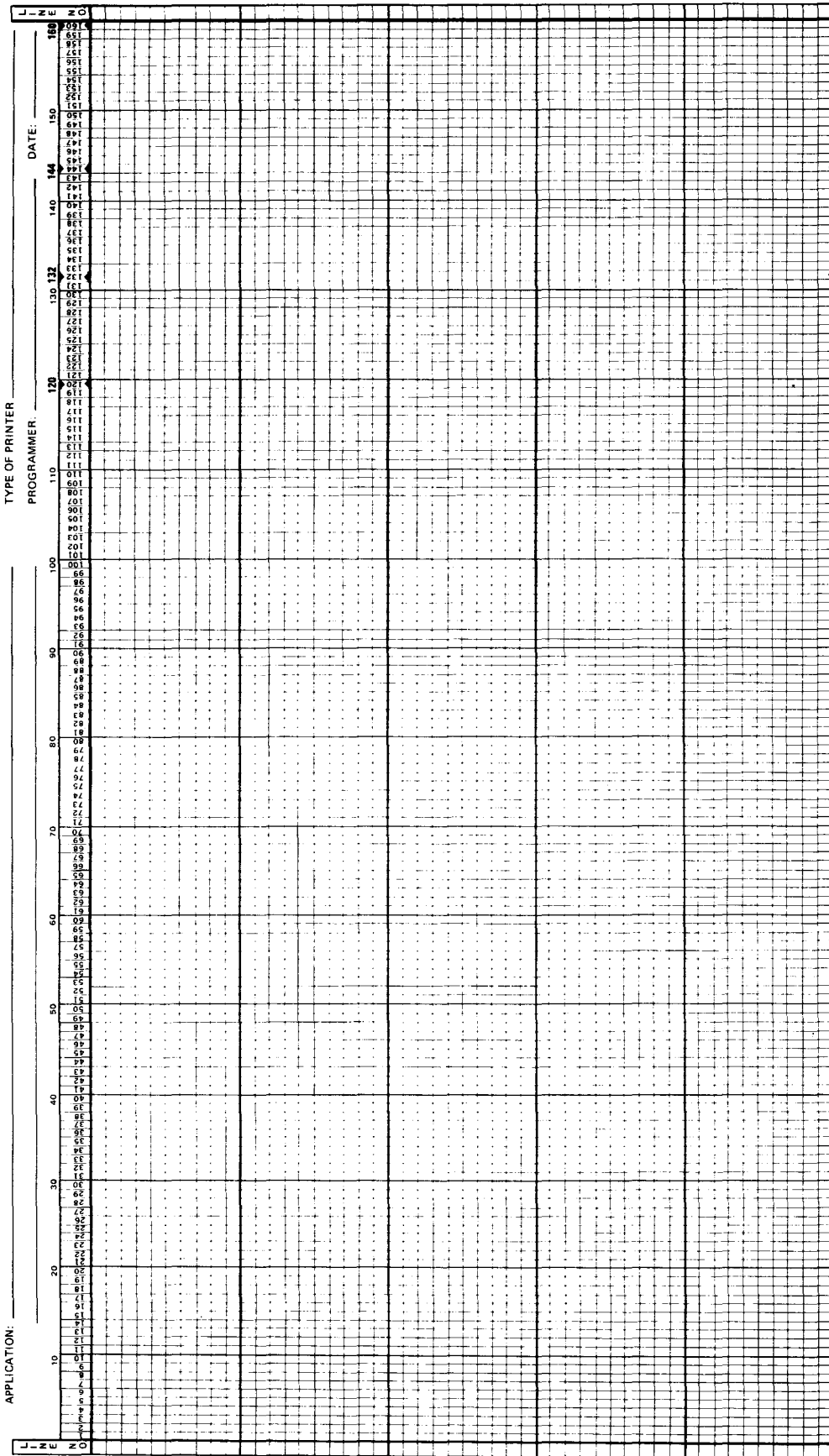
- a preprinted printer format chart that you can lay out your report on;
- an extensive editing capability that you can use in your program to edit the lines to be printed on the report;
- special field names that you can use in your program to cause automatic dating, page numbering, and duplication of fields; and
- the ability to store a report on a line counter file (an intermediate tape or disk file) so that it can be printed when your system is not being used to its fullest capacity.

### 14.2. PRINTER FORMAT CHART

Before you write your program, you must determine what your final product is going to be. If it is a printed report, you must know what is to be printed on each line of the report, that is, heading, detail, or total information. You must also know which fields contain this information, the position on the line where the individual fields are to be placed, and the spacing between each line. This is accomplished by laying out your report on the printer format chart (Figure 14-1). Once the report is laid out, you then use it as a guide to make the necessary entries on the specifications forms to effect the required output.

PRINTER FORMAT CHART  
160 PRINT POSITIONS

UNIVAC



TYPE OF PRINTER  
PROGRAMMER

APPLICATION:

DATE:

UD1:1505

Figure 14-1. Printer Format Chart

The numbers that appear on the top of the form represent the print positions. When you lay out your report, headings and constant information should be spelled out completely in the print positions they are to occupy on the report. Any variable information should be represented by x's. If a field is to be edited, the actual edit word should be entered in the print positions that the field is to occupy.

The use of the columns to the left of the print positions is optional. These columns can be used to specify the line name, the type of line (H - heading information, D - detail information, T - total information), the number of lines to be spaced before or after the line is printed, the number of the channel on the carriage control tape that is to be skipped to before or after the line is printed, whether the information on the line is alphanumeric or numeric (A or N), and the line number (the position of the line relative to the top of the form).

The area to the right of the print positions is used to associate the channel number on the carriage control tape with the line on the form that you want to skip to before or after printing the line. The channel number is indicated by placing an x in one or more of the channel columns in this area such that the sum of the indicated values of the channel columns equals the number of the channel you want to specify. For example, if after printing a line on line 12 you wanted to skip to line 40, you could accomplish this by specifying channel 6 in columns 21 and 22 on the output format specifications form for this line and then indicating channel 6 on the printer format chart carriage control tape area opposite line 40. Note that the channel numbers that you may specify are determined by the printer subsystem that is being used.

Figure 14-2 shows an example of typical entries on the printer format chart.

In this example, a report listing the accounts receivable is laid out. Each page of the report will have headings printed and then the detail information for each customer follows. When all the information for one customer is exhausted, a control break occurs and a subtotal is printed. The next customer is listed and the process continues until all data is exhausted, at which point a grand total of all accounts receivable is printed. Page overflow is to occur at line 40. The heading lines, detail lines, and total lines are indicated in the columns to the left of the print positions by H, D, and T, respectively. The spacing between the lines is also indicated in these columns. The dollar amounts in print positions 101 through 110 are to have a dollar sign (\$) at the extreme left, commas are to be inserted where necessary, they are to be zero suppressed up to the decimal point, and a single asterisk (\*) indicates a subtotal and double asterisks (\*\*) indicate the grand total. This is indicated by the edit words that appear in these print positions. The top of the form (home paper) is indicated by specifying channel 7 on the carriage control tape opposite line 4 and page overflow is indicated by specifying channel 1 opposite line 40; that is, when the printer carriage reaches line 40 it will automatically skip to the top of the next page of the report.

### 14.3. EDITING

With RPG II the editing of numeric fields (including the printing of dollar signs, commas, decimal points or periods, and signs) can be accomplished by using either edit codes or edit words.



PRINTER FORMAT CHART  
180 PRINT POSITIONS

APPLICATION: \_\_\_\_\_ TYPE OF PRINTER: \_\_\_\_\_ DATE: \_\_\_\_\_

PROGRAMMER: \_\_\_\_\_

10	20	30	40	50	60	70	80	90	100	110	120	130	140	150	160	170	180
PURCHASER NUMBER 12345	CUSTOMER NAME UNIVERSITY BOOKS	LOCATION STATE CITY KY N.A.	INVOICE NUMBER KXKX	INVOICE DATE MO DAY 01 01	INVOICE AMOUNT 100.00												

UD-1-1965

Figure 14-2. Example of Entries on Printer Format Chart

### 14.3.1. Edit Codes

A series of 1-character edit codes is provided that edit a field in a predefined way. These edit codes consist of the simple edit codes (X, Y, and Z) and the combined edit codes (1, 2, 3, 4, A, B, C, D, J, K, L, and M).

An edit code is used by entering it in column 38 of the output format specifications form on the line that contains the field to be edited. Note that the end position specified for the output must allow for the insertion of data into the field by the edit code.

If an entire array is to be printed using an edit code, two blanks are inserted to the right of every element in the array before any other editing takes place. In this case, the end position specified for the output must allow for the added blanks.

#### 14.3.1.1. Simple Edit Codes

The editing performed by the simple edit codes is as follows:

- X

Removes the plus from the rightmost character position in the field. A negative sign is not removed and leading zeros are not suppressed. If S or blank is specified in column 40 of the control card specifications form, the plus sign is removed automatically without the use of this edit code.

- Y

Inserts slashes in 3- to 6-character fields as follows:

<u>Field Length</u>	<u>Edited Result</u>
3	xx/x
4	xx/xx
5	xx/xx/x
6	xx/xx/xx

- Z

Suppresses zeros to the left of the most significant digit of the field and removes the sign from the rightmost character position.

Decimal points are not considered and are not printed when this edit code is used.

**14.3.1.2. Combined Edit Codes**

The editing performed by the combined edit codes is shown in Table 14-1.

All of the combined edit codes shown in Table 14-1 suppress leading zeros. The number of decimal positions specified for a field determine where the decimal point is printed. In these cases the last digit to be zero suppressed is the one immediately to the left of the decimal point. If decimal positions are specified for a field and a zero balance is not to be printed, the decimal point is printed if the field contains a nonzero balance. If decimal positions are specified and a zero balance is to be printed, the decimal point is printed and the zero balance is printed to the right of the decimal point. Table 14-2 shows all of the combined edit codes and provides examples of their effect on unedited fields.

*Table 14-1. Combined Edit Codes*

Combined Edit Codes			Editing Performed
Negative Value Indication			
None	CR	--	
1	A	J	Commas are inserted in the field; a zero balance is printed.
2	B	K	Commas are inserted in the field; a zero balance is not printed.
3	C	L	Commas are not inserted in the field; a zero balance is printed.
4	D	M	Commas are not inserted in the field; a zero balance is not printed.

If J is specified in column 21 of the control card specifications form, the commas replace decimal points and vice versa in the edited fields and the last digit to be zero suppressed is the second one to the left of the decimal point.

Asterisk protection or a floating dollar sign can be used with any of the combined edit codes. This is accomplished by specifying '\*' or '\$' in columns 45 through 47 along with a combined edit code in column 38 of the output format specifications form on the line that contains the field to be edited. The dollar sign will appear to the left of the most significant nonzero digit. An asterisk will appear in place of each zero that is suppressed.

**14.3.2. Edit Words**

Generally, the edit codes will be sufficient to handle your editing requirements. Occasionally a situation will arise which requires special or unusual editing. For such situations RPG II allows you to specify your own editing requirements. This is accomplished by specifying an edit word enclosed in apostrophes in columns 45 through 70 of the output format specifications form on the line that contains the field to be edited. If an edit word is used, column 38 on that line must be blank.



Table 14—2. Using Combined Edit Codes

Specified Conditions	Unedited Value in Field	Edit Code Specified											
		1	2	3	4	A	B	C	D	J	K	L	M
Zero balance with no decimal positions	000000	0	blank	0	blank	0	blank	0	blank	0	blank	0	blank
Zero balance with 2 decimal positions	000000	00	blank	.00	blank	00	blank	.00	blank	.00	blank	.00	blank
Negative number with no decimal positions	000890	890	890	890	890	890CR	890CR	890CR	890CR	890-	890	890-	890-
Negative number with 3 decimal positions	000890	.890	.890	.890	.890	.890CR	.890CR	.890CR	.890CR	.890	.890	.890-	.890
Positive number with no decimal positions	5432154	5,432,154	5,432,154	5432154	5432154	5,432,154	5,432,154	5432154	5432154	5,432,154	5,432,154	5432154	5432154
Positive number with 2 decimal positions	5432154	54,321.54	54,321.54	54321.54	54321.54	54,321.54	54,321.54	54321.54	54321.54	54,321.54	54,321.54	54321.54	54321.54

### 14.3.2.1. Edit Word Format

An edit word consists of three parts: the body, sign status, and expansion. The body is the portion of the edit word that provides space for the digits that are to be transferred from the field to be edited and the punctuation that is to appear in the edited field. The body begins at the leftmost position of the edit word and ends at the rightmost position that is to contain a digit from the field to be edited. The sign status is the portion of the edit word that is used to specify whether the field is positive or negative and/or to specify constant information. The sign status begins at the first position to the right of the body. CR, -, or & is used to indicate sign status. If any of these is specified and a field is positive, blank spaces will be substituted in the edited field. If CR or - is specified and a field is negative, the specified indicator will be printed. The expansion consists of characters that will be printed regardless of sign status. Figure 14-3 shows examples of the format of typical edit words.

### 14.3.2.2. Using Edit Words

The following rules apply to edit words:

1. An edit word must be enclosed in apostrophes.
2. A blank in the body of the edit word indicates that this position in the edited field is to contain the character from the same position in the data field.
3. A zero in the body of the edit word indicates that the field is to be zero suppressed. It should be placed in the rightmost position where zero suppression is to stop. Each zero that appears to the left of the stop position in the data is replaced with a blank space in the edited field. Zero suppression begins at the leftmost position in the data and continues up through the stop position unless a nonzero digit is encountered to the left of the stop position. In the latter case, zero suppression stops at the position where the nonzero digit is encountered and that digit and all following are printed.
4. An asterisk (\*) in the body of the edit word indicates that the field is to be edited using asterisk protection. It should be placed in the rightmost position where zero suppression is to stop. Each zero that appears to the left of the stop position in the data is replaced by an asterisk.
5. An ampersand (&) in either the body or the status portion of the edit word indicates that that position in the edited field is to be a blank space.
6. A dollar sign (\$) in the body of the edit word immediately to the left of the zero suppression code (0) indicates that the dollar sign is to appear immediately to the left of the first significant digit in the edited field. This is called the floating dollar sign. A dollar sign in the leftmost position of the edit word indicates that the dollar sign is to appear in that position in the edited field. This is called the fixed dollar sign.

Note that a dollar sign is not replaced by a digit in the field to be edited and cannot be counted as a digit select character. Note also that when you use a floating dollar sign, the number of blanks plus the zero in the edit word must equal the number of digits in the field to be edited.



7. Decimal points and commas in the body of the edit word indicate the position in the edited field where they are to be printed. They will be printed in the indicated positions unless they are to the left of the most significant digit. In that case, they will be replaced by blank spaces or asterisks if asterisk protection is used.
8. CR or a minus (-) in the status portion of the edit word indicates that the specified symbol is to be printed in the edited field if the data is negative. If the data is positive the symbol is not printed.
9. Any character, other than \$, &, -, \*, or comma, that appears in the body of the edit word is printed in the edited field provided that its relative position is to the right of the most significant digit in the data. If it is to the left, a blank space replaces it.
10. Any character other than an ampersand (&) that is included in the status portion of the edit word will be printed in the edited field. If an ampersand is specified, it will be replaced by a blank space.
11. If leading zeros are desired, the edit word should be increased by one position on the left and a zero should be placed in that position.

Figure 14-4 shows examples of typical edit words and the results that will be printed.

#### 14.4. SPECIAL FIELD NAMES

Automatic dating, page numbering, and duplication of fields on your report can be accomplished by specifying certain special RPG II field names in columns 32 through 37 of the output format specifications form. These special field names are D?TE, UDATE, UDAY, UMONTH, UYEAR, PAGE, PAGE1-7, and \*PLACE.

##### 14.4.1. D?TE, UDATE, UDAY, UMONTH, and UYEAR

These special field names allow you to have a date automatically printed on your report in the format you choose. The date that is printed is the system date unless you have specified a different date at execution time with a //△SET△DATE job control statement.

If D?TE or UDATE is specified, the date in the format mmddy (mm = 2-digit month, dd = 2-digit day, and yy = 2-digit year) will be printed. If D, I, or J is specified in column 21 of the control card specifications form, the format will be ddmmy when this field is specified.

If UDAY is specified, only the 2-digit system date day is retrieved.

If UMONTH is specified, only the 2-digit system date month is retrieved.

If UYEAR is specified, only the 2-digit system date year is retrieved.

When UDATE is specified, the Y edit code can be used in column 38 of the output format specifications form to edit the format so that it appears as mm/dd/yy or dd.mm.yy if D, I, or J is specified in column 21 of the control card specifications form.



#### 14.4.2. PAGE, PAGE1-7

These special field names allow you to have the pages of your report automatically numbered starting with 1. Eight field names are provided so that if your program is to produce more than one report, each of the reports can have automatic page numbering by specifying a different field name on the output format specifications form for each report. For example, if three reports are to be produced by one program, automatic page numbering could be specified for each report by using special field name PAGE for the first report, PAGE 1 for the second, and PAGE 3 for the third.

A page field is a 4-digit numeric field with no decimal positions. At the beginning of the program the specified page field is set to zero and then each time before it is written the value of the field is increased by 1.

If page numbering is not to commence with 1, the page field that is being used for your report can be set to commence numbering with the page number you require by specifying the page field in columns 53 through 58 on the input format specifications form or by specifying it as a result field in columns 43 through 48 on the calculation specifications form. In the former case, the page field is defined as an input record field that contains a value and, in the latter, as a field that has had a value placed in it as a result of an operation performed on the calculation specifications form. Note that if either of these methods is used, the value in the field must be one less than the required starting page number because the field value is increased by 1 before it is written.

To reset a page field:

- Specify the field as a result field with no decimal positions (0 in column 52) on the calculations specifications form and use one of the available operations to place zeros in the field.
- Specify B (blank after) in column 39 of the output format specifications form on the line where the page field is specified.
- Specify an indicator (columns 24-25, 27-28, 30-31) for PAGE on the output format specifications form. If the indicator is on, PAGE is set to zero and 1 is added before it is written. If the indicator is off, PAGE is not reset.

The page field is automatically zero suppressed when it is written unless you specified an edit code other than X or an edit word. In this case, the edit code or edit word overrides the zero suppression.

↓  
To define a page field:

- Specify the field as a field name in columns 53 through 58 on the input format specifications.
  - Specify the field as a result field with no decimal positions (0 in column 52) on the calculation specifications form.
  - Referencing PAGE on the output format specifications form will define the field to be a 4-digit numeric field with no decimal positions.
- ↑

### 14.4.3. \*PLACE

This special field name allows you to duplicate (repeat) fields or constants in additional positions in an output record without having to repeat the field name or constant on the output format specifications form.

When \*PLACE is specified in columns 32 through 37 on the output format specifications form, all previously specified data between position 1 and the highest end position for this data is repeated in the output record ending in the end position specified in columns 40 through 43 on the \*PLACE line. The end position specified for the \*PLACE line must be no greater than 256. If this is not done, overlapping of data will occur. Figure 14-5 shows an example of how \*PLACE is used.





### OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	STACKER SELECT/ F-FETCH OVERFLOW	SPACE				SKIP				OUTPUT INDICATORS						FIELD NAME	DATA FORMAT P/B/L/R	CODES			CONSTANT OR EDIT								
				TYPE H/D/T/E	BEFORE	AFTER	BEFORE	AFTER	AND	AND	N-NOT	N-NOT	N-NOT	NONE	CR	J	COMMAS INSERTED														
1	2	3	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45
	01	0	OUTPUT	D											02																
	02	0																									8				'SUBTOTAL'
	03	0																									20				'GRAND TOTAL'
	04	0																													*PLACE
	05	0																													*PLACE

Figure 14-5. Example of Using \*PLACE

The example in Figure 14-5 shows that two constants, SUBTOTAL and GRAND TOTAL, are to be written ending in print position 20. The \*PLACE instructions on line 040 and 050 will cause the constants to be repeated ending in print positions 42 and 64. Figure 14-6 shows how the constants appear in the output record after each line on the output format specifications form is executed. The data added by the execution of each line appears outside the shaded area.

Line	1	10	20	30	40	50	60
020	SUBTOTAL						
030	SUBTOTAL GRAND TOTAL						
040	SUBTOTAL GRAND TOTAL SUBTOTAL GRAND TOTAL						
050	SUBTOTAL GRAND TOTAL SUBTOTAL GRAND TOTAL SUBTOTAL GRAND TOTAL						

Figure 14-6. Example of Using \*PLACE to Repeat Constants

## 14.5. LINE COUNTER FILES

Line counter files are intermediate tape or disk output files that you can store a report on for future offline printing. This type of file is generally used when printing facilities are not immediately available.

### 14.5.1. Using Line Counter Files

To use a line counter file in your program you must specify that your report is to be placed on a line counter file by placing an L in column 39 of the file description specifications on the line where you describe your report file. Figure 14-7 shows an example of how this is specified on the file description specifications form. In the example, a report is to be placed on a line counter file that is to be stored on disk.

## FILE DESCRIPTION SPECIFICATIONS

PAGE NO.	FORM TYPE <b>F</b>		FILE NAME	FILE TYPE					FILE PROCESSING MODE					EXTENSION OR LINE COUNTER CODE		NOT USED	S/N/E														
	LINE NO.	LINE NO.		FILE DESIGNATION	END OF FILE	SEQUENCE	FILE FORMAT	KEY OR RECORD ADDRESS FIELD LENGTH	RECORD ADDRESS TYPE	FILE ORGANIZATION	OVERFLOW INDICATOR	KEY FIELD STARTING LOCATION	DEVICE	LINE COUNTER CODE																	
1	2	3	5	6	7	13	14	15	16	17	18	19	20	23	24	27	28	29	30	31	32	33	34	35	38	39	40	46	47	52	53
	0	1	0	F	OFFLINE																										
	0	2		F																											

Figure 14-7. Specifying a Line Counter File

You must also specify which lines of the printed page of the report are associated with a channel number that is punched on the carriage control tape or specified through a // VFB job control statement. This information is taken from the printer format chart for your report. The information on this form is used to cause printer control characters to be placed in the output records on the intermediate tape or disk file that the report is to be stored on. When the report is subsequently printed, these characters will cause the printer to exercise carriage control in the same manner as if the report were printed during the execution of your program. Figure 14-8 shows an example of how this information is specified on the line counter specifications form. In this example, line 09 is associated with channel 07 (home paper), line 35 with channel 06, and line 50 with channel 01 (forms overflow).

## LINE COUNTER SPECIFICATIONS

PAGE NO.	FORM TYPE <b>L</b>		FILE NAME	1		2		3		4		5		6		LIN NO															
	LINE NO.	LINE NO.		LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.	LINE NO.	CH NO.																
1	2	3	5	6	7	14	15	17	18	19	20	22	23	24	25	27	28	29	30	32	33	34	35	37	38	39	40	42	43	44	45
	0	1	0	L	OFFLINE			9	0	7	13	5	0	6	5	0	0	1													

Figure 14-8. Specifying Lines Associated with Channel Numbers for a Line Counter File

Note that the lines associated with the home paper and forms overflow channel must always be specified on the line counter specifications form.

### 14.5.2. Printing Line Counter Files

You can print line counter files by using the file processing utility as described in the data utilities user guide.

## 15. Incorporating Subroutines in Your Program

### 15.1. GENERAL

If the same series of calculation operations is to be performed at more than one point in your program, this series should be included in a subroutine. By doing this, you avoid having to specify these operations at each point where they are required. Instead, you specify them once in a subroutine, and each time they are required, you branch to the subroutine, execute the series, and return to your main program.

The subroutine can be either an internal subroutine or an external subroutine. An internal subroutine is one that you include within your program, and an external subroutine is one that you code independent of your program in another language or one coded by RPG II.

The internal subroutine is the type that is generally used. There are occasions, however, when the series of operations that you need in your subroutine cannot be easily specified in RPG II language. It is for these occasions that RPG II provides you with ability to use external subroutines.

### 15.2. INTERNAL SUBROUTINES

Internal subroutines are written in RPG II language on the calculation specifications form. The subroutines must be placed on the form after all other main program calculation operations, and each line of a subroutine must have SR specified in columns 7 and 8. If more than one subroutine is written in one program, a unique name must be specified for each one. There are three operations that are used to indicate the beginning, end, and when the subroutine is to be executed. The BEGSR operation is always the first line in the subroutine. It is used to indicate the beginning of the subroutine and the name that is being used for the subroutine. The ENDSR operation is always the last line of the subroutine. It is used to indicate the end of the subroutine and branch back to the main program. The EXSR operation is used within the main program to cause the subroutine to be executed at this point; that is, this operation causes the program to branch to the specified subroutine and execute it. When the subroutine is completed (the ENDSR operation), the program branches back and executes the line immediately following the EXSR operation that caused the subroutine to be executed.

Indicators can be used with the EXSR operation to specify the conditions that must be met before the subroutine is executed. These indicators are specified in columns 7 through 17. If a control level indicator is not specified in columns 7 and 8, the subroutine is always executed during detail time.

Any operation that is used within a subroutine, except the BEGSR operation (first line) and the ENDSR operation (last line), can be conditioned by indicators specified in columns 9 through 17. Control level indicators cannot be specified in columns 7 and 8 because SR is required in these columns for each line in a subroutine.

GOTO (branching) operations are permitted within subroutines. A GOTO operation within a subroutine can specify the name of a TAG operation or ENDSR operation. It cannot specify the name of a BEGSR operation. A GOTO operation within a subroutine can also specify the name of a TAG operation within the main program. However, the reverse is not permitted; that is, a GOTO operation in the main program cannot specify the name of TAG or ENDSR operation in a subroutine. Internal subroutines may be specified in any order on the calculation specifications form. Each subroutine must be a complete entity; that is, a subroutine cannot define another within itself because only one BEGSR and ENDSR operation is permitted in a subroutine. The only relationship that one subroutine can have to another is for one to call for the execution of the other by specifying an EXSR operation.

### 15.2.1. Specifying an Internal Subroutine

Figure 15-1 shows you how you can specify internal subroutines on the calculation specifications form. In this example, there are two internal subroutines: SUBRT and SUBRU. SUBRT is executed at detail time and SUBRU is executed at total time.

Line 010 shows that the subroutine named SUBRT is to be executed at detail time (columns 7 and 8 are blank) if indicator 03 is on and indicator 04 is off. After subroutine SUBRT is executed, the operation on line 020 is executed.

Line 020 is an additional detail time calculation operation.

Line 030 shows that the subroutine named SUBRU will be executed during total time calculations if the L3 indicator is on. After subroutine SUBRU is executed, the operation on line 040 is executed.

Line 040 is a TAG operation with the label INTER.

Line 050 shows that the subroutine named SUBRT will be executed during total time calculations if indicator L4 is on. After subroutine SUBRT is executed, the operation on line 060 is executed.

Lines 060 and 070 are additional L4 total time calculation operations.

Line 080 shows that this BEGSR operation indicates the beginning of the subroutine named SUBRT.

Line 090 is an additional calculation operation within subroutine SUBRT.

Line 100 shows that this ENDSR operation indicates the end of the subroutine named SUBRT.

Line 110 shows that this BEGSR operation indicates the beginning of the subroutine named SUBRU.



Line 170 shows that if indicator 03 is on during the execution of subroutine SUBRU, the program branches to subroutine SUBRT and executes it. When the execution of SUBRT is completed, the program returns to line 180 and the execution of subroutine SUBRU continues.

Line 180 is a TAG operation within the subroutine SUBRU and it is labeled MIDU.

Line 190 shows that when indicators 04 and 13 are on and indicator 03 is off, the GOTO operation is executed and the program branches to the line in the main program labeled INTER (line 040).

Line 200 shows that this ENDSR operation indicates the end of the subroutine named SUBRU.

### 15.3. EXTERNAL SUBROUTINES

The writing of an external subroutine (usually in OS/3 basic assembly language) and its subsequent assembly or compilation is separate from the RPG II program that is to use it. Simply stated, the writing and assembling or compiling of the external subroutine is a separate effort and the writing and compiling of the RPG II program (your program) is a separate effort. The results of these efforts, the RPG II object program and the external subroutine object program, are then combined at linkage editor time to form an executable program that can use the external subroutine.

There are three operations that are used on the calculation specifications form with external subroutines: EXIT, RLABL, and ULABL.

The EXIT operation is used within the main program to cause the external subroutine to be executed at this point; that is, this operation causes the program to branch to the specified external subroutine and execute it. When the subroutine is completed, the program branches back and executes the next sequential operation (other than RLABL or ULABL) in the main program. (Your program creates linkage to the external subroutine by loading register 15 with the address of the subroutine and by issuing a BALR 14, 15. The external subroutine must restore all registers before returning control to your program.) Indicators may be used with the EXIT operation to specify the conditions that must be met before the external subroutine is executed. These indicators are specified in columns 7 through 17. If a control indicator is not specified in columns 7 and 8, the external subroutine is always executed during detail time.

The RLABL operation allows the external subroutine to reference a field, table, array, or indicator in your program. The name of the field, table, or array is specified in columns 43 through 48. If an indicator is referenced, the name that is specified must be in the form INii, where ii is the 2-character indicator code. The RLABL operation need only be used once in your program for each field, table, array, or indicator that is to be referenced by an external subroutine. This applies even when more than one external subroutine is being used by your program.

→ To link your program to a FORTRAN or COBOL subroutine, every EXIT to the external subroutine must be followed immediately by the RLABL fields.



Line 050 shows an RLABL operation that allows the external subroutine to reference the field ACCNO in the RPG II program.

Line 060 shows an RLABL operation that allows the external subroutine to reference the table named TABL.

Line 070 shows an RLABL operation that allows the external subroutine to reference the array named ARRAY.

Line 080 shows an RLABL operation that allows the external subroutine to reference indicator 22 and test its status.

Lines 090 and 100 are detail time calculation operations.

#### 15.4. RPG II SUBROUTINES

You can specify that your RPG II program is to run as a subroutine by coding an S in column 74 of the control card specifications form. When your RPG II subroutine is called, it operates as a main program. The files are opened, the program executes until the last record indicator (LR) is set on or an error condition is encountered, files are closed, and control is returned to the calling program.

The calling program must specify the following when it calls an RPG II subroutine:

- R15 contains the address of the RPG II subroutine.
- R14 contains the return address.
- R13 contains the address of an 18-word register save area.

When control is returned to the calling program, R15 contains 0 if program execution was normal. If a halt indicator was set during program execution, R15 contains 255. If the RPG II subroutine is called again, indicators are reset to initial values; however, fields and tables are not reset.



## 16. IMS - RPG II Action Programs

### 16.1. GENERAL

This section describes, in general terms, how to write IMS action programs in the OS/3 RPG II language. The user should be familiar with the current versions of the IMS action programming in RPG II user guide, UP-9206, and the IMS system support functions user guide, UP-11907, and become familiar with the various RPG II specifications forms described in Section 6 through 8 of this manual. ←

IMS action programs written in RPG II operate under control of IMS to carry out the processing of input messages and to generate output messages. The action programs are compiled by the RPG II compiler. A special IMS configuration program generates the IMS system to which the action program can be linked. Action programs are scheduled and then receive control as a result of an input message request from a terminal or a request from a previously executed action program.

IMS action programs written in RPG II are not reentrant, and, therefore, can process only one transaction at a time. They should be written so that they are serially reusable. After each execution of an action program, RPG II resets all indicators and internal switches. Therefore, you must reset all fields to their original values, as required, before the action program is executed again. You should not assume that fields are blank or zero. During a transaction, if a program calls itself as the successor, you can save some or all of the fields for the next execution of this program. At the end of one execution, the fields can be output to the continuity data area or to a transaction buffer, and then read in from there at the beginning of the next execution. Other reasons for saving fields between executions of the action program are: ←

- IMS may reload the action program.
- Two or more terminals may be processing the same kind of transaction. This causes IMS to switch between the terminals for successive executions of the action program.

Remember that RPG II resets all indicators after each execution of an action program.

## 16.2. RPG II SPECIFICATIONS FORMS

IMS provides defined areas for the communications and control of data between the action program and IMS. These areas plus any user files must be defined and described on the proper RPG II specifications forms. These forms are, in order of their specification:

- Control Card Specifications Form

This form requires an A in column 74 and the program name in columns 75 through 80. Each action program must be identified by a unique name. No other entries are required on this form. The rules for coding any other fields on this form are given in Section 4.

- File Description Specifications Form

The IMS-defined areas and users logical files must be described on this form. The rules for coding the various fields on this form and the sequence of specification follow those for any file written in the OS/3 RPG II language. Refer to Section 5 for details.

- Input Format Specifications Form

The fields within each input file are described on this form. These fields must be described in the same order as the files. Refer to Section 6 for details.

- Calculation Specifications Form

The action of your action program is coded on this form. It should follow the rules specified in Section 7.

- Output Format Specifications Form

The fields within each output file are described on this form. Refer to Section 8 for details.

An IMS action program written in RPG II requires no other RPG II coding forms.

## 16.3. RESTRICTIONS

A number of the RPG II language features are restricted from use in an action program. Table 16-1 lists these features and their restrictions.

Table 16-1. RPG II Language Features with Restricted Use (Part 1 of 2)

Specifications Form/Column/Description	Restriction
Control Card Specifications:	
Col. 8 – Error Analysis Dump	NA
Col. 9 – Operator Control	NA
Col. 41 – First Page Forms Alignment	NA

Table 16-1. RPG II Language Features with Restricted Use (Part 2 of 2)

Specifications Form/Column/Description	Restriction
File Description Specifications:	
Col. 15 – File Type (C and D)	NA
Col. 16 – Table and Array File Designation (T)	S
Col. 20-23 – Block Length	C
Col. 32 – File Organization:	
ADDROUT (D)	S
Record address (blank)	S
Additional I/O areas	NA
SAM tape/disk input files	S
ISAM and indexed IRAM output files	NA
Col. 40-46 – Device:	
CTLRDR	}
READER	
CRP	
PUNCH	
CONSOLE	
PRINTER	NA
Col. 53 – Labels	NA
Col. 54-59 – Name of Label Exit	NA
– Options	NA
Col. 60-65 – Size of ISAM Index	C
Col. 66 – Unordered Load	NA
Col. 67 – Cylinder Overflow Space Percentage	NA
Col. 68-69 – Number of Extents	NA
Col. 70 – Tape Rewind	C
Col. 71-72 – File Conditioners (U1-U8)	NA
Input Form Specifications:	
Col. 19-20 – Spread Card Feature (TR)	NA
Col. 42 – Stacker Select	NA
Calculation Specifications:	
Col. 28-32 – Display Operation (DSPLY)	NA
Output Format Specifications:	
Col. 16 – Stacker Select	NA
Telecommunications Specifications	
	NA

## LEGEND:

- C Ignored by RPG II compiler; must be specified in IMS configuration.
- S Used only with nonindexed sequential IRAM files; must not be used with SAM input files.
- NA Not allowed in an action program.

## 16.4. IMS - RPG II INTERFACE AREAS

The IMS main storage areas that serve as interfaces between IMS and the action programs written in RPG II language are initially defined by IMS and, to be accessed, should also be defined on the appropriate RPG II coding forms. These areas and the required RPG II coding forms are:

- Input Message Area (IMA)

This area contains the input message received from a terminal. The IMA is described as a file on the file description specifications form, and the fields within the file are described on the input format specifications form.

- Program Information Block (PIB)

This area is used to pass control information between IMS and the action program. The PIB is described as a file on the file description specifications form, and the fields within the file are described on either the input format specifications form or the output format specifications form according to their use in the action program.

Any name may be assigned to a field, but the format and position of each field is defined by IMS, and the RPG II specifications must agree with those definitions.

- Output Message Area (OMA)

This area contains the output message to be transmitted to a terminal in response to the input message that originated the action. The OMA is described as a file on the file description specifications form, and the fields within the file are described on the output format specifications form.

This area may also be used for multiple output messages or for transmitting output messages to a terminal other than the initiating terminal. These uses require that disk queueing be included in the IMS configuration.

- Continuity Data Area (CDA)

The use of this area is optional but can be used to pass data from one action program to another while processing a dialog transaction or to save data for the next execution of the action program. The definition of the CDA depends on its use in the action program and can be described as an input, update, or output file. The CDA is described as a file on the file description specifications form, and the fields within the file are described on the input format specifications form or the output format specifications form according to their use in the action program.

- Transaction Buffer Area (TBA)

This area can be used by the action program to acquire and release main storage on a transactional basis. Like the CDA, the TBA can be used in the action program as an input, update, or output file. Using the TBA is faster than using the CDA because data is stored in main storage rather than in a file. To define the TBA as a file, add it to the file description form. Then use the appropriate RPG format specification to describe the fields. To release the transaction buffer from an action program, use the delete option in columns 16 through 18 of the output specifications form.

A summary of the required entries on the file description specifications form for these four areas is given in Table 16-2.

Table 16-2. Summary of Required Entries — File Description Specifications Form

Area	File Type (Column 15)	File Designation (Column 16)	Format (Column 19)	Record Length (Columns 24-27)	Device Name (Columns 40-46)
PIB	I or U	D	F	136	*PIB
IMA	I or U	P, S, or D	F	16 + message size	*IMA
OMA	U or O	D or blank	F	16 + message size	*OMA
CDA	I, U, or O	P, S, D, or blank	F	data size	*CDA
TBA	I, U, or O	P, S, D, or blank	F	data size	*TBA

The record length specified for IMA and OMA must include 16 bytes for the IMS header information that occupies the first 16 bytes of the record. The record length for the IMA must include the 16-byte header information, the transaction code, one blank, and the message text. The record length specified for TBA is restricted to a maximum size of 8192.

Entries on the file description specifications form for the PIB are made according to the way the action program is to use the PIB.

- Input, Demand File

An I in column 15 and a D in column 16 indicates that the action program accesses the data in the PIB but makes no changes to the data.

- Update, Demand File

A U in column 15 and a D in column 16 indicates that the action program accesses the data in the PIB and then updates the data file.

If the OMA is used for multiple output messages during an action program execution, or if the output message is transmitted to a terminal other than the initiating terminal, IMS must be configured for disk queueing.

The normal RPG II rules for the input format specifications form apply to the IMA, PIB, CDA, and TBA definitions. Only those fields used by the action program need to be defined. The same applies to defining the OMA, CDA, and TBA fields on the output format specifications form.

Table 16-3 gives a summary of the file organization, related access methods, and file types that may be used in an IMS action program written in RPG II.

*Table 16-3. Summary of File Organization, Access Methods, and File Types*

File Organization	Related Access Method	File Type
IMS defined		Input/Update
ISAM	Random Sequential, by key	Input/Update Input/Update
IRAM	Indexed Nonindexed	Input/Update/Output Input/Update/Output
SAM	Sequential	Output
DAM	Direct	Input/Update/Output

For further information, details, and requirements for the IMS - RPG II action program interface, refer to the IMS programmer reference, and the OS/3 IMS system support functions user guide/programmer reference.

**PART 4. RPG II TELECOMMUNICATIONS**





## 17. Telecommunications

### 17.1. GENERAL

RPG II telecommunications is not intended for sophisticated applications such as message switching. It is provided so that you can use remote terminals in the same way that you use other input/output devices in a program. There are two types of remote terminals: batch terminals and interactive terminals. The batch terminals can be used in the same way that you use a card reader for an input file, or a card punch or printer for an output file. The interactive terminals have the additional ability to be used in the same way that you use a card reader/punch for combined files.

### 17.2. USING TELECOMMUNICATIONS

Before you can write and compile an RPG II telecommunications program, you must know the name of the communications control area (CCA) that was specified at system generation when the communications network was defined.

This CCA name must be entered in the CCA name field (columns 70 through 73) on the control card specifications form when you write your program to indicate that it is a telecommunications program.

You then specify which files are communications files by entering REMOTE as the device name in columns 40 through 46 on the file description specifications form.

After you have specified which files are communications files, you must describe these files on the telecommunications specifications form. The entries for the various fields are described in Section 11. Keep in mind that the entries in the configuration field (column 15), switched field (column 20), remote terminal (columns 48 through 51), remote device field (columns 65 through 70), and the terminal name field (columns 71 through 74) must match the entries made at system generation when the communications network was defined.

Figure 17-1 shows two examples of how the control card, file description, and telecommunications specifications are used to indicate that a program is a telecommunications program and to specify and describe the communications files. The entries on the control card specifications apply to both examples.

Line 010 on the control card specifications shows that CNET is the CCA name.

The circled numbers on the file description and telecommunications specifications forms refer to the following examples:

■ Example 1

Line 010 on the file description specifications shows that INFILE is an input primary (I in column 15 and P in column 16) and that it is a communications file (REMOTE in columns 40 through 46).

Line 010 on the telecommunications shows that INFILE is a receive file (R in column 16). INFILE is specified as a receive file because your program will receive input data from this file. This line also specifies that a switched line is being used for this terminal and it has unattended answering service (A in column 20). This means that the terminal operator must dial the computer system. In addition, this line specifies that the type of remote terminal is a DCT 2000 (2000 in columns 48 through 51), the permanent error indicator is 44 (44 in columns 53 and 54), and the terminal name is IN1 (IN1 in columns 71 through 74).

■ Example 2

Lines 030 and 040 on the file description specifications show that DCTERM1 and DCTERM2 are output files (O in column 15) and that they are communications files (REMOTE in columns 40 through 46).

Lines 030 and 040 on the telecommunications specifications show that DCTERM1 and DCTERM2 are transmit files (T in column 16). DCTERM1 and DCTERM2 are specified as transmit files because your program will transmit output data to these files. These lines also specify that switched lines are being used for these files and that they have manual call specified for them (M in column 20). This means the system operator must dial the number of the remote terminal. In addition, these lines specify that the type of remote terminal for both files is DCT 1000 (1000 in columns 48 through 51), the permanent error indicators are 77 and 78 respectively (77 and 78 in columns 53 and 54), and the terminal names are OUT1 and OUT2 respectively (OUT1 and OUT2 in columns 71 through 74).

### 17.3. FILE PROCESSING MODES

There are five file processing modes that you can use with remote terminals. Four of these modes are intended to be used primarily with batch terminals, that is, with the SPERRY UNIVAC DCT 1000 Data Communications Terminal, SPERRY UNIVAC DCT 2000 Data Communications Terminal, SPERRY UNIVAC 1004 Card Processor System, SPERRY UNIVAC 9200/9300 Series, the IBM 2780 Data Transmission Terminal, or devices using the binary synchronous communications (BSC) mode. The remaining mode is intended to be used with the interactive terminals, that is, with the UNISCOPE 100 Display Terminal, UNISCOPE 200 Display Terminal, SPERRY UNIVAC DCT 500 Data Communications Terminal, SPERRY UNIVAC DCT 524 Data Communications Terminal, or teletypewriter.



- **Transmit a File, Then Receive Another File**

In this mode, two files are specified for a remote terminal. The data file to be transmitted must be specified as an output file on the file description specifications and it must be specified as a transmit file on the telecommunications specifications. The data file to be received must be specified as an input file on the file description specifications and must be specified as a receive file on the telecommunications specifications. When you use this mode, your program first processes the transmit file and then, when all data has been transmitted to the remote terminal, it processes the receive file.

- **Receive a File, Then Transmit a File**

In this mode, two files are specified for a remote terminal. The data file to be received must be specified as an input file on the file description specifications and must be specified as a receive file on the telecommunications specifications. The data file to be transmitted must be specified as an output file on the file description specifications and must be specified as a transmit file on the telecommunications specifications. When you use this mode, your program first processes the receive file and then, when all data has been received from the remote terminal, it processes the transmit file.

Note that, in this mode and the transmit-a-file, then-receive-a-file mode, it is your responsibility to design your program so that a complete file is processed for a particular remote terminal before another file is processed. This can be done by processing one file during the normal RPG II processing and the other file when the last record indicator (LR) is set on. Another method would be to use demand input and/or exception on the calculation specifications so that all the records of one file are processed first after which all the records of the second file are processed.

Note also that a particular remote terminal is not limited to one input (receive) file and one output (transmit) when you use this mode or the transmit-a-file, then-receive-a-file mode. You can process more than two files on a particular remote terminal if your program is arranged so that the processing does not overlap.

➔ The processing mode used with interactive terminals is as follows:

- **Transmit with Reception of Conversational Reply**

In this mode, a record is transmitted to a remote terminal and a reply is received back from the remote terminal. The file must be specified as a combined file on the file description specifications and must be specified as a transmit file on the telecommunications specifications. The file may be specified as the primary file.

➔













CALCULATION SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	CONDITIONS															CALCULATION										RESULTING INDICATORS										COMMENTS	PROGRAM IDENTIFICATION
			INDICATORS															OPERATION										ARITHMETIC											
1	C	0	99															SETON										L R											

OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW										SPACE SKIP										OUTPUT INDICATORS										FIELD NAME										DATA FORMAT P/B/L/R										CODES										NOT USED	PROGRAM IDENTIFICATION																				
			TYPE H/D/T/E										BEFORE AFTER										AND AND										*AUTO										NEGATIVE VALUE INDICATION										COMMAS INSERTED												ZERO BALANCE TO PRINT										CODES ACTION									
0	O	10	INVUPDT										D										01N99										ITEMNO										1 10																																									
0	O	20																															QUANTY										1 20																																									
0	O	30																															CUSTMR										1 30																																									
0	O	40	PRNTR										H 3										IP																																																													
0	O	50	OR										OF																																																																							
0	O	70																																																																																		
0	O	80																																																																																		
0	O	90																																																																																		
1	O	0	D										2										02N99										ITEM Z										1 10																																									
1	O	10																															REMNDRZ										1 21																																									
1	O	20																															DESC										11 02																																									
1	O	30	D										3										99																																																													
1	O	40																																																																																		
1	O	50																																																																																		

Figure 17-4. Transmit a File, Then Receive Another File Using a BSC Remote Terminal (Part 2 of 2)

The control card specifications form indicates that this is a telecommunications program by the entry of a CCA name (CMNT in columns 70 through 73).

The file description specifications form shows on line 010 that SALES, the primary input file (I in column 15 and P in column 16) is a card file that is read from the card reader (READER in columns 40 through 46). Line 020 shows that NEWINV is the secondary input file (I in column 15 and S in column 16), and that it is a communications file (REMOTE in columns 40 through 46). Line 030 shows that INVUPDT is an output file (O in column 15) and that it is a communications file (REMOTE in columns 40 through 46). Line 040 shows that PRNTR is the second output file (O in column 15) and is a printer file (PRINTER in columns 40 through 46).

The telecommunications specifications form shows that NEWINV is specified as a receive file (R in column 16) and INVUPDT is specified as a transmit file (T in column 16). These files use the same remote terminal (BSC in columns 48 through 51 and MAIN in columns 71 through 74), a switched line with unattended answering service is used (A in column 20), and the permanent error indicator for both files is 99 (99 in columns 53 and 54).

The input format specifications form shows the format of the input records for the primary input file, SALES, on lines 010 through 040 and that indicator 01 is set on when a record is read from this file. Lines 050 through 080 show the format of the input records for the secondary input file, NEWINV (on the BSC remote terminal), and that indicator 02 is set on when a record is read from this file. Since matching fields are not specified, the primary input, SALES, is completely processed first. Then, the secondary input file, NEWINV, is processed.

The calculation specifications form shows that the last record indicator (LR) is to be set on if a permanent error occurs (indicator 99 is on).

The output format specifications form shows the format of the output records for the output file, INVUPDT, on lines 010 through 040 and that records are written to this file when indicator 01 is on. Since indicator 01 conditions output to this file, the output will be written only when the primary input file, SALES, is being read. Lines 050 through 090 show the heading that is to be printed on the printer during the first program cycle. Lines 100 through 130 show the format of the output records for the printer file. Since indicator 02 conditions output to this file, the output is written only when the secondary input is being received from the BSC remote terminal (the secondary input file NEWINV). Line 140 shows the message that is printed if a permanent error occurs (indicator 99 is on). The program then terminates because, when indicator 99 is on, the last record indicator (LR) is set on.

### 17.3.2. Using Interactive Remote Terminals

The following subsections describe how to use interactive remote terminals in RPG II telecommunications programs.

#### 17.3.2.1. UNISCOPE 100 and UNISCOPE 200 Display Terminals

When these terminals are used, the terminal operator must enter and transmit each individual data record. First, the RPG II program transmits a message. The operator then replies by entering an input record. This process continues until the operator indicates that there are no more records by typing in /\*.

Each position on the screen corresponds to a position in an RPG II input or output record. The number of characters per line (64 or 80) determines where an input record is entered, or an output record is displayed on the screen. For an 80-character screen, record position 1 corresponds to line 1, position 1; record position 80 corresponds to line 1, position 80; record position 81 corresponds to line 2, position 1; record position 160 corresponds to line 2, position 80; and so on. Table 17-1 provides a summary of the input/output record position ranges and the screen line where a record in a particular range is entered or displayed.

Table 17-1. Summary of Record Position Ranges and Screen Line Numbers for Entry or Display

64-Character Screen		80-Character Screen	
Input/Output Record Position Range	Screen Line Record Is Entered or Displayed on	Input/Output Record Position Range	Screen Line Record Is Entered or Displayed on
1-64	1	1-80	1
65-128	2	81-160	2
129-192	3	161-240	3
193-256	4	241-320	4
257-320	5	321-400	5
321-384	6	401-480	6
385-448	7	481-560	7
449-512	8	561-640	8
513-576	9	641-720	9
577-640	10	721-800	10
641-704	11	801-880	11
705-768	12	881-960	12
769-832	13	961-1040	13
833-896	14	1041-1120	14
897-960	15	1121-1200	15
961-1024	16	1201-1280	16
1025-1088	17	1281-1360	17
1089-1152	18	1361-1440	18
1153-1216	19	1441-1520	19
1217-1280	20	1521-1600	20
1281-1344	21	1601-1680	21
1345-1408	22	1681-1760	22
1409-1472	23	1761-1840	23
1473-1536	24	1841-1920	24

Since we are dealing with UNISCOPE 100 or 200 Display Terminals, a combined file is used for the input and output records. The RPG II program transmits an output record to the terminal at the beginning of the program and in each cycle thereafter to format the screen for data entry and to prompt the terminal operator to enter an input record.

Figure 17-5 shows an example of how to specify an output record that is to be transmitted to the remote terminal to prompt the operator; how to specify the input record that is to be entered as a reply; and how to use the terminal to enter the input record.











The calculation specifications form shows that a test is performed to see whether an error has occurred; that is, if the \*ERROR field contains other than hexadecimal 00, the HO indicator is set on. If the \*ERROR field contains other than hexadecimal 00, indicator 04 is set on and this, in turn, causes the HO indicator to be set off. If it does not, a CHAIN operation will be performed. This test is necessary because a multiterminal file is being used. If it were not present, an error involving one terminal could cause the program to terminate.

The output format specifications form shows on lines 010 through 030 that the prompting message, CUSTOMER NUMBER, is displayed on the remote terminal at the beginning of program execution (indicator 1P is on) or when the terminal operator types in a record consisting of the character E (indicator 99 is on). The latter is used in those cases where an error occurs and it is necessary to reformat the screen. Lines 040 through 060 show the message displayed when an error occurs that sets the HO indicator on (indicator 040 is on). Lines 080 through 100 show the message displayed when a record cannot be found (indicator 01 is on, indicator 03 is off, and indicator 04 is off). Lines 110 through 180 show the format of the output record when the CHAIN operation is successful (indicator 01 and 02 are on). Note that this record includes the prompting message for the next input record (line 120).

#### 17.3.2.2. Teletypewriter and DCT 500 Series Data Communications Terminals

When these terminals are used, the terminal operator must enter and transmit each individual data record. The RPG II program will transmit a message. The operator then replies by entering an input record. This process continues until the operator indicates that there are no more records by typing in /\*.

Each position on a line corresponds to a position in an RPG II input or output record. The maximum record length depends upon maximum number of characters per line of remote terminal that is being used; that is, 64 or 80 for teletypewriter and 132 for DCT 500 series. If a multiterminal file is used with different types of terminals, the maximum record length for the file is the shortest line length; that is, if some of the remote terminals have a line length of 80 and the others have a line length of 132, the maximum record length for that file is 80 characters.

Since we are dealing with a teletypewriter or a DCT 500 series display terminal, a combined file is specified for the input and output records. The RPG II program transmits an output record to the terminal at the beginning of program execution and in each cycle to indicate the keyboard printer format for data entry and to prompt the operator to enter an input record.

Figure 17-7 shows an example of how to specify an output record that is to be transmitted to the remote terminal to prompt the operator; how to specify the input record that is to be entered as a reply; and how to use the terminal to enter the input record.



If a hardware error occurs during transmission, the RPG II program requests retransmission of the input data by returning the carriage to the beginning of the next line and printing an R.

If recovery is not possible, the system operator is notified of a line down or terminal down status. The permanent error indicator is also set on if one is specified in the program. If there is more than one terminal in the program, RPG II continues processing. If there is only one terminal in the program, RPG II terminates the program. In a multiterminal program, the terminal operator can resume transmission if the terminal problem can be corrected.

### 17.3.2.2.1. Transmit with Reception of Conversational Reply Using Two DCT 500 Data Communications Terminals and a Teletypewriter Terminal

Figure 17-8 shows an example in which two DCT 500 data communications terminals and a teletypewriter terminal on separate switched lines are used to access an indexed sequential file. The data to be entered consists of a 3-digit item number and a 5-digit quantity. The item number is used as a key to obtain the price of the item, and this price is then multiplied by the quantity to determine the total purchase price. The item price and the total purchase price are then transmitted back to the remote terminal that requested the information. The remote terminal operators gain access to the RPG II program by dialing the central site. As each operator dials in, his requests are serviced by the RPG II program. The program can run with from one to three terminals. Program operation continues until all remote terminals have signalled end of file or they are down.

LINE	POSITION	1	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80
1	1																	
2	81																	
3	161																	
4	241																	
5	321																	
6	401																	
7	481																	
8	561																	
9	641																	
10	721																	
11	801																	
12	881																	

#### CONTROL CARD SPECIFICATIONS

PAGE NO.	LINE NO.	FORM TYPE	COMPILATION MODE	ERROR ANALYSIS DUMP	GENERATE DEBUG CODE	INVERTED PRINT	ALTERNATE COLLATING SEQUENCE	FORMS ALIGNMENT	INDICATOR INITIALIZATION	SUBROUTINE OR ACTION PROGRAM	PROGRAM IDENTIFICATION
1	2	0	10	14	15	16	17	18	19	20	21
2	3	0	10	14	15	16	17	18	19	20	21
3	4	0	10	14	15	16	17	18	19	20	21
4	5	0	10	14	15	16	17	18	19	20	21
5	6	0	10	14	15	16	17	18	19	20	21
6	7	0	10	14	15	16	17	18	19	20	21
7	8	0	10	14	15	16	17	18	19	20	21
8	9	0	10	14	15	16	17	18	19	20	21
9	10	0	10	14	15	16	17	18	19	20	21
10	11	0	10	14	15	16	17	18	19	20	21
11	12	0	10	14	15	16	17	18	19	20	21
12	13	0	10	14	15	16	17	18	19	20	21
13	14	0	10	14	15	16	17	18	19	20	21
14	15	0	10	14	15	16	17	18	19	20	21
15	16	0	10	14	15	16	17	18	19	20	21
16	17	0	10	14	15	16	17	18	19	20	21
17	18	0	10	14	15	16	17	18	19	20	21
18	19	0	10	14	15	16	17	18	19	20	21
19	20	0	10	14	15	16	17	18	19	20	21
20	21	0	10	14	15	16	17	18	19	20	21
21	22	0	10	14	15	16	17	18	19	20	21
22	23	0	10	14	15	16	17	18	19	20	21
23	24	0	10	14	15	16	17	18	19	20	21
24	25	0	10	14	15	16	17	18	19	20	21
25	26	0	10	14	15	16	17	18	19	20	21
26	27	0	10	14	15	16	17	18	19	20	21
27	28	0	10	14	15	16	17	18	19	20	21
28	29	0	10	14	15	16	17	18	19	20	21
29	30	0	10	14	15	16	17	18	19	20	21
30	31	0	10	14	15	16	17	18	19	20	21
31	32	0	10	14	15	16	17	18	19	20	21
32	33	0	10	14	15	16	17	18	19	20	21
33	34	0	10	14	15	16	17	18	19	20	21
34	35	0	10	14	15	16	17	18	19	20	21
35	36	0	10	14	15	16	17	18	19	20	21
36	37	0	10	14	15	16	17	18	19	20	21
37	38	0	10	14	15	16	17	18	19	20	21
38	39	0	10	14	15	16	17	18	19	20	21
39	40	0	10	14	15	16	17	18	19	20	21
40	41	0	10	14	15	16	17	18	19	20	21
41	42	0	10	14	15	16	17	18	19	20	21
42	43	0	10	14	15	16	17	18	19	20	21
43	44	0	10	14	15	16	17	18	19	20	21
44	45	0	10	14	15	16	17	18	19	20	21
45	46	0	10	14	15	16	17	18	19	20	21
46	47	0	10	14	15	16	17	18	19	20	21
47	48	0	10	14	15	16	17	18	19	20	21
48	49	0	10	14	15	16	17	18	19	20	21
49	50	0	10	14	15	16	17	18	19	20	21
50	51	0	10	14	15	16	17	18	19	20	21
51	52	0	10	14	15	16	17	18	19	20	21
52	53	0	10	14	15	16	17	18	19	20	21
53	54	0	10	14	15	16	17	18	19	20	21
54	55	0	10	14	15	16	17	18	19	20	21
55	56	0	10	14	15	16	17	18	19	20	21
56	57	0	10	14	15	16	17	18	19	20	21
57	58	0	10	14	15	16	17	18	19	20	21
58	59	0	10	14	15	16	17	18	19	20	21
59	60	0	10	14	15	16	17	18	19	20	21
60	61	0	10	14	15	16	17	18	19	20	21
61	62	0	10	14	15	16	17	18	19	20	21
62	63	0	10	14	15	16	17	18	19	20	21
63	64	0	10	14	15	16	17	18	19	20	21
64	65	0	10	14	15	16	17	18	19	20	21
65	66	0	10	14	15	16	17	18	19	20	21
66	67	0	10	14	15	16	17	18	19	20	21
67	68	0	10	14	15	16	17	18	19	20	21
68	69	0	10	14	15	16	17	18	19	20	21
69	70	0	10	14	15	16	17	18	19	20	21
70	71	0	10	14	15	16	17	18	19	20	21
71	72	0	10	14	15	16	17	18	19	20	21
72	73	0	10	14	15	16	17	18	19	20	21
73	74	0	10	14	15	16	17	18	19	20	21
74	75	0	10	14	15	16	17	18	19	20	21
75	76	0	10	14	15	16	17	18	19	20	21
76	77	0	10	14	15	16	17	18	19	20	21
77	78	0	10	14	15	16	17	18	19	20	21
78	79	0	10	14	15	16	17	18	19	20	21
79	80	0	10	14	15	16	17	18	19	20	21

#### FILE DESCRIPTION SPECIFICATIONS

PAGE NO.	LINE NO.	FILE NAME	FILE TYPE	FILE DESIGNATION	END OF FILE	SEQUENCE	FILE FORMAT	BLOCK LENGTH	RECORD LENGTH	FILE PROCESSING MODE	KEY OR RECORD ADDRESS FIELD LENGTH	RECORD ADDRESS TYPE	FILE ORGANIZATION	OVERFLOW INDICATOR	KEY FIELD STARTING LOCATION	EXTENSION OR LINE COUNTER CODE	DEVICE	LABELS	NUMBER OF BYTES IN MAIN STORAGE TO BE RESERVED FOR INDEX	FILE ADDITION/UNORDERED LOAD	CYLINDER OVERFLOW SPACE PERCENTAGE (X10)	NUMBER OF EXTENTS	TAPE REWIND OPTION	FILE CONDITIONERS	PROGRAM IDENTIFICATION	
1	2	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
2	3	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
3	4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
4	5	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
5	6	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
6	7	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
7	8	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
8	9	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
9	10	0	1	0	1	0																				



The control card specifications form indicates that this is a telecommunications program by the entry of a CCA name (DCTY in columns 70 through 73).

The file description specifications form shows on line 010 that the primary file, RMODTY, is a combined file (C in column 15 and P in column 16) and that it is a communications file (REMOTE in columns 40 through 46). Line 020 shows that RDSKTY is a chained input file (I in column 15 and C in column 16), the file is an indexed sequential file that is to be processed randomly by using the CHAIN operation (R in column 28, A in column 31, and I in column 32), the key field length is 3 and the key field starts in record position 1 (3 in columns 29 and 30 and 1 in columns 35 through 38), and that the file is contained on a disk (DISC in columns 40 through 46).

The telecommunications specifications form shows that RMODTY is a transmit file (T in column 16), that two DCT 500 data communications terminals and a teletypewriter terminal are being used with this file (500 on lines 010 and 030 and TTY on line 020 in column 48 through 51), and the terminal names are TRM1, TRM2, and TRM3 (TRM1, TRM2, TRM3 in columns 71 through 74). The asterisks (\*) in column 14 indicate that this is a multiterminal file; consequently, all terminals on the \* lines are associated with this file.

The input format specifications form shows the format of the input records for the combined primary file, RMODTY, on lines 010 through 040. If a record consisting of the character E is transmitted from a terminal, indicator 99 is set on. If the item number is typed in positions 21 through 23 and the quantity is typed in positions 41 through 45, indicator 01 is set on. Lines 040 and 050 show the format of the input record for the input chained file, RDSKTY, and that indicator 02 is set on when a record is read from this file.

The calculation specifications form shows that a test is performed to see whether an error has occurred; that is, if the \*ERROR field contains other than hexadecimal 00, the HO indicator is set on. If \*ERROR field contains other than hexadecimal 00, indicator 06 is set on and this, in turn, causes the HO indicator to be set off. If it does not, the CHAIN operation to find the item price and the calculation of the total price is performed. This test is necessary because a multiterminal file is being used. If it was not present, an error involving one terminal could cause the program to terminate.

The output format specifications form shows, on lines 010 through 040, the message printed when an error occurs that sets the HO indicator on (indicator 06 is on). Lines 060 and 070 show the message that is printed when a record cannot be found (indicator 01 is on, indicator 05 is off, and indicator 06 is off). Lines 080 through 130 show the format of the normal output records; that is, the CHAIN operation is successful (indicators 01 and 02 are on) and the item price and total price are printed on the printer. Lines 140 through 160 show the prompting message that is displayed at the beginning of program execution, when the normal output records (lines 080 through 130) are printed, and when the terminal operator types in the character E (used in those cases where an error occurs and it is necessary to reformat the printer).

## 17.4. NETWORK DEFINITION REQUIREMENTS FOR RPG II TELECOMMUNICATIONS PROGRAMS

As was stated previously, all RPG II telecommunications programs require that a communications network be defined at system generation time. This network definition is accomplished by using the integrated communications access method (ICAM). The fundamentals of ICAM, direct data interface, the system installation user guide/programmer reference, and the integrated communications access method (ICAM) utilities user guide contain the information required to define the communications network and create the ICAM load module that the RPG II telecommunications program interfaces with when the program is executed. The interface type used by RPG II telecommunications programs is the direct data interface (DDI) level 3; consequently, when the CCA parameters are specified, TYPE=(DDI,3) must be specified. No buffers are specified in the CCA. The buffers are generated in your program by the RPG II compiler.

Figure 17-9 shows an example of the cards that would be required to generate an ICAM load module. This example assumes that you have an existing SYSRES; consequently, two job control streams must be executed. First, the cards shown in Figure 17-9 are placed in the system card reader and RUN SG\$PARAM is typed on the system console.

The cards are read and checked for errors. The cards are printed on the system printer along with any errors. If there are no errors you may proceed by typing RUN SG\$COMMK on the system console. The ICAM modules required by your network are assembled and the load module is linked and stored on the SYSRES.

1	10	16
1.	COMMCT	
2.	NET1	CCA TYPE=(DDI,3)
3.		BUFFERS 0,0,0,ARP=15
4.	LNE1	LINE DEVICE=(1004),TYPE=(2000,SYNC,SWCH,UNAT),ID=04
5.	TEN4	TERM FEATURES=(1004),ANSWER=(5,C,1004),AUX1=(PCH)
6.	LNE2	LINE DEVICE=(UNISCOPE),TYPE=(2000,SYNC,SWCH,UNAT),ID=08
7.	SCP1	TERM ADDR=(28,51),FEATURES=(U200)
8.	SCP2	TERM ADDR=(28,52),FEATURES=(U200)
9.	SCP3	TERM ADDR=(29,53),FEATURES=(U200)
10.	SCP4	TERM ADDR=(29,54),FEATURES=(2000)
11.		ENDCCA
12.		MCP MCPNAME=C7
13.		CACH=(04,NET1,01)
14.		CACH=(08,NET1,02)
15.	END	
16.	// FIN	

Figure 17-9. ICAM Load Module Generation

---

<u>Line</u>	<u>Explanation</u>
1	An ICAM generation is being done.
2	The network name (NET1) and the interface type (DDI, level 3) are specified.
3	No buffers are needed. They are generated in your program. There are to be 15 activity request packets.
4	The first line name is LNE1. It is a 1004 line that is a 2000 baud synchronous transmission line. It is switched with automatic answering. The port ID is 4.
5	The terminal name is defined as TEN4. It is a 1004. The site ID is 1004. There is a card punch attached to the 1004.
6	The second line name is LNE2. It is a UNISCOPE terminal line that is a 2000 baud synchronous transmission line. It is switched with automatic answering. The port ID is 8.
7-10	Four UNISCOPE 200 terminals are defined and named SCP1-SCP4.
11	End of CCA
12	The ICAM module name is defined as C7.
13	Port 4 is assigned to NET1 line 1.
14	Port 8 is assigned to NET1 line 2.
15	End of system generation parameters
16	End of job control stream

You do not need complete information about the communications network because you are only concerned with the network information you must enter on the RPG II specifications forms when you write your program and the ICAM load module name (MCP name) when you execute your program. Additional remote terminals may be added to the network or the network configuration may be changed without requiring you to change your RPG II program. If, however, a remote terminal is moved from one multiplexer to another, it may be necessary for you to change the configuration entry (column 15) on the telecommunications specifications form.

## 17.5. PROGRAM EXECUTION

The ICAM load module must be loaded before an RPG II telecommunications program can be executed. The module is loaded by typing in the module name (MCP name) on the system console. If you attempt to execute an RPG II telecommunications program without first loading the ICAM load module, the program terminates abnormally and the supervisor error code 21 is displayed on the system console.

After the type-in has been made, the program is executed by using the normal job control statements. When the RPG II program files are opened, ICAM messages are displayed on the system console if switched lines are being used. If unattended answering was specified, the ICAM message MC#98 is displayed. If manual dialing was specified, the ICAM message MC#03 is displayed. After the line connections have been made, the execution of the RPG II program continues. Additional messages may be displayed on the system console as required by line and terminal conditions. The system messages programmer/operator reference describes all ICAM messages.

## 17.6. ERROR PROCESSING

The normal ICAM error recovery procedures should be tried in the event a line or terminal error occurs. If recovery is not successful and you have specified a permanent error indicator in your program, that indicator is set on. If not, the HO indicator is set on and the program terminates, unless the HO indicator is set off.



**PART 5. PROCESSING YOUR RPG II PROGRAM**



## 18. Compiling, Link Editing, and Executing

### 18.1. GENERAL

Your RPG II programs are processed by the SPERRY UNIVAC Operating System/3 (OS/3). The operating system is an organized collection of programs executed as required to schedule and process user programs. The programs include the RPG II compiler, service programs, and the supervisor. The supervisor control program supervises the execution of the processing programs and controls the location, storage, and retrieval of data. It also schedules jobs for continuous processing.

A request to the operating system for facilities and scheduling of program execution is called a job. A job consists of one or more job steps, each of which specifies execution of a program. You make these requests by use of job control statements that supply the information that the operating system needs to perform the job.

Three job steps are involved in compiling, link editing, and executing a program. Each step can be done separately as a job by itself, or in combination with either or both of the other job steps.

- **Compilation**

Compilation is the process by which your source program is translated by the RPG II compiler into executable machine language. The result of this compilation is called an object module. This object module cannot be executed until it is processed by the linkage editor.

- **Link Editing**

The object module is processed by the linkage editor at this point. The linkage editor transforms the object module into a relocatable load module. Your program is now in a form that can be loaded into the system and executed.

- **Execution**

The final step in the complete processing of a program is execution. The load module is loaded into the system at this point, and processing begins. The input files are read, the calculations are performed, and the desired results are produced.

## 18.2. JOB CONTROL STATEMENTS

The job control statements required to process RPG II programs vary (depending on the application) to such an extent that they are beyond the scope of this manual. Job control statements for a specific job, however, always begin with a JOB statement and end with a /& (end of job) statement.

A specific job consists of one or more job steps. Each job step is initiated by an EXEC statement. Preceding the EXEC statement are job control statements that assign the input/output devices and allocate the file space that is required to execute the job step. These are the DVC, VOL, EXT, LBL, and LFD job control statements. These statements must always appear in the order given. The DVC statement is always the first in a device assignment set, and the LFD statement is always the last. The use of the other statements depends on the device that is required. The EXEC statement can be followed by job control statements that direct how the job step is to be executed or indicate the start and end of data that is related to the job step. The PARAM statement is used to direct job step execution, the /\$ statement indicates the start of data, and the /\* statement indicates the end of data. For more detailed information on job control statements, refer to the job control user guide.

The first method is for you to use one of the job control procedure call statements provided by Sperry Univac. By using this method, you can save considerable time because you do not have to write the individual job control statements. Instead, the job control procedure call statement automatically generates them for you.

There are three RPG II job control procedure call statements: RPG, RPGL, and RPGLG. RPG allows you to compile your source program; RPGL allows you to compile your source program and link-edit the generated object module to create a load module; and RPGLG allows you to compile, link-edit, and immediately execute the load module.

There are four RPG II auto report job control procedure call statements: AUTO, AUTRPG, AUTRPGL, and AUTRPGLG. AUTO allows you to process auto report source programs. AUTRPG allows you to process auto report source programs and then compile the RPG II generated source programs. AUTRPGL allows you to process auto report source programs, compile the generated RPG II source program, and link-edit the generated object module to create a load module. AUTRPGLG allows you to process auto report source programs, compile the generated RPG II source program, link-edit the generated object module to create a load module, and immediately execute the load module.

The second method is for you to write all the job control statements for each job step. This tends to be time-consuming because of the effort required to write the individual statements.

### 18.2.1. RPG II Job Control Procedure Call Statements - RPG, RPGL, and RPGLG

These procedure call statements generate the necessary job control statements to run the RPG II language processor. Optionally, they can generate the job control statements that specify the following:

- input-source library (RPG, RPGL, and RPGLG);
- output-object library (RPG only); and
- PARAM control statements to define the format of the compiler listing.

## Format:

```

//[symbol] { RPG
           { RPGL
           { RPGLG } [ PRNTR={ lun[,dest]
                    { N[,dest]
                    { *[,dest]
                    [ ,IN={ (vol-ser-no, iblname)
                          { (RES)
                          { (RES, iblname)
                          { (RUN, iblname)
                          { (*, iblname)
                    [ ,OUT={ (vol-ser-no, iblname)
                          { (RES, iblname)
                          { (RUN, iblname)
                          { (*, iblname)
                          { N
                          { (RUN, SYSRUN)
                    [ ,LST={ (K)
                          { (M)
                          { (N)
                          { (S)
                    [ ,SCR1={ vol-ser-no } [ ,SCR2={ vol-ser-no }
                          { RES } [ ,SCR2={ RUN }
                    [ ,EMB={ YES } [ ,MOD={ 3 } [ ,COL=7 ]
                          { NO } [ ,MOD={ 4 }
                          { 5 }
                          { IRAM }
                    [ ,ALTLOD={ (vol-ser-no, iblname)
                          { (RES, iblname)
                          { (RUN, iblname)
                          { (*, iblname)
                          { (RES, SYSRUN)
                    [ ,ERRFIL=(vol-ser-no, iblname, module-name) ]
                    [ ,CONSOLE=ifname ]
                    [ ,MIRAM=ALL ]

```

## Label:

## symbol

Specifies the 1- to 6-character source module name; only needed when the IN parameter is used.

## Operation:

## RPG

A procedure call statement used to compile an RPG II source program.

## RPGL

A procedure call statement used to compile an RPG II source program and link-edit the generated object module to create a load module.

**RPGLG**

A procedure call statement used to compile an RPG II source program, link-edit the generated object module to create a load module, and immediately execute the load module.

**Keyword Parameter PRNTR:**

Specifies the logical unit number of the printer. The options are:

**PRNTR lun[.dest]**

The logical unit number of the printer. The identifier, dest, is a destination identifier of 1 to 6 alphanumeric characters for the remote destination for spooling.

**PRNTR N[.dest]**

Specifies that a DVC-LFD sequence of job control statements is not to be generated for a printer. If you use this option, you must supply your own printer definition. Note that this option also allows you to use // LCB and // VFB job control statements. The identifier, dest, is a destination identifier of 1 to 6 alphanumeric characters for the remote destination for spooling.

If omitted, the logical unit number of the printer is 20.

**Keyword Parameter IN:**

Specifies the input file definition and generates a PARAM IN control statement. The options are:

**IN (vol-ser-no.lblname)**

Specifies the file identifier and the volume serial number (vol-ser-no) where the source input is located.

**IN (RES)**

Specifies that the source input is located on the SYSRES device in \$Y\$SRC.

**IN (RES.lblname)**

This is used if your source input is located on the SYSRES device, but the file identifier is of your own specification, not \$Y\$SRC.

**IN (RUN.lblname)**

Specifies that the source input is located on the job's \$Y\$RUN file, with the file identifier of your own specification.

**IN- (\*.lblname)**

Specifies that the source input is located on a catalog file identified by the file identifier.

If omitted, the source input is in the form of embedded data cards ( /\$.source deck, \*) in the job control stream.

Keyword Parameter OUT (used only with RPG; not used with RPGL or RPGLG):

Specifies the output file definition and generates a PARAM OUT control statement. The options are:

OUT= (vol-ser-no. lblname)

Specifies the file identifier and the volume serial number where the object module is located.

OUT= (RES. lblname)

Specifies that the object module is located on the SYSRES device with the file identifier of your own specification.

OUT= (RUN. lblname)

Specifies that the object module is located on the job's \$Y\$RUN file, with a file identifier of your own specification.

OUT= (\* . lblname)

Specifies that the object code is to be located on a catalog file identified by the file identifier.

OUT= (N)

Specifies that an object module is not to be produced.

If omitted, the object module is located on the job's \$Y\$RUN file.

Keyword Parameter LST:

Specifies the format of the compiler listing. The options are:

LST K

Do not print error flags for sequence errors.

LST M

Do not print the source language statements and error messages.

LST N

Do not print any listings.

LST S

Do not print the main storage map.

If omitted, the complete compiler listing is printed.

**Keyword Parameter SCR1:**

Specifies the volume serial number of the work file labeled SCR1.

**SCR1 vol-ser-no**  
The volume serial number.

If omitted, the default is RES.

**Keyword Parameter SCR2:**

Specifies the volume serial number of the work file labeled SCR2.

**SCR2 vol-ser-no**  
The volume serial number.

If omitted, the default is RUN.

**Keyword Parameter EMB:**

Specifies whether or not embedded linkage editor control statements are to be generated in the RPG II object module.

**EMB NO**  
Do not generate embedded linkage editor control statements in the RPG II object module. No overlay structure is generated for the load table or dump table file.

If omitted, the default is YES.

**Keyword Parameter MOD:**

↓  
Specifies that the program is to be compiled in the IBM System/3-System/34 mode or that if the program is to be compiled in one of the other compilation modes, IRAM is to be used to process all disk files in a Series 90 environment and MIRAM is to be used in a System 80 environment.

**MOD=3**  
↑ The program is to be compiled in the IBM System/3-System/34 mode. When this mode is specified, IRAM or MIRAM will be used to process all disk files. Also, the logical file definition (// LFD) for printer files is changed to PRNTR, PRNTR1...PRNTRn and the control reader (CTRLDR) is used for card input even though the data management reader (READER) is specified.

**MOD=4**  
Same as MOD=3 except that printer files are generated with the same names as used in the program and reader files use the data management reader (READER).

→ **MOD=5**  
The program is to be compiled in the IBM System/3-System/34 mode with native mode data management accessing the disk files.



**MOD IRAM**

IRAM is to be used to process all disk files.

If omitted, the program is compiled in OS/3 native mode.

**Keyword Parameter COL:**

Specifies that compile time tables can begin in position 7 rather than in position 1.

**COL 7**

Enables the user to carry sequence numbers in columns 1-5 for each statement in the RPG source program. The OS/3 librarian can then be used to maintain disk file source for RPG programs that contain compile time tables and arrays.

If omitted, compile-time tables begin in position 1.

**Keyword Parameter ALTLOD:**

Specifies the location of the alternate load library.

**ALTLOD (vol-ser-no.lblname)**

Specifies the volume serial number (vol-ser-no) and the file identifier of an alternate load library that contains the RPG II compiler.

**ALTLOD (RES.lblname)**

Specifies that the alternate load library is located on the job's SYSRES device, in the file identified by the file identifier.

**ALTLOD (RUN.lblname)**

Specifies that the alternate load library is located on the job's \$Y\$RUN file with the file identifier specified by the user.

**ALTLOD (\*.lblname)**

Specifies that the alternate load library is located on a catalog file identified by the file identifier.

If omitted, the compiler is loaded from \$Y\$RUN.

**Keyword Parameter ERRFIL:**

Specifies that error diagnostic messages are written to a file that is accessed by the error file processor. When you specify this parameter, error records are created for every error note generated by the compiler.

**ERRFIL (vol-ser-no.lblname.module-name)**

vol-ser-no specifies the volume serial number of the file. lblname specifies the file identifier (name of the file that the module is placed into). module-name is the name of the module that is referenced by the error file processor.

If omitted, the error file is not created.

`ERRFIL=module-name/lfdname`

The module name is the name assigned to the error by the file element being created by the compiler. The module name may be from 1 to 8 characters in length and it doesn't need to match or be related to the RPG II source or object module name. The lfdname is the logical file name of the error log file and can be from 1 to 8 characters in length.

If omitted, the error log file is not created.

#### Keyword Parameter **CONSOLE**:

Specifies that the file is a **CONSOLE** (interactive data entry) file and not a system console file.

`CONSOLE=lfdname`

The lfdname is the name of the **CONSOLE** file.

If omitted, the file is a system console file and not an interactive data entry file.



#### Keyword Parameter **MIRAM**:

Specifies that **MIRAM** is used for disk files when operating under the Series 90 environment. To select individual files for **MIRAM**, use the **MIRAM** parameter with the **PARAM** statement.

`MIRAM=ALL`

All disk files use **MIRAM**.

If omitted, disk files use **IRAM**, **ISAM**, **DAM**, or **SAM**.



#### **18.2.1.1. Using the RPG Job Control Procedure Call Statement**

Figure 18-1 shows how you can use the RPG job control procedure call to compile an RPG II program. In this example, assume that you only want to compile your program with the standard compiler default options; that is:

- the source language statements are read from the source deck in the job control stream;
- a listing is printed that contains the error flags for sequence errors, a main storage map, the source language statements, and any associated error messages; and
- the object module that is produced is stored in the job's `$Y$RUN` file.

	1	10	20	30	40	50	60
1.	//	JOB	RPGCPO				
2.	//	RPG					
3.	/\$						
4.		.					
5.		source	deck				
6.		.					
7.	/*						
8.	/&						
9.	//	FIN					

Figure 18—1. Using the RPG Job Control Procedure Call Statement with Standard Default Options



In Figure 18-1, line 1 indicates that the name of the job is RPGCPO.

Line 2 indicates that the RPG II job control procedure RPG is being called. There are no keyword parameters specified; consequently, the standard compiler default options apply.

Line 3 indicates the start-of-data.

Lines 4 through 6 represent the source deck that contains the source language statements for the program to be compiled.

Line 7 indicates the end-of-data.

Line 8 indicates the end-of-job.

Line 9 causes the card reader (job control stream reader) to stop reading cards.

Figure 18-2 shows you the job control statements that are generated by the entries in Figure 18-1.

	1	10	20	30	40	50	60
1.	//	JOB	RPGCPO				
2.	//	DVC	20 //	LFD	PRNTR		
3.	//	DVC	RES				
4.	//	EXT	ST.C.3.CYL.1				
5.	//	LBL	\$\$SCR1 //	LFD	\$\$SCR1		
6.	//	DVC	RUN				
7.	//	EXT	ST.C.3.CYL.1				
8.	//	LBL	\$\$SCR2 //	LFD	\$\$SCR2		
9.	//	EXEC	RPGII				
10.	/\$						
11.							
12.			source deck				
13.							
14.	/*						
15.	/&						
16.	//	FIN					

Figure 18-2. Job Control Statements Generated by the RPG Procedure Call Statement with Standard Default Options

In Figure 18-2, line 1 indicates that the job name is RPGCPO.

Line 2 indicates the default logical unit number and LFD name of the printer.

Lines 3 through 5 indicate that the first work file needed for compilation is, by default, on the SYSRES device, has both a file identifier and LFD name of \$\$SCR1, and uses the system access technique; and that the allocation is contiguous, with three cylinders allocated for the secondary increment and one cylinder allocated for the first extent.

Lines 6 through 8 indicate that the second work file needed for compilation is, by default, on the device containing the job's \$Y\$RUN file. Both the file identifier and the LFD name are \$SCR2, and the file extent specification is the same as the first work file.

Line 9 loads the RPG II compiler from \$Y\$LOD.

Line 10 indicates the start-of-data.

Lines 11 through 13 represent the source deck that contains the source language statements for the program to be compiled.

Line 14 indicates the end-of-data.

Line 15 indicates end-of-job.

Line 16 causes the card reader (job control stream reader) to stop reading cards.

### 18.2.1.2. Using a Job Control Procedure Call Statement that Defines Keyword Parameters

Figure 18-3 shows how you can use the keyword parameters in a job control procedure call statement to specify other than the standard default options. In this example, assume that you are using the RPG job control procedure to compile your program and you want to use a printer other than the default printer; you want to read the source language statements for your program from an input file rather than from cards within the job control stream deck; you want to specify a different device for the first work file needed for compilation; you want to store the object module on the SYSRES device with your own file identifier; and you do not want any compiler listings.

	1	10	20	30	40	50	60	72
1.	//	JOB	RPGCPO					
2.	//	PROGNM	RPG	PRNTR	21	IN	(DSC1.U\$SRC).	X
3.	//	1	OUT	(RES.U\$OBJ).				X
4.	//	2	SCR1	DSC1.LST=N				
5.	//	&						
6.	//	FIN						

Figure 18-3. Using the RPG Job Control Procedure Call Statement with Nonstandard Options

In Figure 18-3, line 1 indicates that the name of the job is RPGCPO.

Line 2 indicates that the RPG II job control procedure RPG is being called. The source module name is PROGNM. The logical unit number of the printer is 21, and the input file has a volume serial number of DSC1, with a file identifier of U\$SRC.

Line 3 indicates that the output file is to be stored on the SYSRES device, with a file identifier of U\$OBJ.

Line 4 indicates that the first work file needed for compilation is on the device with a volume serial number of DSC1. By default, the device for the second work file is the one that contains the job's \$YSRUN file. The LST keyword parameter specifies that no compiler listings are to be printed.

Line 5 indicates the end-of-job.

Line 6 causes the card reader (job control card reader) to stop reading cards.

Figure 18-4 shows you the job control statements that are generated by the entries in Figure 18-3.

	1	10	20	30	40	50	60
1.	//	JOB	RPGCPO				
2.	//	DVC	21 //	LFD	PRNTR		
3.	//	DVC	50 //	VOL	DSC1		
4.	//	LBL	U\$SRC //	LFD	INRPUT		
5.	//	DVC	RES				
6.	//	LBL	U\$OBJ //	LFD	OUTRPUT		
7.	//	DVC	50 //	VOL	DSC1		
8.	//	EXT	ST.C.3.CYL.1				
9.	//	LBL	\$SCR1 //	LFD	\$SCR1		
10.	//	DVC	RUN				
11.	//	EXT	ST.C.3.CYL.1				
12.	//	LBL	\$SCR2 //	LFD	\$SCR2		
13.	//	EXEC	RPGII				
14.	//	PARAM	IN=PROGNM/INRPUT				
15.	//	PARAM	OUT=OUTRPUT				
16.	//	PARAM	LST N				
17.	//	&					
18.	//	FIN					

Figure 18-4. Job Control Statements Generated by RPG Procedure Call Statement with Nonstandard Options

In Figure 18-4, line 1 indicates that the name of the job is RPGCPO.

Line 2 indicates that the printer is to be assigned to the logical unit number 21, with an LFD name of PRNTR. This was generated by the PRNTR parameter on line 2 of Figure 18-3.

Line 3 indicates that the input file volume serial number is DSC1. It assigned the device with a logical unit number of 50, which was the first available number in the range 50 through 54. This was generated by the IN parameter on line 2 of Figure 18-3.

Line 4 indicates that the input file is labeled U\$SRC with an LFD name of INRPUT. This was generated by the IN parameter on line 2 in Figure 18-3.

Line 5 indicates that the output file is to be stored on the SYSRES device. This was generated by the OUT parameter on line 3 in Figure 18-3.

Line 6 indicates that the output file is labeled U\$OBJ with an LFD name of OUTRPUT. This was generated by the OUT parameter on line 3 in Figure 18-3.

Lines 7 through 9 indicate that the first work file needed for compilation has a volume serial number of DSC1. Since this volume serial number was already used, this work file uses the same device logical number of 50. This work file has the file identifier and LFD name of \$SCR1. It uses the system access technique; allocation is contiguous, with three cylinders allocated for the secondary increment and one cylinder allocated for the first extent. This was generated by the \$SCR1 parameter on line 4 in Figure 18-3.

Lines 10 through 12 indicate that the second work file needed for compilation is, by default, on the device containing the job's \$Y\$RUN file. This work file has the file identifier and LFD name of \$SCR2, and it has the same file extent specification as the first work file.

Line 13 loads the RPG II compiler from \$Y\$LOD.

Lines 14 through 16 are PARAM control statements that identify the processing options for the RPG II compiler. On line 14, the module name PROGNM is generated from the label field on line 2 in Figure 18-3.

The file name INRPUT is generated automatically when the IN parameter is used. On line 15, the filename OUTRPUT is generated automatically when the OUT parameter is used. On line 16, the N list option is generated by the LST parameter on line 4 in Figure 18-3.

Line 17 indicates end-of-job.

Line 18 causes the card reader (job control stream reader) to stop reading cards.

### 18.2.1.3. Using the RPGL Job Control Procedure Call Statement

Figure 18-5 shows you how you can use the RPGL job control procedure call to compile and link-edit an RPG II program with the standard compiler default options as described in 18.2.1.1.

	1	10	20	30	40	50	60
1.	//	JOB	RPGCLO				
2.	//	RPGL					
3.	/\$						
4.							
5.		source	deck				
6.							
7.	/*						
8.	/&						
9.	//	FIN					

Figure 18-5. Using the RPGL Job Control Procedure Call Statement



In Figure 18-5, line 1 indicates that the name of the job is RPGCLO.

Line 2 indicates that the RPGL job control procedure is being called. There are no keyword parameters specified; consequently, the standard compiler default options apply. The program is link-edited immediately after compilation, and the load module is stored on the \$Y\$RUN file.

Line 3 indicates the start-of-data.

Lines 4 through 6 represent the source deck that contains the source language statements for the program to be compiled.

Line 7 indicates end-of-data.

Line 8 indicates end-of-job.

Line 9 causes the card reader (job control stream reader) to stop reading cards.

Figure 18-6 shows you the job control statements that are generated by the entries in Figure 18-5.

	1	10	20	30	40	50	60
1.	//	JOB	RPGCLO				
2.	//	OPTION	LINK				
3.	//	DVC	20 // LFD	PRNTR			
4.	//	DVC	RES				
5.	//	EXT	ST.C.3.CYL.1				
6.	//	LBL	\$SCR1 // LFD	\$SCR1			
7.	//	DVC	RUN				
8.	//	EXT	ST.C.3.CYL.1				
9.	//	LBL	\$SCR2 // LFD	\$SCR2			
10.	//	EXEC	RPGII				
11.	/\$						
12.	.						
13.		source	deck				
14.	.						
15.	..*						
16.	..&						
17.	//	FIN					

Figure 18-6. Job Control Statements Generated by RPGL Procedure Call Statement

In Figure 18-6, line 1 indicates that the job name is RPGCLO.

Line 2 indicates that the program is link-edited immediately after compilation.

Line 3 indicates the default logical unit number and the LFD name of the printer.

Lines 4 through 6 indicate (1) that the first work file needed for compilation is, by default, on the SYSRES device, has both a file identifier and LFD name of \$SCR1, and uses the system access technique and (2) that the allocation is contiguous, with three cylinders allocated for the secondary increment and one cylinder allocated for the first extent.

Lines 7 through 9 indicate that the second work file needed for compilation is, by default, on the device containing the job's \$Y\$RUN file. Both the file identifier and the LFD name are \$SCR2, and the file extent specification is the same as the first work file.

Line 10 loads the RPG II compiler from \$Y\$LOD.

Line 11 indicates start-of-data.

Lines 12 through 14 represent the source deck that contains the source language statement to be compiled.

Line 15 indicates end-of-data.

Line 16 indicates end-of-job.

Line 17 causes the card reader (job control stream reader) to stop reading cards.

#### 18.2.1.4. Using the RPGLG Job Control Procedure Call Statement

Figure 18-7 shows how you can use the RPGLG job control procedure to compile, link-edit, and execute an RPG II program with the standard compiler default options as described in 18.2.1.1.

	1	10	20	30	40	50	60
1.	//	JOB	RPGLGO				
2.							
3.		device	assignments				
4.		for	your	program			
5.							
6.	//	RPGLG					
7.	/S						
8.							
9.		source	deck				
10.							
11.	/*						
12.	/&						
13.	//	FIN					
14.		data					
15.	/*						

Figure 18-7. Using the RPGLG Job Control Procedure Call Statement

In Figure 18-7, line 1 indicates that the name of the job is RPGCLGO.

Lines 2 through 5 represent the job control statements that assign the input/output devices and allocate the file space required to execute your program.

Line 6 indicates that the RPG II job control procedure RPGLG is being called. There are no keyword parameters specified; consequently, the standard compiler default options apply. The program is link-edited immediately after compilation and it is executed immediately after link-editing.

Line 7 indicates start-of-data.

Lines 8 through 10 represent the source deck that contains the source language statements for the program to be compiled.

Line 11 indicates end-of-data.

Line 12 indicates end-of-job.

Line 13 causes the card reader (job control stream reader) to stop reading cards.

Line 14 indicates data (on cards).

Line 15 indicates end-of-data.

Figure 18-8 shows you the job control statements that are generated by the entries in Figure 18-7.

```
1 // JOB RPGCLGO
2
3     device assignments
4     for your program
5
6 // OPTION LINK.GO
7 // DVC 20 // LFD PRNTR
8 // DVC RES
9 // EXT ST.C.3.CYL.1
10 // LBL $SCR1 // LFD $SCR1
11 // DVC RUN
12 // EXT ST.C.3.CYL.1
13 // LBL $SCR2 // LFD $SCR2
14 // EXEC RPGII
15 /$
16
17     source deck
18
19 /*
20 /&
21 // FIN
22     data
23 /*
```

Figure 18-8 Job Control Statements Generated by RPGLG Procedure Call Statement

In Figure 18-8, line 1 indicates that the name of the job is RPGCLGO.

Lines 2 through 5 represent the job control statements that assign the input/output devices and allocate the file space required to execute your program.

Line 6 indicates that the program is link-edited immediately after compilation and executed immediately after link-editing.

Line 7 indicates the default logical unit number and the LFD name of the printer.

Lines 8 through 10 indicate (1) that the first work file needed for compilation is, by default, on the SYSRES device, has both a file identifier and LFD name of \$SCR1, and uses the system access technique and (2) that the allocation is contiguous, with three cylinders allocated for the secondary increment and one cylinder allocated for the first extent.

Lines 11 through 13 indicate that the second work file needed for compilation is, by default, on the device containing the job's \$Y\$RUN file. Both the file identifier and the LFD name are \$SCR2, and the file extent specification is the same as the first work file.

Line 14 loads the RPG II compiler from \$Y\$LOD.

Line 15 indicates the start-of-data.

Lines 16 through 18 represent the source deck that contains the source language statements for the program to be compiled.

Line 19 indicates the end-of-data.

Line 20 indicates the end-of-job.

Line 21 causes the card reader (job control stream reader) to stop reading cards.

Line 22 indicates data.

Line 23 indicates end-of-data.

When you use the RPGLG job control procedure call statement, you create and name both an object module and a load module, temporarily store it in the job's \$Y\$RUN file, and then execute it. The load module is stored in the job's \$Y\$RUN file until execution of the job is completed.

If your system has the shared data management feature and you want to execute your program with the feature, you cannot use the RPGLG job control procedure call statement because job control must scan the load module in \$Y\$LOD for this feature. The GO option generated by the RPGLG job control procedure call statement cannot be used.

If you want to execute your program and use the shared data management feature, you can do this by using the RPG job control procedure call with a separate LINK job control procedure call statement or the RPGL job control procedure call statement to compile and link-edit your program and then using a separate EXEC statement to execute your program.

If your system has the shared data management feature and you use the RPGLG job control procedure call statement, your program will be executed without using the shared data management feature and the message K085-GO OPTION IN EFFECT-SHARE PARAMETER RESET will be printed to inform you that shared data management was not used.

### 18.2.2. Auto Report Job Control Procedure Call Statements (AUTO, ATRPG, ATRPGL, and ATRPGLG)

These procedure call statements generate the necessary job control statements to run RPG II auto report. Optionally, they can generate the job control statements that specify the following:

- Input – auto report source library (all jprocs)
- Output – object program object library (ATRPG jproc only)
- Output – auto report source library identification (all jprocs)
- PARAM control statements to define the format of the RPG II compiler listings (ATRPG, ATRPGL, and ATRPGLG jprocs)
- PARAM control statements to control auto report generation of the RPG II program skip to channel or skip to line printer specifications (all jprocs)
- PARAM control statements to control generation of the program identification in columns 75 – 80 of the generated code
- Other RPG II compiler control options (ATRPG, ATRPGL, and ATRPGLG jprocs)

Format:

```

//[symbol] { AUTO
            { ATRPG
            { ATRPGL
            { ATRPGLG } [ PRNTR={ l un[.dest]
                       { N[.dest]
                       { 20[.dest]
                       }
                       ]
            [ .IN={ (vol-ser-no, lblname)
                  (RES)
                  { (RES, lblname)
                  (RUN, lblname)
                  (*, lblname)
                  }
            ]
            [ .OUT={ (vol-ser-no, lblname)
                   (RES, lblname)
                   (RUN, lblname)
                   (*, lblname)
                   N
                   (RUN, $YSRUN)
            ]
            [ ,OUTSRC={ vol-ser-no, lblname, lfdname, module-name }
                  { RES, lblname, lfdname, module-name }
            ]
            [ LST={ K
                  M
                  N
                  S
            ]
  
```

(continued)

```

[ ,SCR1={vol-ser-no} [ ,SCR2={vol-ser-no} ]
[ ,EMB={YES} [ ,MOD={3
              {NO} ] [ 4
                      {5
                      |RAM} ] ] ]
[ ,SKIP=C ]
[ ,COPYn={ (vol-ser-no, lblname, lfdname)
           { (RES, lblname, lfdname)
           { (RUN, lblname, lfdname) } } ] ]
[ ,ALTL0D={ (vol-ser-no, lblname)
           { (RES, lblname)
           { (RUN, lblname)
           { *, lblname)
           { RES, $$$LOD } } } } ] ]
[ ,ERRFIL=(vol-ser-no, lblname, module-name) ]
[ ,PROGID=N ]

```

**Label:****symbol**

Specifies the 1- to 6-character source module name; only needed when the IN parameter is used.

**Operation:****AUTO**

A procedure call statement used to process auto report source programs.

**AUTRPG**

A procedure call statement used to process auto report source programs and then compile the RPG II generated source program.

**AUTRPL**

A procedure call statement used to process auto report source programs, compile the generated RPG II source program, and link-edit the generated object module to create a load module.

**AUTRPLG**

A procedure call statement used to process auto report source programs, compile the generated RPG II source program, link-edit the generated object module to create a load module, and immediately execute the load module.

**Keyword Parameter PRNTR:**

Specifies the logical unit number of the printer. The options are:

**PRNTR=|un[.dest]**

The logical unit number of the printer. The identifier, dest, is a destination identifier of 1 to 6 alphanumeric characters for the remote destination for spooling.

**PRNTR N[.dest]**

Specifies that a DVC-LFD sequence of job control statements is not to be generated for a printer. If you use this option, you must supply your own printer definition. Note that this option also allows you to use // LCB and // VFB job control statements. The identifier, dest, is a destination identifier of 1 to 6 alphanumeric characters for the remote destination for spooling.

If omitted, the logical unit number of the printer is 20.

**Keyword Parameter IN:**

Specifies the input file definition and generates a PARAM IN control statement. The options are:

**IN (vol-ser-no.lblname)**

Specifies the file identifier and the volume serial number (vol-ser-no) where the source input is located.

**IN (RES)**

Specifies that the source input is located on the SYSRES device in \$Y\$SRC.

**IN (RES.lblname)**

This is used if your source input is located on the SYSRES device, but the file identifier is of your own specification, not \$Y\$SRC.

**IN (RUN.lblname)**

Specifies that the source input is located on the job's \$Y\$RUN file, with the file identifier of your own specification.

**IN (\*.lblname)**

Specifies that the source input is located on a catalog file identified by the file identifier.

If omitted, the source input is in the form of embedded data cards (^\*,source deck ^\*) in the job control stream.

**Keyword Parameter OUT (used only with ATRPG, not used with ATRPGL or ATRPGLG):**

Specifies the output file definition and generates a PARAM OUT control statement for the RPG II compiler. The options are:

**OUT: (vol-ser-no.lblname)**

Specifies the file identifier and the volume serial number where the object module is located.

**OUT (RES.lblname)**

Specifies that the object module is located on the SYSRES device with the file identifier of your own specification.



OUT=(RUN, lblname)

Specifies that the object module is located on the job's \$Y\$RUN file, with a file identifier of your own specification.

OUT=(\*, lblname)

Specifies that the object code is to be located on a catalog file identified by the file identifier.

OUT=(N)

Specifies that an object module is not to be produced.

If omitted, the object module is located on the job's \$Y\$RUN file.

#### Keyword Parameter OUTSRC:

Specifies the output source file definition for the cataloged source file option on the auto report (U) specification. A DVC, VOL, LBL, and LFD control statement sequence is generated. The options are:

OUTSRC=(vol-ser-no, lblname, lfdname, module-name)

Specifies the volume serial number, the file identifier (lblname), the logical file descriptor (lfdname) where the generated source is located, and the module name of the generated source. The lfdname and module-name must correspond to information specified in the auto report U specification.

OUTSRC=(RES, lblname, lfdname, module-name)

Specifies the generated source located on the SYSRES device with the file identifier (lblname), logical file descriptor (lfdname), and module name of the generated source. The lfdname and module-name must correspond to information specified in the auto report U specification.

If omitted, there is no output file definition for the cataloged source file option.

#### Keyword Parameter LST:

Specifies the format of the compiler listing. The options are:

LST=K

Do not print error flags for sequence errors.

LST=M

Do not print the source language statements and error messages.

LST=N

Do not print any listings.

LST=S

Do not print the main storage map.

If omitted, the complete compiler listing is printed.

↓

**Keyword Parameter SCR1:**

Specifies the volume serial number of the work file labeled SCR1.

**SCR1=vol-ser-no**

The volume serial number.

If omitted, the default is RES.

**Keyword Parameter SCR2:**

Specifies the volume serial number of the work file labeled SCR2.

**SCR2=vol-ser-no**

The volume serial number.

If omitted, the default is RUN.

**Keyword Parameter EMB:**

Specifies whether or not embedded linkage editor control statements are to be generated in the RPG II object module.

**EMB=NO**

Do not generate embedded linkage editor control statements in the RPG II object module. No overlay structure is generated for the load table or dump table file.

If omitted, the default is YES.

**Keyword Parameter MOD:**

Specifies that the program is to be compiled in the IBM System/3-System/34 mode or that if the program is to be compiled in one of the other compilation modes, IRAM is to be used to process all disk files in a Series 90 environment and MIRAM is to be used in a System 80 environment.

**MOD=3**

The program is to be compiled in the IBM System/3-System/34 mode. When this mode is specified, IRAM or MIRAM will be used to process all disk files. Also, the logical file definition (// LFD) for printer files is changed to PRNTR, PRNTR1...PRNTRn and the control reader (CTRLDR) is used for card input even though the data management reader (READER) is specified.

**MOD=4**

Same as MOD=3 except that printer files are generated with the same names as used in the program and reader files use the data management reader (READER).

**MOD=5**

The program is to be compiled in the IBM System/3-System/34 mode with native mode data management accessing the disk files.

↑

**MOD=IRAM**

IRAM is to be used to process all disk files.

If omitted, the program is compiled in OS/3 native mode.

**Keyword Parameter SKIP:**

Specifies type of skipping used for printer output, generating a PARAM SKIP job control statement.

**SKIP=C**

SKIP=C causes auto report to generate SKIP to channel for the printer output file. If this option is not used, auto report generates SKIP to line numbers for the printer output file.

If omitted, you get spacing based on line numbers.

**Keyword Parameter COPYn:**

Specifies the input source file definition for /COPY statements within auto report source input. The number (n) of copies corresponds to the number of unique files used to copy.

**COPYn=(vol-ser-no, lblname, lfdname)**

Specifies the volume serial number, the file identifier (lblname) and the logical file descriptor (lfdname) where the copy source is located.

**COPYn=(RES, lblname, lfdname)**

Specifies that the copy source is located on the SYSRES device with the file identifier (lblname) and logical file descriptor (lfdname) of your own specification.

**COPY=(RUN, lblname, lfdname)**

Specifies that the copy source is located on the job's \$Y\$RUN file, with the file identifier (lblname) and logical file descriptor (lfdname) of your own specification.

If omitted, there is no input source file definition for /COPY statements.

**Keyword Parameter ALTLOD:**

Specifies the alternate load library containing the auto report product (AUTO#).

**ALTLOD=(vol-ser-no, lblname)**

Specifies the volume serial number (vol-ser-no) and the file identifier where the auto report (AUTO#) load module resides.



`ALTLOD=RES, l b l name )`

Specifies that the alternate load library is located on the job's SYSRES device, in the file identified by the file identifier.

`ALTLOD=(RUN, l b l name )`

Specifies that the alternate load library is located on the job's \$Y\$RUN file with the file identifier specified by the user.

`ALTLOD=( *, l b l name )`

Specifies that the alternate load library is located on a catalog file identified by the file identifier.

If omitted, the AUTO# load module is located in the \$Y\$LOD file on the SYSRES device.

#### Keyword Parameter ERRFIL:

Specifies that error diagnostic messages are written to a file that is accessed by the error file processor. When you specify this parameter, error records are created for every error note generated by auto report.

`ERRFIL=(vol-ser-no, l b l name , module - name )`

vol-ser-no specifies the volume serial number of the file. l b l name specifies the file identifier (name of the file that the module is placed into). module-name is the name of the module that is referenced by the error file processor.

If omitted, the error file is not created.

#### Keyword Parameter PROGID:

Specifies that auto report does not generate a program identification in columns 75 through 80 of the generated source.

`PROGID=N`

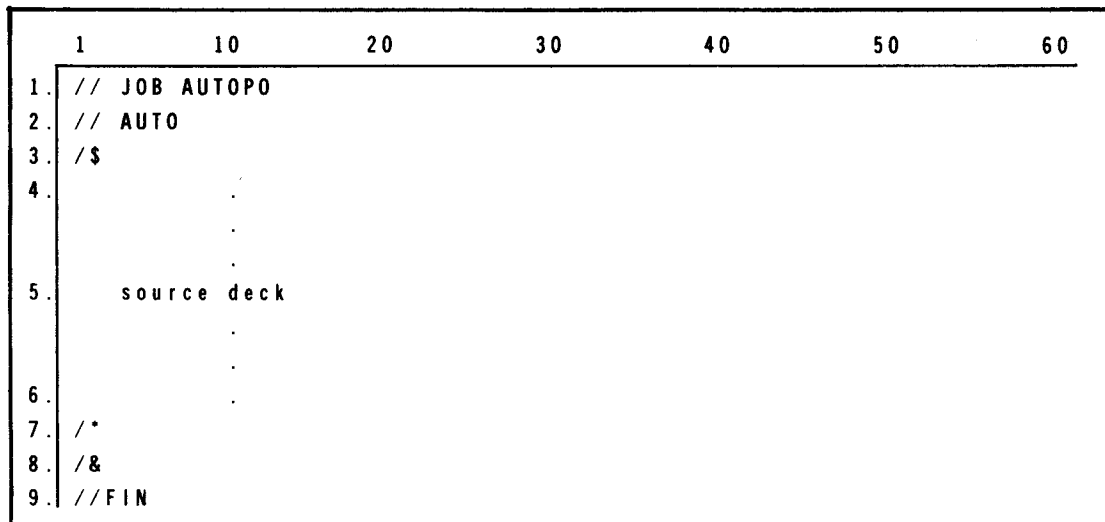
No program identification is generated.

If omitted, the program identification on the control specification is generated in all source output.

#### 18.2.2.1. Using the AUTO Job Control Procedure Call Statement

Figure 18-9 shows how you can use the AUTO job control procedure call to run RPG II auto report. In this example, the following conditions exist:

- The source language statements are read from the source deck in the job control stream.
- A listing is printed that contains original source, generated source, and error notes.
- The generated source module is stored in the job's \$Y\$SRC file.



```
1. // JOB AUTOPO
2. // AUTO
3. /$
4. .
5. source deck
6. .
7. /*
8. /&
9. //FIN
```

Figure 18-9. Using the AUTO Job Control Procedure Call Statement with Standard Default Options

In Figure 18-9, line 1 indicates the name of the job is AUTOPO.

Line 2 indicates that the procedure AUTO is being called.

Line 3 indicates start-of-data.

Lines 4 through 6 represent the source deck that contains the source language statements for the program to be compiled.

Line 7 indicates the end-of-data.

Line 8 indicates the end-of-job.

Line 9 causes the card reader (job control stream reader) to stop reading cards.

Figure 18-10 shows the job control statements that are generated by the entries in Figure 18-9.



1	10	20	30	40	50	60
1 .	//	JOB	AUTOPO			
2 .	//	DVC;20	//	LFD	PRNTR	
3 .	//	DVC	RES			
4 .	//	EXT	ST,C,3,CYL,1			
5 .	//	LBL	\$SCR1			
6 .	//	LFD	\$SCR1			
7 .	//	DVC	RES			
8 .	//	LBL	\$Y\$SRC			
9 .	//	LFD;F1				
10 .	//	EXEC	AUTO#,\$Y\$L0D			
11 .	/	\$				
12 .	.					
13 .		source	deck			
14 .	.					
15 .	/*					
16 .	/&					
17 .	//	FIN				

Figure 18-10. Job Control Statements Generated by the AUTO Procedure Call Statement with Standard Default Options

In Figure 18-10, line 1 indicates that the job name is AUTOPO.

Line 2 indicates the default logical unit number and LFD name of the printer.

Lines 3 through 6 indicate that the first work file needed for compilation is, by default, on the SYSRES device, has both a file identifier and LFD name of #SCR1, and uses the system access technique; and that the allocation is contiguous, with three cylinders allocated for the secondary increment and one cylinder allocated for the first extent.

Lines 7 through 9 indicate that the generated source is output to the system pack (DVC RES) to file \$Y\$SRC (LBL \$Y\$SRC).

Line 10 loads and executes auto report from \$Y\$L0D.

Line 11 indicates the start-of-data.

Lines 12 through 14 represent the source deck that contains the source language statements for the program to be compiled.

Line 15 indicates end-of-data.

Line 16 indicates end-of-job.

Line 17 causes the card reader (job control stream reader) to stop reading cards.

### 18.2.2.2. Using the AUTO Job Control Procedure Call that Defines Keyword Parameters

Figure 18-11 shows how you can use the keyword parameters in a job control procedure call statement to specify other than the standard default options. In this example, assume that the source is to be input from a permanent file; you want to copy several source statements from a copy file; and the source is to be output to a permanent file.

	1	10	20	30	40	72
1.	//	JOB	AUTOPO			
2.	//	PROGNM	AUTO	IN=(DSC1,AUTSRC),		X
3.	//	1		COPY1=(DSC1,CPYSRC,CPYIN1),		X
4.	//	2		OUTSRC=(DSC1,OUTSRC,AUTOUT)		
5.	/	&				
6.	//	FIN				

Figure 18-11. Using the AUTO Job Control Procedure Call Statement with Nonstandard Options

In Figure 18-11, line 1 indicates that the job name is AUTOPO.

Line 2 indicates that the procedure AUTO is being called, with source located on volume (VOL) DSC1 in file (LBL) AUTSRC. The element (program) name is the tag PROGNM.

Line 3 indicates that copy statement will obtain its input from volume (VOL) DSC1 having file (LBL) CPYSRC. The logical file descriptor (LBL) is CPYIN1.

Line 4 indicates source is to be output to volume DSC1 in file (LBL) OUTSRC and logical file descriptor of AUTOUT. This information should correspond to information given in columns 8-24 of the auto report (U) specifications form. See 19.5.1.2.

Line 5 indicates the end-of-job.

Line 6 causes the card reader (job control stream reader) to stop reading cards.

Figure 18-12 shows the job control statements that are generated by the entries.

1	10	20	30	40	50	60
1.	//	JOB	AUTOPO			
2.	//	DVC	20	//	LFD	PRNTR
3.	//	DVC	RES			
4.	//	EXT	ST,C,3,CYL,1			
5.	//	LBL	\$SCR1			
6.	//	LFD	\$SCR1			
7.	//	DVC	50	//	VOL	DSC1
8.	//	LBL	OUTSRC			
9.	//	LFD	AUTOUT			
10.	//	DVC	50	//	VOL	DSC1
11.	//	LBL	AUTSRC			
12.	//	LFD	INFILE			
13.	//	DVC	50	//	VOL	DSC1
14.	//	LBL	CPYSRC			
15.	//	LFD	CPYIN1			
16.	//	DVC	RES			
17.	//	LBL	\$Y\$SRC			
18.	//	LFD	F1			
19.	//	EXEC	AUTO#,\$Y\$LOD			
20.	//	PARAM	IN=PROGNM,INFILE			
21.	//	/&				
22.	//	FIN				

Figure 18-12. Job Control Statements Generated by the AUTO Procedure Call Statement with Nonstandard Options

In Figure 18-12, line 1 indicates that the job name is AUTOPO.

Line 2 indicates the default logical unit number and LFD name of the printer.

Lines 3 through 6 indicate that the first work file needed for compilation is, by default, on the SYSRES device; has both a file identifier and LFD name of \$SCR1; uses the system access technique; and that the allocation is contiguous, with three cylinders allocated for the secondary increment and one cylinder allocated for the first extent.

Lines 7 through 9 indicate the DVC-LFD sequence for the permanent file where the output source will be written. The volume (VOL) is DSC1, the file (LBL) is OUTSRC, and the LFD name is AUTOUT.

Lines 10 through 12 indicate the DVC-LFD sequence for the file containing the input source. The volume (VOL) is DSC1, the file (LBL) is AUTSRC, and the LFD name is INFILE.

Lines 13 through 15 indicate the DVC-LFD sequence for the file containing statements to be copied. The volume (VOL) is DSC1, the file (LBL) is CPYSRC, the file (LBL) is CPYSRC, and the LFD name is CPYIN1.



Lines 16 through 18 indicate that the generated source is also output to the system pack (DVC RES) to file \$Y\$SRC (LBL \$Y\$SRC).

Line 19 loads and executes auto report from \$Y\$LOD.

Line 20 indicates that input source is in the file having an LFD name of INFILE with element name PROGNM.

Line 21 indicates the end-of-job.

Line 22 causes the card reader (job control stream reader) to stop reading cards.

### 18.2.2.3. Using the ATRPG Job Control Procedure Call Statement

Figure 18-13 shows how you can use the ATRPG job control procedure call to run auto report and to do an RPG II compilation of the generated source.

	1	10	20	30	40	50	60
1.	//	JOB	AUTCMP				
2.	//	ATRPG					
3.	/\$						
4.		.					
		.					
5.		source	deck				
		.					
6.		.					
7.	/*						
8.	/&						
9.	//	FIN					

Figure 18-13. Using the ATRPG Job Control Procedure Call Statement

In Figure 18-13, line 1 indicates the name of the job is AUTCMP.

Line 2 indicates that the procedure ATRPG is being called.

Line 3 indicates start-of-data.

Lines 4 through 6 represent the source deck that contains the source language statements for the program to be compiled.

Line 7 indicates the end-of-data.

Line 8 indicates the end-of-job.

Line 9 causes the card reader (job control stream reader) to stop reading cards.

Figure 18-14 shows the job control statements that are generated by the entries in Figure 18-13.

	1	10	20	30	40	50	60
1.	//	JOB	AUTCMP				
2.	//	DVC	20	//	LFD	PRNTR	
3.	//	DVC	RES				
4.	//	EXT	ST,C,3,CYL,1				
5.	//	LBL	\$SCR1				
6.	//	LFD	\$SCR1				
7.	//	DVC	RES				
8.	//	LBL	\$Y\$SRC				
9.	//	LFD	F1				
10.	//	EXEC	AUTO#,\$Y\$LOD				
11.	/\$						
12.	.						
13.	.	source	deck				
14.	.						
15.	/*						
16.	//	DVC	RES				
17.	//	EXT	ST,C,3,CYL,1				
18.	//	LBL	\$SCR1	//	LFD	\$SCR1	
19.	//	DVC	RUN				
20.	//	EXT	ST,C,3,CYL,1				
21.	//	LBL	\$SCR2	//	LFD	\$SCR2	
22.	//	EXEC	RPGII				
23.	//	PARAM	IN=RPGSRC/\$Y\$SRC				
24.	/&						
25.	//	FIN					

Figure 18-14. Job Control Statements Generated by the AUTRPG Procedure Call Statement

In Figure 18-14, line 1 indicates the name of the job is AUTCMP.

Line 2 indicates the default logical unit number and LFD name of the printer.

Lines 3 through 6 indicate that the first work file needed for compilation is, by default, on the SYSRES device; has both a file identifier and LFD name of \$SCR1; uses the system access technique; and that the allocation is contiguous, with three cylinders allocated for the secondary increment and one cylinder allocated for the first extent.

Lines 7 through 9 indicate the DVC-LFD sequence to output generated source to the system pack (DVC RES) to file \$Y\$SRC (LBL \$Y\$SRC).

Line 10 loads and executes auto report from \$Y\$LOD.

Line 11 indicates start-of-data.

Lines 12 through 14 represent the source deck that contains the source language statements for the program to be compiled.

Line 15 indicates the end-of-data.

Lines 16 through 18 indicate the job control language for the first work file needed for RPG II compilation.

Lines 19 through 21 indicate the second work file needed for RPG II compilation.

Line 22 loads the RPG II compiler from \$Y\$LOD.

Line 23 indicates the source input will be from \$Y\$SRC with element name of RPGSRC.

Line 24 indicates end-of-job.

Line 25 causes the card reader (job control stream reader) to stop reading cards.

### 18.2.3. Using the EXEC and PARAM Job Control Statements

As mentioned before, instead of using the job control procedure call statements (RPG, RPGL, RPGLG, AUTO, AUTRPG, AUTRPGL, and AUTRPGLG) to compile, link-edit, and execute your program, you can write all the job control statements for each job step. This can be time-consuming because of the effort required to write the individual statements. With this method, you must use the EXEC and PARAM job control statements. The EXEC statement identifies the name of the load module and can optionally specify the library that contains the load module as well as the task-switching priority. The PARAM statement introduces processing options; it must immediately follow the EXEC statement.

Format:

```
//[symbol] PARAM [IN=program name/ldname]
    [ ,OUT={ldname}
        { (N)
          (I) } ]
    [ ,LST={ (K)
              (M)
              (N)
              (S) } ]
    [ ,MOD={ (3)
              (4)
              (5)
              (IRAM) } ]
    [ ,COL=7 ]
    [ ,MIRAM={ldname1, . . . , ldname20}
              { ALL } ]
    [ ,SKIP=C ]
    [ ,ERRFIL=module-name/ldname ]
    [ ,CONSOLE=ldname ]
```

## Label:

`symbol`

Specifies the 1- to 6-character label.

## Operation:

`PARAM`

A job control statement used to introduce processing options. The operands are the variable information you want to introduce into the job.

## Keyword Parameter IN:

Specifies the program name and lfdname when the source code is input from a disk file.

`IN=program name/lfdname`

Identifies the source program and the disk file it resides on.

If omitted, the source input is in the form of embedded data cards (/ \$, source deck, /\*) in the job control stream.

## Keyword Parameter OUT:

Specifies the output file definition. The options are:

`OUT=filename`

Specifies that the file name contains the generated object module. The default file name is \$Y\$RUN.

`OUT=(N)`

Specifies that an object module is not to be produced.

`OUT=(I)`

Do not generate embedded linkage editor control statements in the RPG II object module. No overlay structure is generated for the load table or dump table file.

If omitted, the object module is located on the job's \$Y\$RUN file.

## Keyword Parameter LST:

Specifies the format of the compiler listing. The options are:

`LST=K`

Do not print error flags for sequence errors.

`LST=M`

Do not print the source language statements and error messages.

`LST=N`

Do not print any listings.

`LST=S`

Do not print the main storage map.

If omitted, the complete compiler listing is printed.

#### Keyword Parameter MOD:

Specifies that the program is to be compiled in the IBM System/3-System/34 mode or that, if the program is to be compiled in one of the other compilation modes, IRAM is to be used to process all disk files in a Series 90 environment and MIRAM is to be used in a System 80 environment.

##### MOD=3

The program is to be compiled in the IBM System/3-System/34 mode. When this mode is specified, IRAM or MIRAM will be used to process all disk files. Also, the logical file definition (// LFD) for printer files is changed to PRNTR, PRNTR1...PRNTRn and the control reader (CTLRDR) is used for card input even though the data management reader (READER) is specified.

##### MOD=4

Same as MOD=3 except that the printer files are generated with the same names as used in the program and reader files use the data management reader (READER).

##### MOD=5

The program is to be compiled in the IBM System/3-System/34 mode with the ISAM, SAM, or DAM processors accessing the disk files. ←

##### MOD=IRAM

IRAM is to be used to process all disk files.

If omitted, the program is compiled in OS/3 native mode.

#### Keyword Parameter COL:

Specifies that compile time tables can begin in position 7 rather than in position 1.

##### COL=7

Enables the user to carry sequence numbers in columns 1-5 for each statement in the RPG source program. The OS/3 librarian can then be used to maintain disk file source for RPG programs that contain compile time tables.

If omitted, compile-time tables begin in position 1.

#### Keyword Parameter MIRAM:

When operating under the mixed Series 90 environment, this parameter designates which disk files use MIRAM. Those disk files not mentioned use Series 90 data management only. All other files, such as card or tape, use System 80 data management and need not be specified here.

##### MIRAM=lfname1, . . . , lfname20

The file name is a disk file using MIRAM.

##### MIRAM=ALL

All disk files use MIRAM.

**Keyword Parameter SKIP:**

Specifies that auto report will skip the printer to channel numbers (vertical format buffer) for the printer output specifications.

`SKIP=C`

Skips the printer to channel numbers for the printer output specifications.

If omitted, auto report generates skips to line numbers.

**Keyword Parameter ERRFIL:**

Specifies that error diagnostic messages are written to a file that is accessed by the error file processor. When you specify this parameter, along with the necessary `// DVC...// LFD` statements, error records are created for every error note generated by the compiler.

`ERRFIL=module-name/lfdname`

The module name is the name assigned to the error by the file element being created by the compiler. The module name may be from 1 to 8 characters in length and it doesn't need to match or be related to the RPG II source or object module name. The lfdname is the logical file name of the error log file and can be from 1 to 8 characters in length.

If omitted, the error file is not created.

**Keyword Parameter CONSOLE:**

Specifies that the file is a CONSOLE (interactive data entry) file and not a system console file.

`CONSOLE=lfdname`

The lfdname is the name of the CONSOLE file.

If omitted, the file is a system console file and not an interactive data entry file.

**Keyword Parameter PROGID:**

Specifies that auto report does not generate a program identification in columns 75 through 80 of the generated source.

`PROGID=N`

No program identification is generated.

If omitted, the program identification on the control specification is generated in all source output.

Figure 18-15 shows how you can use the EXEC and PARAM statements to compile, link-edit, and execute an RPG II program.

1	10	20	30	40	50	60
1.	//	JOB	RPGPARAM			
2.	//	DVC	20	//	LFD	PRNTR
3.	//	WORK1				
4.	//	WORK2				
5.	//	EXEC	RPGII			
6.	//	PARAM	LST=N			
7.	/\$					
8.	.					
9.	source	deck				
10.	.					
11.	/*					
12.	//	WORK1				
13.	//	EXEC	LNKEDT			
14.	/\$					
15.		LOADM	load-module-name			
16.		INCLUDE	program-name			
17.	/*					
18.	.					
19.	device	assignments				
20.	for	your	program			
21.	.					
22.	//	EXEC	load-module-name			
23.	/&					
24.	//	FIN				

Figure 18-15. Using the EXEC and PARAM Job Control Statements

In Figure 18-15, line 1 indicates the name of the job is RPGPARAM.

Line 2 indicates the default logical unit number and LFD name of the printer.

Lines 3 and 4 indicate temporary work files used to store intermediate processing results.

Line 5 loads the RPG II compiler from \$Y\$LOD.

Line 6 indicates no listings will be printed.

Line 7 indicates start-of-data.

Lines 8 through 10 represent the source deck that contains the source language statement to be compiled.

Line 11 indicates end-of-data.

Line 12 indicates a temporary work file used to store intermediate processing results.

Line 13 creates the load module.

Line 14 indicates start-of-data.

Line 15 assigns the load module name.

Line 16 informs the linkage editor which object modules will comprise the load module.

Line 17 indicates end-of-data.

Lines 18 through 21 represent the job control statements that assign the input/output devices and allocate the file space required to execute your program.

Line 22 executes the load module (your program).

Line 23 indicates end-of-job.

Line 24 causes the card reader (job control stream reader) to stop reading cards.

↓  
For job control used with auto report:

- If auto report retrieves the source program for input from a library file (MIRAM or SAT) on disk, use the PARAM IN job control statement to identify the source program (module name) and the disk file it resides on:

```
// PARAM IN=program name/lfdname
```

- If auto report retrieves the source program for input from an embedded data file in the job control stream, don't use the PARAM IN job control statement. Instead, embed the source program in the job control stream as a data file and delimit it with the /\$ and /\* job control statements:

```
/$
.
.   source program
.
/*
```

- Use the PARAM SKIP job control statement to control how auto report will generate the skips to channel numbers for the program printer file:

```
// PARAM SKIP=C
```

- Use the PARAM ERRFIL=module-name/lfdname job control statement to create an error file to be read by the error file processor. When you specify this parameter along with the necessary // DVC...// LFD statements, error file records are created for every diagnostic output by auto report. The module name is the name assigned to the element being created by auto report. The module name may be from one to eight characters in length. The lfdname is the logical file name of the error file and can be from one to eight characters in length.
- ↑



```
// PARAM ERRFIL=module-name/lfdname
```

- Use the product name of AUTO# for auto report on the EXEC job control statement:

```
// EXEC AUTO#
```

- Use one work file:

```
// WORK1
```

- Use a line printer device (with or without spooling) for auto report output listings:

```
// DVC 20 // LFD PRNTR
```

- Optionally, use a disk or diskette for library files that contain source and copy input to auto report, as well as source output from auto report.

The following job control stream in Figure 18-16 uses the EXEC and PARAM statements to execute auto report with embedded data, then compiles, link-edits, and executes RPG II using the RPGLG jproc:

1	10	20	30	40	50	60
1.	//	JOB	name,,D000,,D000			
2.	//	DVC 20 //	LFD PRNTR			
3.	//	WORK1				
4.	[optional	DVC...LFD	device assignment set for /COPY	statement files --		
	auto	report	input]			
5.	[optional	DVC...LFD	device assignment set for saved	source program --		
	auto	report	output]			
6.	//	EXEC	AUTO#			
7.	//	PARAM	SKIP=C			
8.	/\$					
9.	.	auto	report	source	input (on	cards)
10.	/*					
11.	[optional	DVC...LFD	device assignment sets for	object	program	
	execution	files]				
12.	//RPGSRC	RPGLG	IN=(RES)			
13.	/&					
14.	//	FIN				
15.	//	DATA	FILEID=file-identifier			
16.	.	data	cards			
17.	/*					
18.	//	FIN				

Figure 18-16. Using the EXEC and PARAM Job Control Statements with Auto Report (with Input on Cards)

↓  
In Figure 18-16, line 1 indicates the name of the job. You need a minimum of 50K bytes of main storage.

Line 2 indicates the printer device needed for auto report.

Line 3 indicates the work file needed for auto report.

Lines 4 and 5 indicate optional assignment sets needed for auto report input and output. When you don't designate a saved source file, the default is the RPGSRC program stored in \$Y\$SRC.

Line 6 executes auto report.

Line 7 indicates that auto report will generate skips to channel numbers for the printer file.

Line 8 indicates start-of-data.

Line 9 represents the source deck that contains the auto report specifications.

Line 10 indicates end-of-data.

Line 11 indicates optional assignment sets needed for object program execution files.

Line 12 indicates that the RPG II job control procedure RPGLG is being called. It compiles, links, and executes the RPGSRC program from \$Y\$SRC.

Line 13 indicates end-of-job.

Line 14 causes the card reader to stop reading cards.

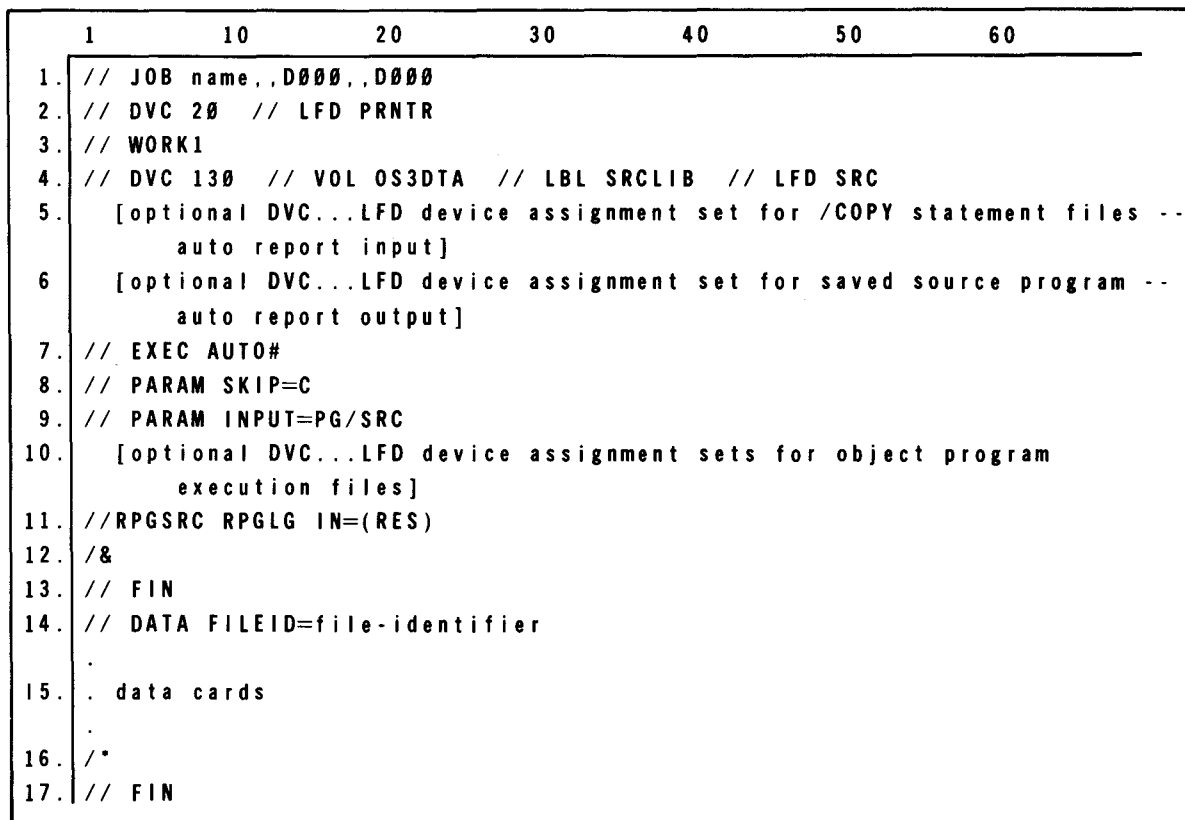
Line 15 indicates that the input card data will be spooled.

Line 16 represents the data cards.

Line 17 indicates end-of-data.

Line 18 causes the card reader to stop reading cards.

↑  
The following job control stream in Figure 18-17 uses the EXEC and PARAM statements to execute auto report with source input from diskette, then compiles, link-edits, and executes RPG II using the RPGLG jproc:



```
1. // JOB name,,D000,,D000
2. // DVC 20 // LFD PRNTR
3. // WORK1
4. // DVC 130 // VOL OS3DTA // LBL SRCLIB // LFD SRC
5. [optional DVC...LFD device assignment set for /COPY statement files --
   auto report input]
6. [optional DVC...LFD device assignment set for saved source program --
   auto report output]
7. // EXEC AUTO#
8. // PARAM SKIP=C
9. // PARAM INPUT=PG/SRC
10. [optional DVC...LFD device assignment sets for object program
    execution files]
11. //RPGSRC RPGLG IN=(RES)
12. /&
13. // FIN
14. // DATA FILEID=file-identifier
15. . data cards
16. /*
17. // FIN
```

Figure 18-17. Using the EXEC and PARAM Job Control Statements with Auto Report (with Input on Diskette)

In Figure 18-17, line 1 indicates the name of the job. You need a minimum of 50K bytes of main storage.

Line 2 indicates the printer device needed for auto report.

Line 3 indicates the work file needed for auto report.

Line 4 indicates source input coming from diskette.

Lines 5 and 6 indicate optional assignment sets needed for auto report input and output.

Line 7 executes auto report.

Line 8 indicates that auto report will generate skips to channel numbers for the printer file.



Line 9 indicates source LFD is SRC and the element or program name is PG.

Line 10 indicates optional assignment sets needed for object program execution files.

Line 11 indicates that the RPG II job control procedure RPGLG is being called. It compiles, links, and executes the RPGSRC program from \$Y\$SRC.

Line 12 indicates end-of-job.

Line 13 causes the card reader to stop reading cards.

Line 14 indicates that the input card data will be spooled.

Line 15 represents the data cards.

Line 16 indicates end-of-data.

Line 17 causes the card reader to stop reading cards.

**NOTE:**

*For information about auto report job control procedure call statements, see 18.2.2.*

#### 18.2.4. Input Deck Sequence

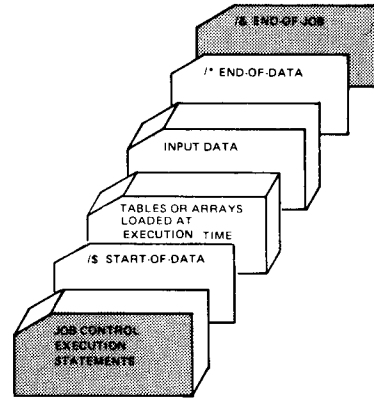
↓  
↑  
Figure 18-18 shows the input sequence of job control statements, RPG II source statement specifications, and source input data. You must arrange your deck in the order shown. Job streams containing calls on RPG II or auto report job control procedures have references to volume serial numbers that are assigned DVC numbers from 50 to 59.

### 18.3. MAIN STORAGE REQUIREMENTS

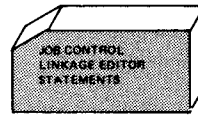
The RPG II compiler requires a minimum of 24,064 (hexadecimal 5E00) bytes of main storage. It is recommended, however, that you use 32,768 (hexadecimal 8000) bytes for compilation. If you use the minimum amount of main storage, there is the possibility that your program will not be compiled when using tables or arrays because the symbol table requires additional space. If there is insufficient space for the symbol table, the compilation takes longer.

If your program was originally compiled with the error analysis dump option requested (a D in column 8 of the control card specifications form), to save main storage you should compile your program without the error analysis dump option (column 8 is left blank on the control card specifications form). By doing this, you will save considerable main storage space during the execution of your program; that is, if the error analysis dump option request remains in your program, an additional 5632 (hexadecimal 1600) bytes of main storage will be required.

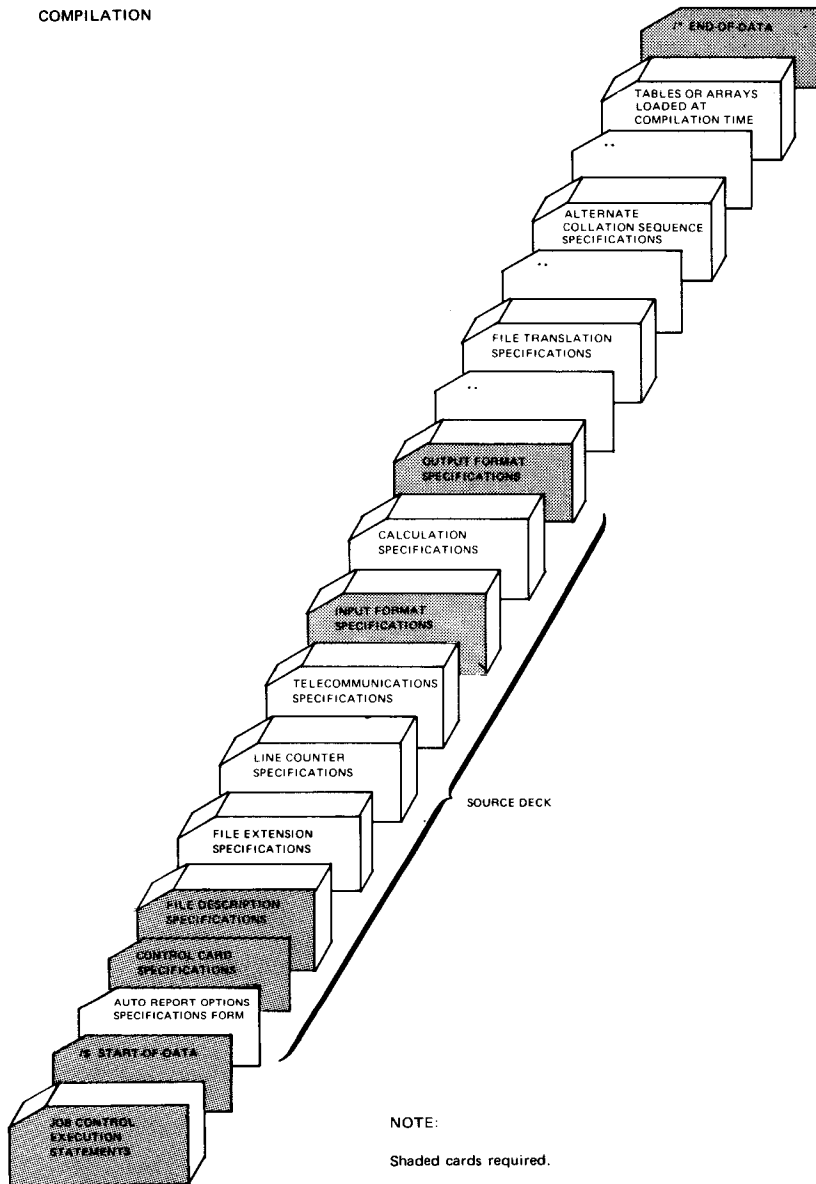
EXECUTION



LINK EDITING



COMPILATION



NOTE:

Shaded cards required.

Figure 18-18. Deck Arrangement for Compiling, Link Editing, and Execution

## 18.4. DEVICE INDEPENDENCE

In a System 80 environment, you can change from one device to another at program execution time without recompiling the RPG II program. The only factor that prohibits changing devices is the record size. For example, suppose that on your file description specifications form you choose a card reader to supply input. Later, you decide on a disk for input. To do this, merely change the device unit number in your job control stream from 30 (a card reader) to 50 (a disk) at program execution time. You need not change your file specifications form or recompile the program. You cannot change from a disk to a card reader, however, since the record size on a disk is probably more than 80 bytes, the limit of a card reader.

If your computer environment is partially System 80 and partially Series 90, you can still attain device independence. However, in addition to differences in record size, conflicting record formats also prohibit device independence. For example, if you wish to change from a disk, which has records organized indexed sequentially, to a tape, you would have problems because a tape receives records sequentially. Device independence is not available in a strictly Series 90 environment.

**PART 6. AUTO REPORT**





## 19. Auto Report

### 19.1. GENERAL

Auto report is a simple way for you to produce printed reports that contain page headings, column headings centered over fields, and accumulated totals. It is a stand-alone product that operates before the execution of the RPG II compiler.

The auto report precompiler creates an auto report source module and then the RPG II compiler uses that source module to create an RPG II object module. Auto report statements are entered on the output format specifications form. They replace other output specifications.

You specify the simple and easy-to-use auto report statements along with your standard RPG II source program. These auto report statements generate standard RPG II specifications and can copy standard RPG II specifications from a library file. The generated or copied specifications are combined with the RPG II specifications for your source program by auto report to produce a final RPG II source program. The RPG II compiler then executes this complete source program to produce a well-formatted report.

Even if you are an inexperienced programmer, you can easily code a program to print a simple report. If you are an experienced programmer, you can code programs faster. Auto report frees you from coding the same specifications in different programs, planning the format of reports, and coding specifications that accumulate totals for numeric fields.

In addition, auto report reduces the time it takes you to plan and code RPG II programs. Since auto report is so easy to use, you make fewer errors, which reduces debugging time.

Auto report operates in the minimum processing environment. It needs 50K bytes for execution.

There are five auto report specifications:

1. H-\*AUTO

This specification gives you a simple way to print page headings.

You enter H-\*AUTO specifications on the output format (O) specifications form.

## 2. D-\*AUTO

This specification gives you a simple way to print a detail report that has a line printed for each individual record that is read. A detail report can also accumulate and print subtotals and a final total for numeric fields.

You enter D-\*AUTO specifications on the output format (O) specifications form.

## 3. T-\*AUTO

This specification gives you a simple way to print a total report that has lines printed only for totals. There are no detail lines. In other words, data is summarized for a group of input records and only totals are printed on the report. You can specify subtotals with a final total or a final total only. A line is not printed for each individual record that is read but only after a complete control group is read.

You enter T-AUTO specifications on the output format (O) specifications form.

## 4. Auto Report Options Specifications Form (U in Column 6)

This form gives you a simple way to obtain these auto report options:

- Cataloging the generated source program into a library file on disk
- Suppressing the generated date and page number so they aren't printed
- Suppressing asterisks so they aren't printed next to the generated totals
- Suppressing an auto report source listing that contains specifications copied from the library file, the generated specifications, the standard RPG II specifications, and error messages

You enter the auto report options on the auto report options specifications form (U in column 6). When you use this specifications forms it must be the first specification in the program.

## 5. /COPY

This statement gives you a simple way to copy cataloged RPG II specifications from a library file on disk and insert them into your RPG II source program. You can also use /COPY modifier statements to modify file description and input field specifications as they are copied from the library file.

You can enter the /COPY statement on any specifications form except the control card (H) or auto report options (U) specifications form.

You enter the /COPY modifier statements on either the file description (F) or input format (I) specifications form.

Figure 19-1 summarizes the auto report function.

Auto report provides the following facilities:

- Retrieval of source input from a disk library file or from the job control stream
- Storage of the generated source program module in the system resident file \$Y\$SRC (with the module name RPGSRC) and optionally in a specified disk library file (with the program name as the module name)
- Printing of requested listings of the original source program, the generated source program, and error messages
- Retrieval of input from one or more disk files by the /COPY statement

Auto report runs in background (nonconversational) mode with initiation and termination messages displayed to the originating workstation and to the job log.

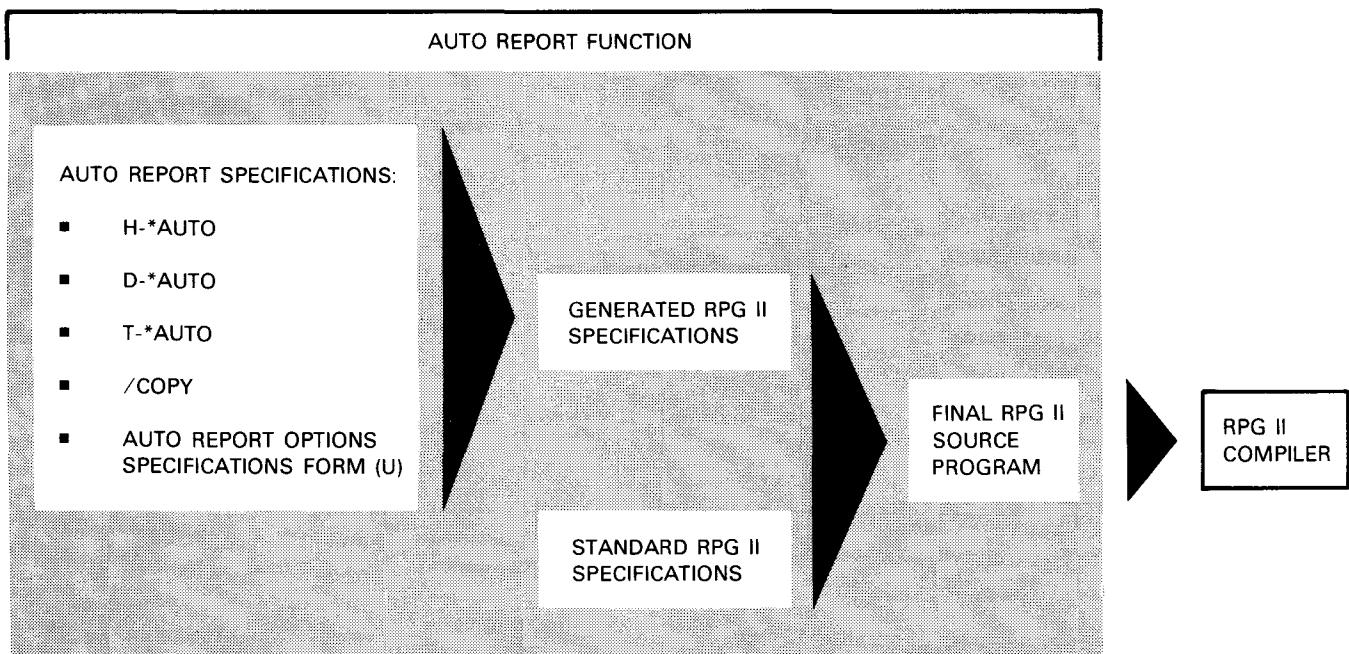


Figure 19-1. Auto Report Function

## 19.2. H-\*AUTO PAGE HEADINGS

You use the H-\*AUTO specification on the output specifications form to print a page heading at the top of every page of a report. This page heading can contain a date, page number, and title.

You can use up to five H-\*AUTO page headings if you want a page heading that has more than one line. If you specify both standard RPG II heading lines and H-\*AUTO page headings, they are printed in the order you use them in the output specifications. You can specify H-\*AUTO page headings for only one printer file per program.

The standard RPG II output specification form is used for two types of entries: output file identification (columns 7 through 31) and field description (columns 23 through 70). H-\*AUTO specifications have the same two types of entries, but their use is not the same.

### 19.2.1. Output File Identification and Control Entries (Columns 7 through 37)

You specify an H in column 15 and \*AUTO in columns 32 through 36 for an auto report output file identification. This indicates that the report contains a page heading. You can follow it with one or more field descriptions that specify a title or other information for the report. You use each output file identification to specify a separate page heading line.

Figure 19-2 illustrates the H-\*AUTO output file identification.

COLS:	6	7-13	15	16	17-18	19-22	23-31	32-37	38-70
	O	FILENAME	H	BLANK	BLANK OR SPACE	BLANK OR SKIP	BLANK OR INDICATORS	*AUTO	BLANK

Figure 19-2. H-\*AUTO Output File Identification

#### 19.2.1.1. File Name (Columns 7 through 13)

You use this field to specify the name of the printer file that prints the report.

Enter the name of the printer file. You only need to specify a file name on the first output file identification for a file. The file name can be up to seven characters in length. See 5.2.1 for more explanation.

**NOTE:**

*The file name can be eight characters long.*

#### 19.2.1.2. Type (Column 15)

You use this field to indicate that you want a report that contains a page heading.

Enter an H in column 15 and \*AUTO in columns 32 through 36. You can use up to five H-\*AUTO specifications for one output file.

The first H-\*AUTO specification you specify generates a page heading consisting of a date, a page number, and, optionally, a title:

1. Generated date

The date generated by H-\*AUTO is left-justified in print positions 1 through 8. The format of the date is determined by the date option on the control specifications form and UDATE on the output specifications form.

If you want to suppress the date on the first page heading, enter an N in column 27 on the auto report options specifications form.

2. Generated title (optional)

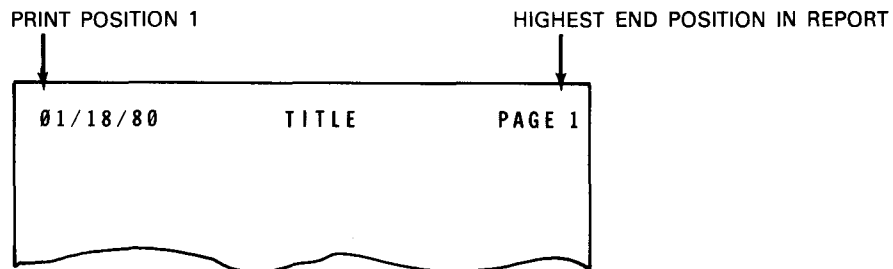
For an explanation of the title, see 19.2.2.1.1.

3. Generated page number

The page number generated by H-\*AUTO is right-justified and follows the word PAGE. It is four digits in length, zero-suppressed, and printed with an end position that is the same as the highest end position of the longest line in the report.

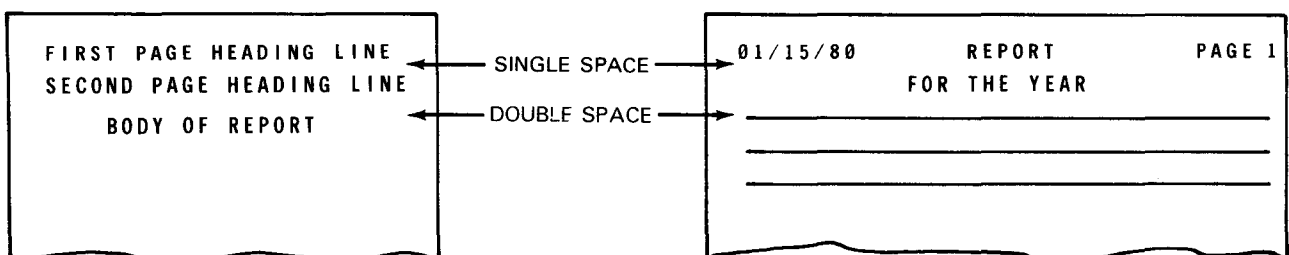
For page numbering, auto report uses an unused PAGE field (PAGE, PAGE1-PAGE7). If you have already used all the PAGE fields in the program, auto report doesn't number the pages.

If you want to suppress the page number on the first page heading, enter an N in column 27 of the auto report option specifications form.



19.2.1.3. Space (Columns 17 and 18)

You use this field to control the spacing of page headings. When you want auto report to automatically use one blank line after each single page heading line, leave these columns blank. When you use more than one page heading line, auto report uses a single space after each page heading line except the last, which is followed by a double space:







If you specified an overflow indicator on the file description specification for the printer file, the generated page heading is conditioned by that indicator. If you didn't specify an overflow indicator, auto report defines an unused overflow indicator for the printer file on the file description specification and the line is conditioned by that indicator.

If you specified the name of a field, table, or array in columns 32 through 37 of the field description, auto report generates an N1P output indicator and this conditions those fields. The N1P output indicator prevents the field, table, or array from being printed on the first page. If it were printed on the first page, the field would probably contain meaningless data because the first record hasn't been read. Auto report will not generate N1P if you enter the following special field names in columns 32 through 37 of the field description:

PAGE

PAGE1

PAGE2

PAGE3

PAGE4

PAGE5

PAGE6

PAGE7

UPDATE

UDAY

UMONTH

UYEAR





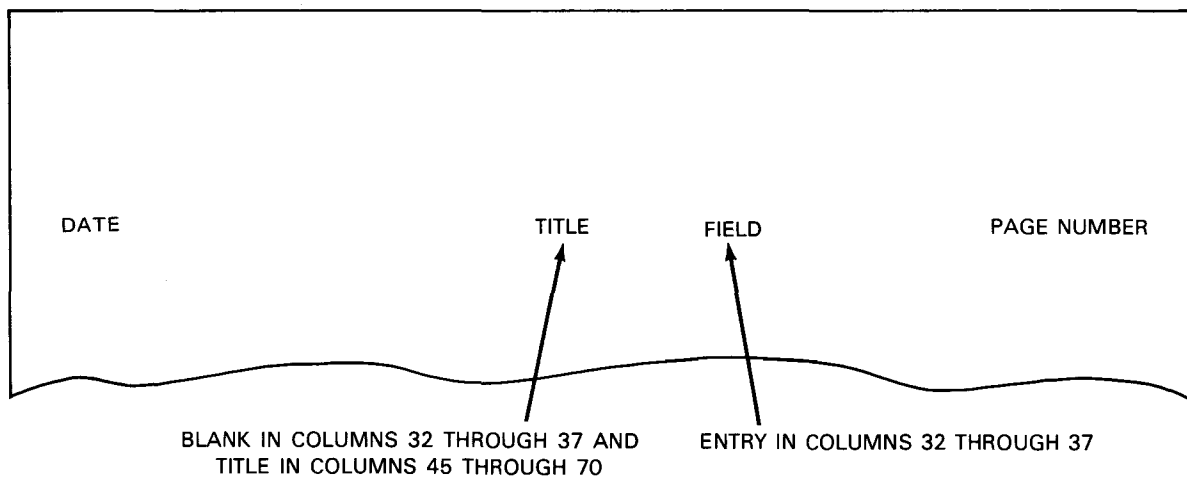


### 19.2.2. Field Description and Control Entries (Columns 32 through 70)

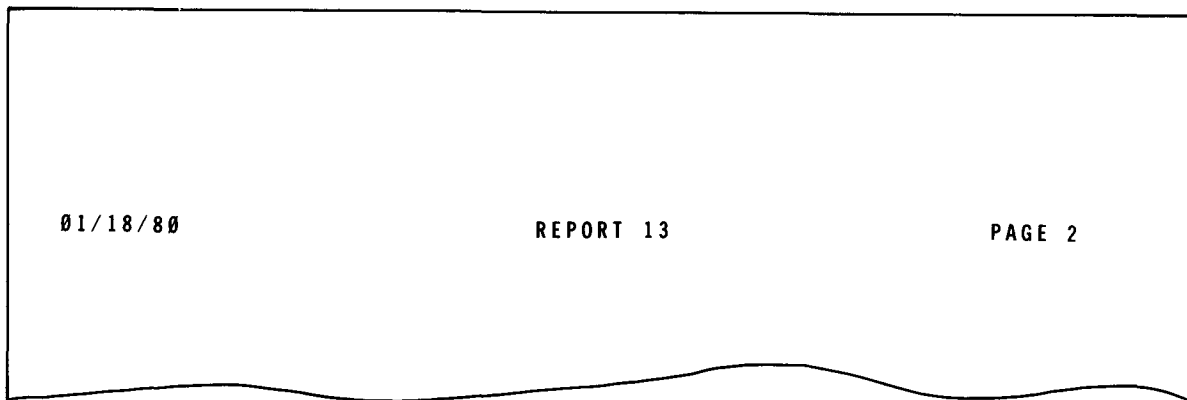
You can follow the H-\*AUTO output file identification with one or more field descriptions. Each of the following field descriptions allows you to format the page heading in the way you want:

- A field description that prints a title on the page heading (blanks in columns 32 through 37 and title in columns 45 through 70)
- A field description that prints a field on the page heading (field name in columns 32 through 37)

Figure 19-4 illustrates the functions of the H-\*AUTO field descriptions.



a. H-\*AUTO specifications



b. Example of printed report

Figure 19-4. H-\*AUTO Field Descriptions

### 19.2.2.1. Field Description that Prints a Title on the Page Heading (Blanks in Columns 32 through 37 and Title in Columns 45 through 70)

You use this field description to print a title (constant) on the page heading line.

Figure 19-5 illustrates the H-\*AUTO field description that has blanks in columns 32 through 37 and a title in columns 45 through 70.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	BLANK	BLANK	BLANK	BLANK	BLANK	TITLE

Figure 19-5. H-\*AUTO Field Description (Blanks in Columns 32 through 37 and Title in Columns 45 through 70)

#### 19.2.2.1.1. Title (Columns 45 through 70)

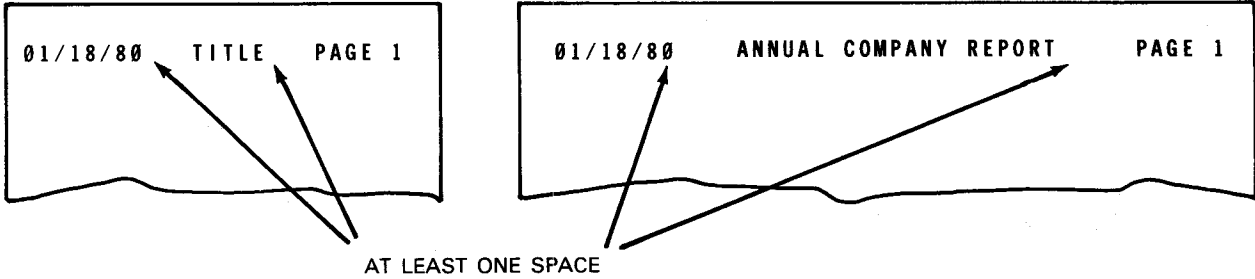
You use this field to print the title (constant) on the page heading.

→ When you want the page heading to contain only the date and page number but no title, leave this field description blank.

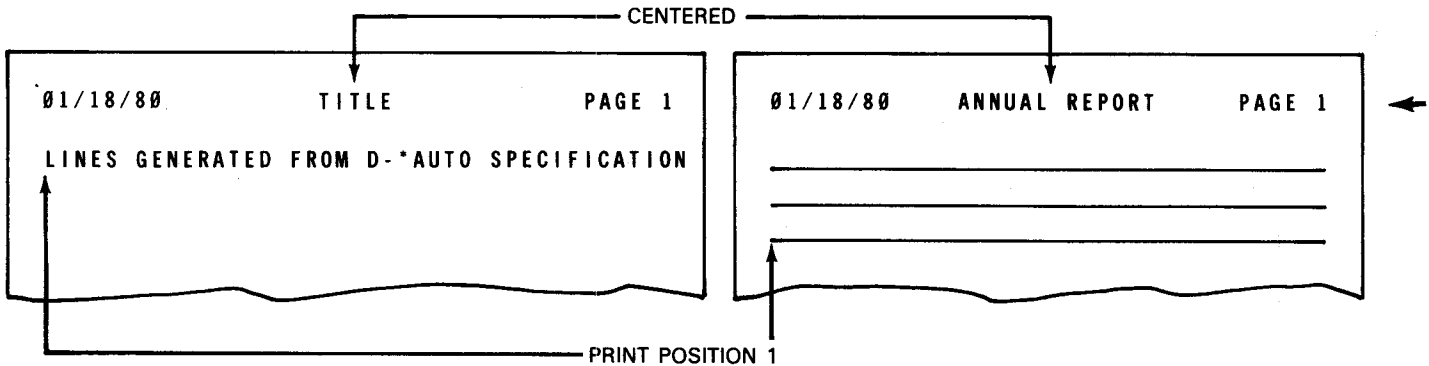
When you specify a title, you must leave all the other fields on the field description blank. You must enclose the title within apostrophes. No spaces are provided within a title - you must incorporate spaces within it to provide for additional spacing.

The placement of the title depends on which is longer, the page heading or the lines generated by D-\*AUTO or T-\*AUTO specifications.

When the page heading is longer than the lines generated by D-\*AUTO or T-\*AUTO, one blank space separates the date from the title and the title from the word PAGE:



If the page heading is shorter than the lines generated by D-\*AUTO or T-\*AUTO specifications, the D or T line begins in print position 1 and the title is centered above it:



If you specify more than one page heading line, the shorter page headings are centered on the longest page heading line.

If the page heading is longer than the record length you specified for the printer file, the characters on the right are truncated. There is no overflow line.

















### 19.3. D-\*AUTO DETAIL REPORTS

You use the D-\*AUTO specification on the output specifications form to print a report that can contain both detail lines and total lines. In other words, the report has detail lines printed for each individual record that is read and has total lines that accumulate numeric totals.

You can specify the D-\*AUTO specification alone or in combination with standard RPG II specifications. You can't use both a D-\*AUTO and T-\*AUTO specification in the same program. You can specify D-\*AUTO for only one printer file per program.

You can also use the D-\*AUTO specification to:

- Print up to three lines of column headings over a field
- Accumulate several levels of totals
- Print information next to a total
- Format the report

The standard RPG II output specifications form is used for two types of entries: output file identification (columns 7 through 31) and field description (columns 23 through 70). D-\*AUTO specifications have the same two types of entries, but their use is not the same.

#### 19.3.1. Output File Identification and Control Entries (Columns 7 through 37)

You must specify a D in column 15 and \*AUTO in columns 32 through 36 for an auto report output file identification. This indicates that the report can contain both detail lines and total lines. You must follow output file identification with at least one field description that describes when, where, or how the lines are printed.

Figure 19-10 illustrates the D-\*AUTO output file identification.

COLS:	6	7-13	15	16	17-18	19-22	23-31	32-37	38-70
	O	FILENAME	D	BLANK OR FETCH OVERFLOW	BLANK OR SPACE	BLANK OR SKIP	BLANK OR INDICATORS	*AUTO	BLANK

Figure 19-10. D-\*AUTO Output File Identification

### 19.3.1.1. File Name (Columns 7 through 13)

You use this field to specify the name of the printer file that prints the report.

Enter the name of the printer file. The name you specify for this file must be the same name you specified on the H-\*AUTO page heading if you used one. You only need to specify a file name on the first output file identification for a file.

The file name can be up to seven characters in length. See 5.2.1 for more explanation.

**NOTE:**

→ *The file name can be eight characters long.*

### 19.3.1.2. Type (Column 15)

You use this field to indicate that you want a report that can contain both detail lines and total lines.

Enter a D in this column and \*AUTO in columns 32 through 36. Remember that you can't use both D-\*AUTO and T-\*AUTO specifications in the same program.

### 19.3.1.3. Fetch Overflow (Column 16)

You use this field to indicate that you want fetch overflow processing. Fetch overflow causes overflow processing at this point in the program rather than at the completion of the total cycle.

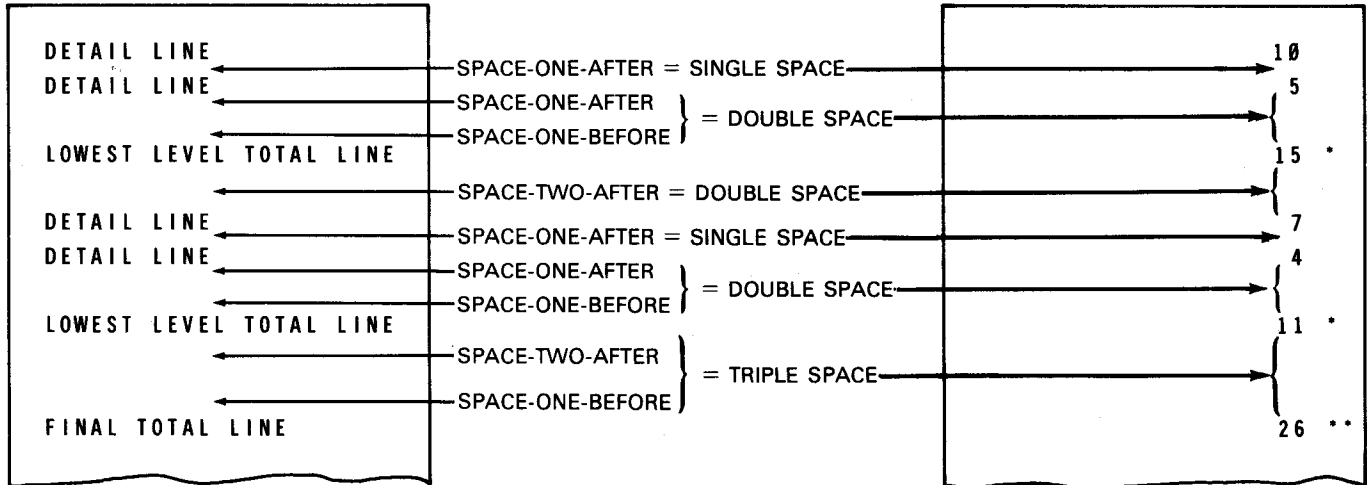
When you don't want fetch overflow processing, leave this column blank.

When you want fetch overflow, enter an F. It applies only to detail lines. See 8.2.3 for further information about fetch overflow.

### 19.3.1.4. Space (Columns 17 and 18)

You use this field to control the spacing of detail lines.

If you want auto report to automatically use a single space (space-one-after) after each detail line and a double space (space-two-after) after each total line, leave these columns blank. Auto report additionally generates one blank line before the lowest level total line and before the final total line (space-one-before):



If you want to specify other spacing values for detail lines, enter them in these columns. You can't specify spacing rules for column headings or total lines. See 8.2.6 for rules on spacing.

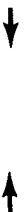
When the detail or total line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess information is printed on the overflow line in the order you specified and is right-justified. If you specified entries in the space field (columns 17 and 18), the spacing for the overflow line is as follows:

- The space-before entries you specified are for the original detail line. Auto report uses a single space after the original line.
- The space-after entries you specified are for the overflow line. Auto report generates blanks for space-before for the overflow line.

See 19.7.6 for more information about overflow lines.

### 19.3.1.5. Skip (Columns 19 through 22)

You use this field to control skipping of detail lines. When you want no skipping by the printer, leave these columns blanks. When you want to specify skipping values for detail lines, enter them in these columns. See 8.2.7 for rules on skipping.



If the detail or total line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess information is printed on the overflow line in the order you specified and is right-justified. If you specified entries in the skip field (columns 19 through 22), the skipping for the overflow line is as follows:

- The skip-before entries you specified are for the original detail line. Auto report uses a single space after the original line.
- The skip-after entries you specified are for the overflow line. Auto report generates blanks for skip-before for the overflow line.

**19.3.1.6. Output Indicators - Records (Columns 23 through 31)**

You use this field to specify output indicators that condition the printing of the detail line. The detail line is printed only when the conditions set by the indicators are met.

If you don't want to specify an output indicator, leave these columns blank. Auto report generates an N1P indicator and this conditions the detail line. Because of this, the detail line isn't printed at first page time.

If you want to specify an output indicator, enter it in these columns. You can use AND or OR if you specify an output indicator on the first D-\*AUTO specification. See 8.2.8 for information about output indicators.

**19.3.1.7. \*AUTO (Columns 32 through 37)**

You use this field to indicate an auto report that can contain both detail lines and total lines.

Enter \*AUTO in these columns and a D in column 15.

**19.3.1.8. Examples of Entries on D-\*AUTO Output File Identification**

Figure 19-11 illustrates the most basic output file identification for a detail report:

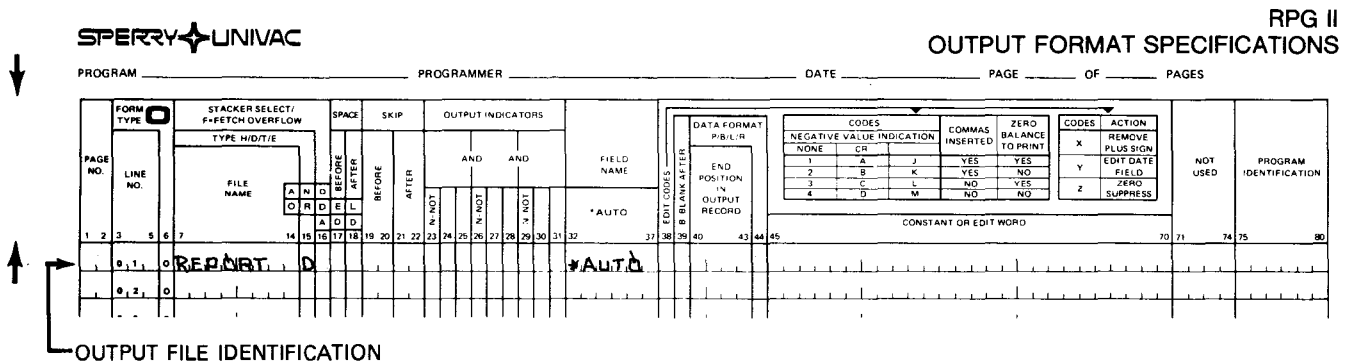


Figure 19-11. Examples of Entries in D-\*AUTO Output File Identification



### 19.3.2. Field Description and Control Entries (Columns 23 through 70)

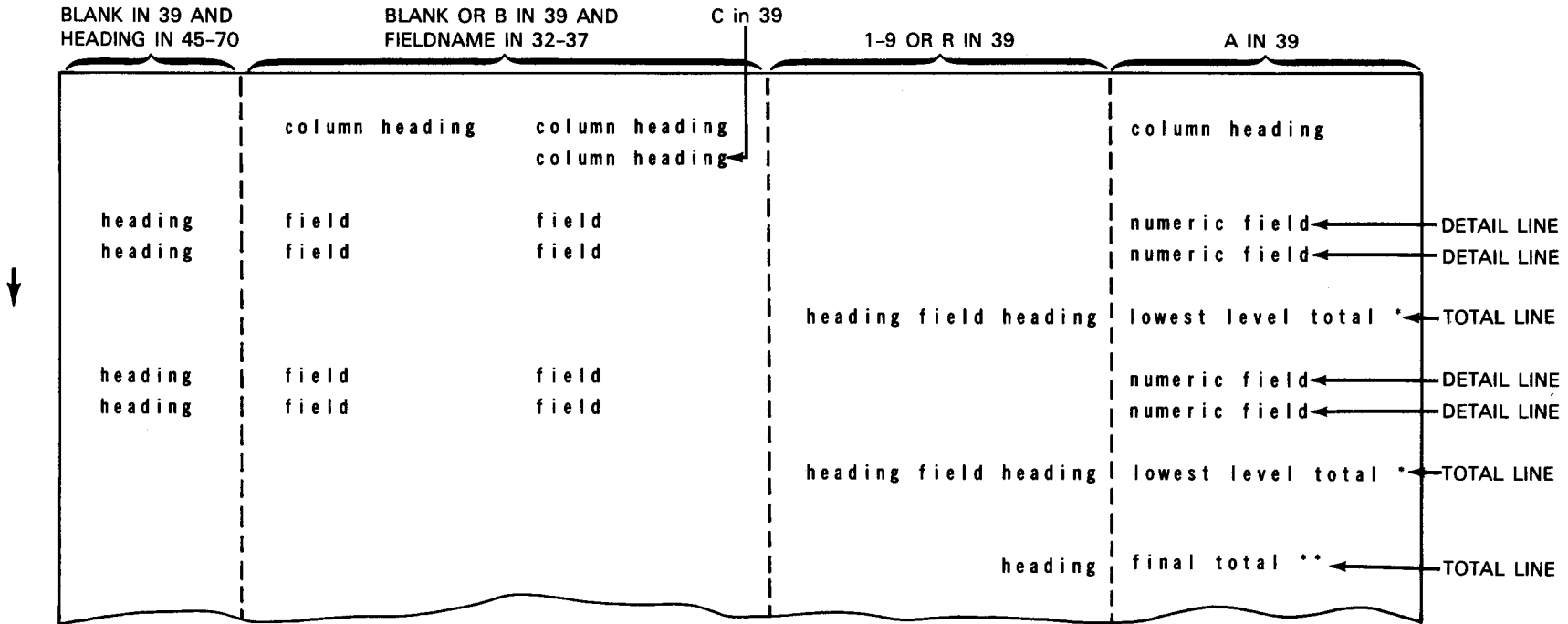
You must follow the D-\*AUTO output file identification with at least one field description. The entry you specify in column 39 determines how a field description is used.

Each of the following field descriptions allow you to format your report in the way you want:

- A field description that prints an alphanumeric or numeric field on each line and optionally prints a column heading over it (blank or B in column 39 and field name in columns 32 through 37)
- A field description that prints a heading (constant) on each detail line (blank in column 39 and heading in columns 45 through 70)
- A field description that prints a numeric field on each detail line and accumulates totals for it on total lines. You can optionally print a column heading over the numeric fields (A in column 39)
- A field description that prints a second or third line of a column heading (C in column 39)
- A field description that prints an alphanumeric or numeric field next to a specific total line (1 through 9 or R in column 39 and a field name in columns 32 through 37)
- A field description that prints a heading (constant) next to a specific total line (1 through 9 or R in column 39 and a heading in columns 45 through 70)

The remaining entries on the field description have different meanings depending on the entry you specify in column 39.

Figure 19-12 illustrates the functions of the D-\*AUTO field descriptions.



a. D-\*AUTO specifications

Figure 19-12. D-\*AUTO Field Descriptions (Part 1 of 2)

BLANK IN 39 AND HEADING IN 45-70	BLANK OR B IN 39 AND FIELDNAME IN 32-37	C in 39	1-9 OR R IN 39	A IN 39
	DEPARTMENT	ANNUAL REPORT		LEVEL
GROUP1	10	PA		10,000.00 ← DETAIL LINE
GROUP2	11	PA		5,000.00 ← DETAIL LINE
			DIVISION 5 SALES	15,000.00 * ← TOTAL LINE
GROUP3	12	PA		4,000.00 ← DETAIL LINE
GROUP4	13	PA		3,000.00 ← DETAIL LINE
			DIVISION 6 SALES	7,000.00 * ← TOTAL LINE
			GRAND TOTAL	22,000.00 ** ← TOTAL LINE

b. Example of printed report

Figure 19-12. D-\*AUTO Field Descriptions (Part 2 of 2)

### 19.3.2.1. Field Description that Prints a Field and Column Heading (Blank or B in Column 39 and Field Name in Columns 32 through 37)

You use this field description to print an alphanumeric or numeric field on a detail line and optionally print a column heading over it.

Figure 19-13 illustrates the D-\*AUTO field description that has a blank or B in column 39 and a field name in columns 32 through 37.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK OR INDICATORS	FIELD NAME	BLANK OR EDIT CODE	BLANK OR B	BLANK OR END POSITION	BLANK	BLANK OR COLUMN HEADING

Figure 19-13. D-\*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)

#### 19.3.2.1.1. Output Indicators - Fields (Columns 23 through 31)

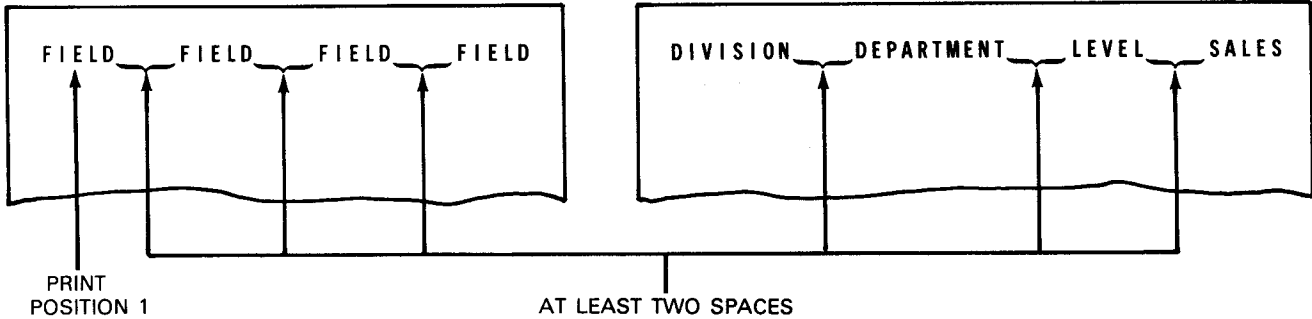
You use this field to specify output indicators that condition the printing of the field you specified in columns 32 through 37. The field is printed on each detail line only when the conditions set by the indicators are met. If you don't want to specify an output indicator, leave these columns blank. Auto report prints the associated field on each detail line conditioned by the indicator.

If you want to specify an output indicator to condition the printing of the associated field, enter it in these columns. By specifying an output indicator, you can suppress printing of common fields and thus reduce repetitive information. If you specify a column heading in columns 45 through 70 to be printed over the field you named in columns 32 through 37, the output indicators don't affect the column heading. See 8.2.8 for information about output indicators.

#### 19.3.2.1.2. Field Name (Columns 32 through 37)

You use this field to specify the name of a field, an indexed array, or a table. The associated field, array, or table value is printed on the detail line.

When you specify more than one name, the fields are printed from left to right on a line in the order you specify. At least two spaces are inserted between each field on a line. The first field, however, begins in print position 1:



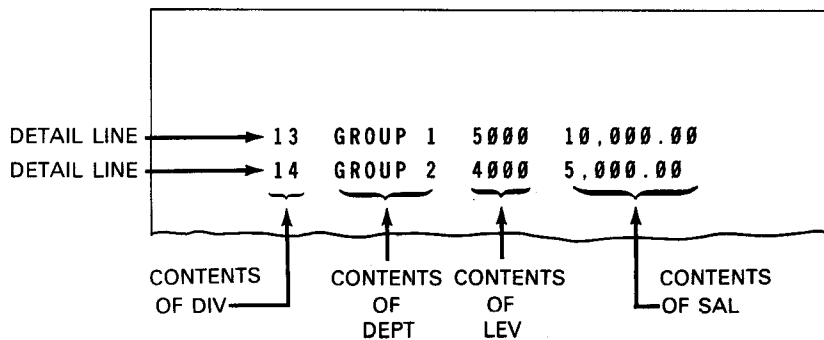
The following example shows you the use of a field name entry and the resulting printed report:

RPG II  
OUTPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES

PAGE NO.	LINE NO.	FORM TYPE	STACKER SELECT/ P-FETCH OVERFLOW	TYPE H/D/T/E	SPACE		SKIP		OUTPUT INDICATORS		FIELD NAME	DATA FORMAT P/B/L/R	CODES				NOT USED	PROGRAM IDENTIFICATION
					BEFORE	AFTER	BEFORE	AFTER	AND	AND			NEGATIVE VALUE INDICATION	COMMAS INSERTED	ZERO BALANCE TO PRINT	CODES		
0,1	0										*AUTO							
0,2	0										DIV							
0,3	0										DEPT							
0,4	0										LEV							
0,5	0										SAL							

OUTPUT FILE IDENTIFICATION      FIELD DESCRIPTIONS



### **19.3.2.1.3. Edit Codes (Column 38)**

You use this field to enter an edit code for a numeric field, indexed array, or table you named in columns 32 through 37.

If you specified an alphanumeric field, table, or indexed array in columns 32 through 37, you must leave this column blank.

If you named a numeric field, indexed array, or table in columns 32 through 37, you can enter an edit code. If you choose not to enter an edit code, auto report generates a K edit code, which prints a numeric field or numeric element with commas and a decimal point where needed (for example, 2,123.77). The K edit code suppresses zeros so that zero balances are not printed and prints negative balances with a minus sign on the right.

### **19.3.2.1.4. Blank After (Column 39)**

You use this field to indicate whether or not you want to reset an alphanumeric field to blanks and a numeric field to zeros after it is printed on the detail line.

If you don't want to reset an alphanumeric field to blanks or reset a numeric field (that isn't totaled) to zeros, leave this column blank.

If you want to reset the field to blanks or zeros, enter a B.

### **19.3.2.1.5. End Position in Output Record (Columns 40 through 43)**

You use this field to specify the end print position of the rightmost character of the field that is printed on the detail line.

If you want auto report to generate end positions for fields and to center column headings, leave these columns blank.

If you want to specify the end position of the rightmost character of the field, enter an end position. Specify end positions to eliminate the automatic spacing between fields or to spread out a report on the page. The end position you specify may be changed a little by auto report when the line is centered or when the column heading and field are positioned in relation to each other. If the end position you specify causes an overlay with another field, auto report generates a new end position.

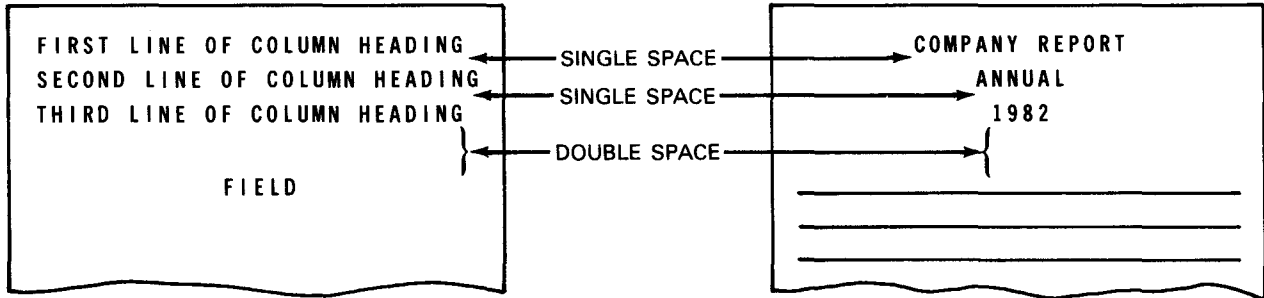
### **19.3.2.1.6. First Column Heading Line (Columns 45 through 70)**

You use this field to print the first line of a column heading (constant) over the field you named in columns 32 through 37.

If you don't want a column heading, leave these columns blank.

If you want to print a column heading, enter it in these columns. You must enclose it within apostrophes. The value is printed over the field you specified in columns 32 through 37.

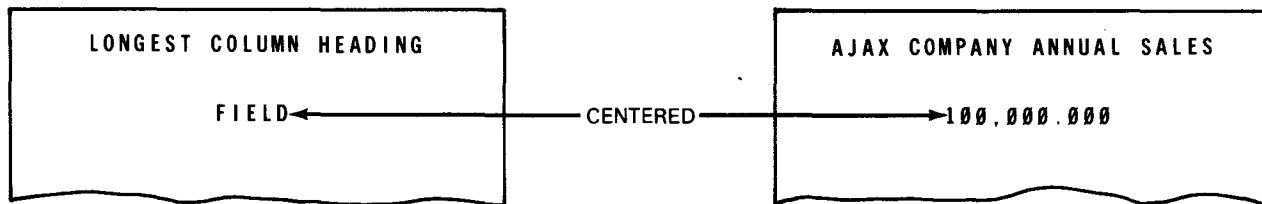
If you specify a column heading, auto report automatically uses a double space after each single column heading line. If you specify a second or third line of column heading (C in column 39), auto report uses a single space after each column heading line except the last, which is followed by a double space:



You can't specify any other spacing for column heading lines.

When you use a column heading, its placement above the field you specified in columns 32 through 37 depends on which is longer, the column heading or the field (including edit characters).

When the column heading is longer than its associated field, the field is centered under the longest column heading constant:



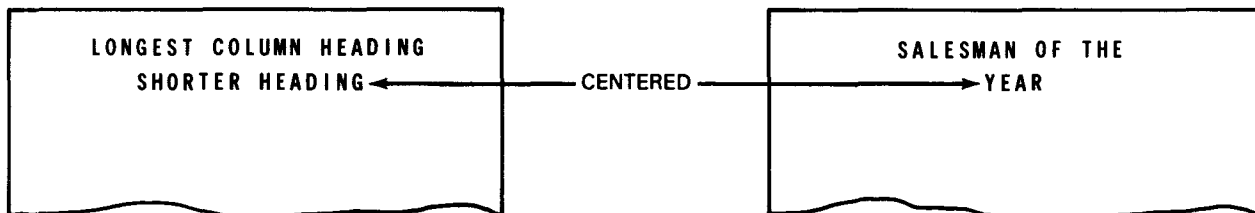
When the column heading is shorter than its associated field, the column heading is left-justified over an alphanumeric field and right-justified over a numeric field:

COLUMN HEADING	COLUMN HEADING
ALPHANUMERIC FIELD	NUMERIC FIELD

DEPT	SALES
DIVISION 1	1,000.00

When you specify more than one column heading, they are printed from left to right on a line in the order you specify. No spaces are provided within a column heading - you must incorporate spaces within the column heading to provide for additional spacing.

You can use column heading continuation lines (C in column 39) to specify a second or third line of a column heading. When you use a multiple line column heading, the shorter column headings are centered under the longest column heading:



If the column heading line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess column headings are printed on the overflow line in the order you specified and are right-justified. See 19.7.6 for more information on overflow lines.

If you want to specify your own positions for the column headings and associated fields, enter the end position in columns 40 through 43 of the \*AUTO field description.





### 19.3.2.2. Field Description that Prints a Heading (Blank in Column 39 and Heading in Columns 45 through 70)

You use this field description to print a heading (constant) on a detail line. There is no column heading over this heading.

Figure 19-15 illustrates the D-\*AUTO field description that has a blank in column 39 and a heading in columns 45 through 70.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK OR INDICATORS	BLANK	BLANK	BLANK	BLANK OR END POSITION	BLANK	HEADING

Figure 19-15. D-\*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)

#### 19.3.2.2.1. Output Indicators - Fields (Columns 23 through 31)

You use this field to specify output indicators that condition the printing of the heading (constant) you specified in columns 45 through 70. The heading is printed on each detail line only when the conditions set by the indicators are met.

If you don't want to specify an output indicator, leave these columns blank. Auto report prints the headings on each detail line conditioned by the indicator.

If you want to specify an output indicator to condition the printing of the heading, enter it in these columns. See 8.2.8 for information about output indicators.

#### 19.3.2.2.2. End Position in Output Record (Columns 40 through 43)

You use this field to specify the end print position of the rightmost character of the heading (constant) that is printed on the detail line.

If you want auto report to generate end positions for headings, leave these columns blank.

If you want to specify the end position of the rightmost character of the heading, enter an end position. Specify end positions only to eliminate the automatic spacing between fields or to spread out a report on the page. If the end position you specify causes an overlay with another field, auto report generates a new end position.



### 19.3.2.3. Field Description that Prints a Numeric Field and Column Heading and Accumulates Totals (A in Column 39)

You use this field description to print a numeric field on a detail line and accumulate totals for it on total lines. You can optionally print a column heading over the numeric field. A total is printed for each control level (L1 through L9) you defined in columns 59 and 60 of the input specifications, and a final total is printed.

Figure 19-17 illustrates the D-\*AUTO field description that has an A in column 39.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK OR INDICATORS	FIELD NAME	BLANK OR EDIT CODE	A	BLANK OR END POSITION	BLANK	BLANK OR COLUMN HEADING

Figure 19-17. D-\*AUTO Field Description (A in Column 39)

Auto report generates calculations to accumulate totals for the fields you specify in columns 32 through 37 and it places the calculations in the following order:

1. Detail calculations you specify
2. EXSR statement for the subroutine named A\$\$SUM generated by auto report. The A\$\$SUM subroutine accumulates the values from the field you specify on an A-type specification onto the lowest level total lines. The EXSR statement is conditioned by the same indicators that condition the D-\*AUTO output file specification. Each add statement in the subroutine is conditioned by the field indicators you specify with the D-\*AUTO field descriptions with an A in column 39.
3. Total calculations generated by auto report. They are sorted by level, starting with L1. These total calculations accumulate totals for the fields you name on D-\*AUTO specifications that have an A in column 39.
4. Total calculations you specify
5. Subroutines you specify
6. RPG II subroutine named A\$\$SUM generated by auto report that accumulates the lowest level total

### 19.3.2.3.1. Output Indicators - Fields (Columns 23 through 31)

You use this field to specify output indicators that condition the printing of the field you specified in columns 32 through 37. The field is printed only when the conditions set by the indicators are met.

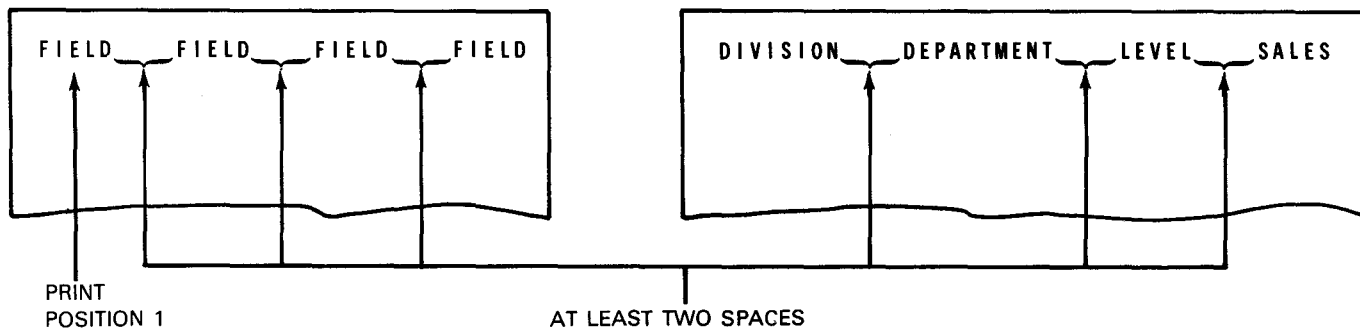
If you don't want to specify an output indicator, leave these columns blank. Auto report prints the associated field on each detail line.

If you specify a column heading in columns 45 through 70 to be printed over the field you named in columns 32 through 37, the output indicators don't affect the column heading. The output indicator you specify also doesn't affect the field descriptions generated by auto report for totals. However, if you specify an N1P output indicator on the D-\*AUTO output file specification, the calculations generated for the totaling fields are also conditioned by N1P. This causes a diagnostic in the RPG II compiler. See 8.2.8 for information about output indicators. ←

### 19.3.2.3.2. Field Name (Columns 32 through 37)

You use this field to specify the name of a numeric field that is accumulated and printed on each detail line.

When you specify more than one name, the fields are printed from left to right on a line in the order you specify. At least two spaces are inserted between each field on the line. The first field, however, begins in print position 1:



Auto report generates an RPG II subroutine named A\$\$SUM that creates and names additional totaling fields based on the field name you specify to generate calculation and output specifications that accumulate and print the various levels of totals required.

→ If you want to accumulate a numeric field on each detail line, enter the name of the field in columns 32 through 37 and an A in column 39. The associated field is printed on the detail line and totals for it are printed on total lines. You can't identify an array, array element, or table.

You can specify totaling for any particular field only once in each program.

→ Pay attention to field names that end with a number from 1 through 9 or an R in an auto report program that accumulates totals, because auto report generates total fields that end in those characters.

If the field name you specify has fewer than 6 characters, a character from 1 through 9 or an R is added to the name to create a name for the total field. This added character corresponds to the total indicators L1 through L9 and LR, respectively. For example, if you specify a field name of GROUP and all nine control levels are defined, the generated field names are GROUP1, GROUP2,...GROUP9, and GROUPLR.

If the field name you specify has 6 characters, the last character is replaced by a character from 1 through 9 or an R. For example, if you specify a field name of SUPPLY and all nine control levels are defined, the generated field names are SUPPL1, SUPPL2,..., SUPPL9, and SUPPLR.

You can't use any one field name more than once if you use an A in column 39. Also, if you specify a 5- or 6-character field name with an A in column 39, you can't specify a second 5- or 6-character field name that has the same first five characters with an A in column 39. For example, if you specify the following three field names with an A in column 39, all but the first are invalid:

GROUP	Valid
GROUPW	Invalid because the first five characters duplicate the first five characters of the first field
GROUP	Invalid because it is a duplicate of the first field



DIVISION	DEPARTMENT	LEVEL	SALES
_____	_____	_____	12,000.00
_____	_____	_____	5,000.00
			17,000.00 * ← L1
_____	_____	_____	11,000.00
_____	_____	_____	4,000.00
			15,000.00 * ← L1
			32,000.00 ** ← L2
_____	_____	_____	7,000.00
_____	_____	_____	5,000.00
			12,000.00 * ← L1
_____	_____	_____	10,000.00
_____	_____	_____	11,000.00
			21,000.00 * ← L1
			33,000.00 ** ← L2
			65,000.00 *** ← LR

**19.3.2.3.3. Edit Codes (Column 38)**

You use this field to enter an edit code for the numeric field you named in columns 32 through 37 as well as for all the generated total fields.

If you don't want to enter an edit code, leave this column blank. Auto report generates a K edit code, which prints a numeric field with commas and a decimal point where needed (for example, 2,123.77). The K edit code also suppresses zeros so that zero balances are not printed and negative balances are printed with a minus sign on the right.

If you want to enter an edit code, enter it in this column. See 8.3.3 for information about edit codes.



#### 19.3.2.3.4. Accumulate Totals (Column 39)

You use this field to indicate that you want to accumulate and print totals for the numeric field you named in columns 32 through 37. You can specify totaling only once for any particular field in each program.

If you want to accumulate and print totals for the numeric field you named in columns 32 through 37, enter an A. Auto report generates a subroutine named A\$\$SUM that accumulates the values from the fields into the lowest level total lines.

Auto report generates a B (blank after) in column 39 of all the detail and total field descriptions generated for the field name you specify. Thus, the value in the field you specified (and in any generated field) is reset to zero after the field value is printed.

Total fields are generated and named for all control level indicators you defined in columns 59 and 60 of the input specifications. Total calculations accumulate totals for the field you named in columns 32 through 37. Total calculations roll (add) the total from the lowest level total field to that of the next higher level field and end with a final total. This process is called total rolling.

For example, if you assign L1 and L3 to control fields on the input specification and specify the field PRD in columns 32 through 37 of the field description, three total fields (PRD1, PRD3, and PRDR) are generated and named by auto report. All total fields generated for the same level, such as PRD1 and SUPPL1 are printed on the same total line, and that line is conditioned by the corresponding control level indicator.

Generated total fields are two digits longer than the original field, but the number of decimal positions remain the same in the generated fields. For example, if you use an edit code to define a field with a length of four characters (xx.xx), the length of each generated field is six characters (x,xxx.xx).

If the generated field is identical to a previously defined numeric field, the generated field is assigned the previously defined length and number of decimal positions (if the previously defined field is numeric).

Auto report prints asterisks to the right of generated total lines so you can tell they're total lines. One asterisk is printed to the right of the lowest level total line and two asterisks are printed to the right of the next level total and so on. For example, if you defined L1 and L3 control level indicators in a program, one asterisk is printed to the right of the L1 line, two asterisks are printed on the L3 line, and three asterisks are printed on the LR line. As many as 10 asterisks are printed on the LR line if you define control level indicators in the program.

To suppress the generation of asterisks on total lines, enter an N in column 28 of the auto report options specifications form.

**19.3.2.3.5. End Position in Output Record (Columns 40 through 43)**

You use this field to specify the end print position of the rightmost character of the field that is printed on the total line.

If you want auto report to generate end positions for fields and to center column headings, leave these columns blank.

If you want to specify the end position of the rightmost character of the field, enter an end position. Specify end positions only to eliminate the automatic spacing between fields or to spread out a report on the page. The end position you specify may be changed a little by auto report when the line is centered or when the column heading and field are positioned in relation to each other. If the end position you specify causes an overlay with another field, auto report generates a new end position.

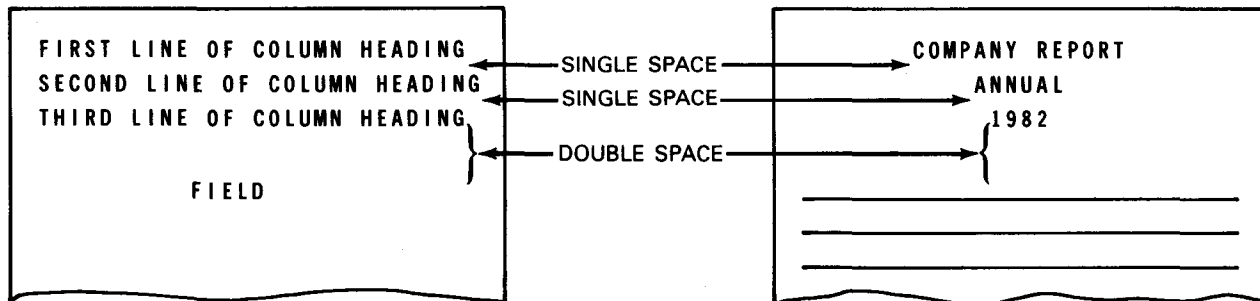
**19.3.2.3.6. First Column Heading Line (Columns 45 through 70)**

You use this field to print the first line of a column heading (constant) over the field you named in columns 32 through 37.

If you don't want a column heading, leave these columns blank.

If you want to print a column heading, enter it in these columns. You must enclose it within apostrophes. The value is printed over the accumulated total of the field you specified in columns 32 through 37.

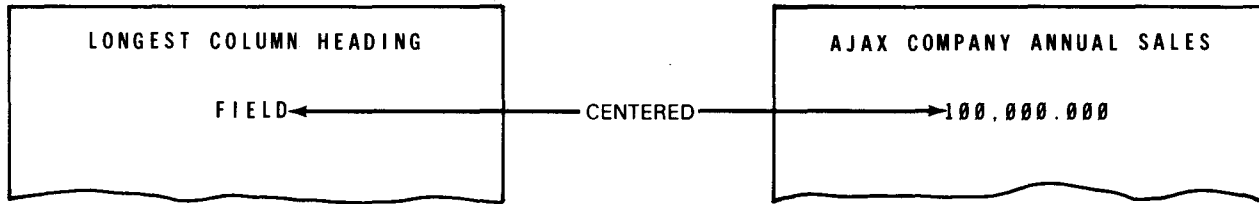
When you specify a column heading, auto report automatically uses a double space after each single column heading line. If you specify a second or third line of column heading (C in column 39), auto report uses a single space after each column heading line except the last, which is followed by a double space:



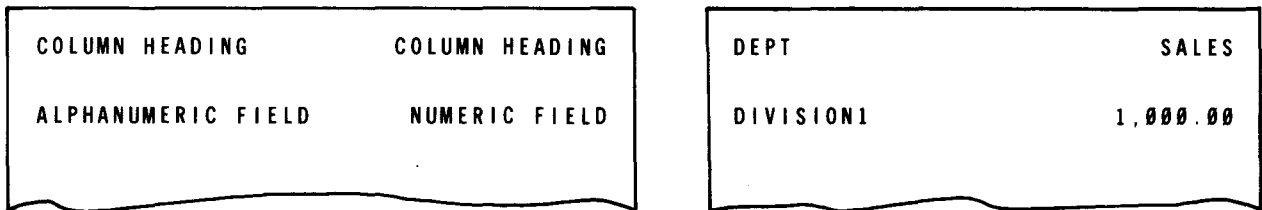
You can't specify any other spacing for column heading lines.

When you use a column heading, its placement above the field you specified in columns 32 through 37 depends on which is longer, the column heading or the field (including edit characters).

When the column heading is longer than its associated field, the field is centered under the longest column heading constant:

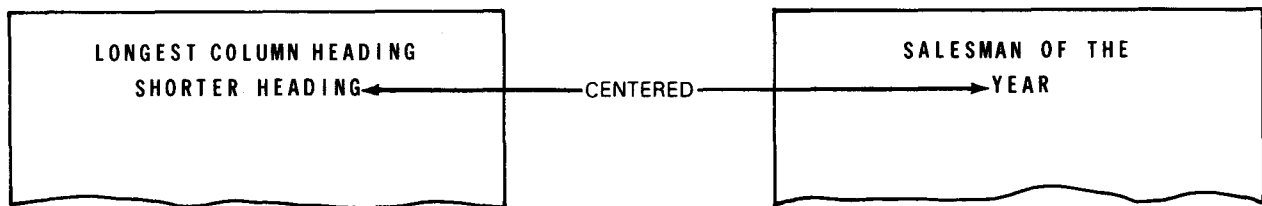


When the column heading is shorter than its associated field, the column heading is left-justified over an alphanumeric field and right-justified over a numeric field:



When you specify more than one column heading, these headings are printed from left to right on a line in the order you specify. No spaces are provided within a column heading – you must incorporate spaces within the column heading to provide for additional spacing.

You can use column heading continuation lines (C in column 39) to specify a second or third column heading. When you use a multiple line column heading, the shorter column headings are centered under the longest column heading.



If the column heading line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess column headings are printed on the overflow line in the order you specified and are right-justified.

If you want to specify your own end position for column headings and their associated fields, enter the end position in columns 40 through 43 of the D-\*AUTO specification.



#### 19.3.2.4. Field Description that Prints a Second and Third Column Heading Line (C in Column 39)

You use this field description to print a second or third line of a column heading. You can use one or two C specifications following a field description that has an A, B, or blank in column 39 and a field name in columns 32 through 37.

Figure 19-19 illustrates a D-\*AUTO field description that has a C in column 39.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	BLANK	BLANK	C	BLANK	BLANK	COLUMN HEADING

Figure 19-19. D-\*AUTO Field Description (C in Column 39)

##### 19.3.2.4.1. Column Heading Continuation Lines (Column 39)

You use this field to indicate that you want a second and third column heading line when there is more information in a column heading than can be contained on one line.

If you want to specify continuation lines for a column heading, enter a C. The column heading you specify in columns 45 through 70 is printed on the second or third line of the column heading.

##### 19.3.2.4.2. Second and Third Column Heading Line (Columns 45 through 70)

You use this field to print a second or third column heading line.

If you don't want to specify a second or third column heading line, leave this field description blank. ←

If you want to specify continuation lines for a column heading you specified on a field description with a B, blank, or A in column 39, enter them in these columns. If this is the first C specification, you are specifying the second line of the column heading. If this is the second C specification, you are specifying the third line of the column heading. You can't use more than two C specifications. You can specify column headings continuation lines that are up to 24 characters long, including blanks. You must enclose them within apostrophes. Short column headings are centered under the long column headings.



### 19.3.2.5. Field Description that Prints a Heading next to a Total (1 through 9 or R in Column 39)

You use this field description to print a heading (constant) on a specific total line generated from an A in column 39. You can only use this field description if you used a field description with an A in column 39. The heading you enter is printed on the total line that corresponds to the level (L1 through L9 or LR) you specified in column 39.

Figure 19-21 illustrates the D-\*AUTO field description that has a number from 1 through 9 or an R in column 39.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	BLANK	BLANK	1-9, R	BLANK	BLANK	HEADING

Figure 19-21. D-\*AUTO Field Description (1 through 9 or R in Column 39)

#### 19.3.2.5.1. Specific Total Line (Column 39)

You use this field to identify a specific total line on which the heading (constant) is printed. The heading is printed to the left of the leftmost total on the line.

If you didn't define any of the field descriptions with an A in column 39, you can't use an entry in this field description. ←

If you want to print the heading on a specific total line, enter a number from 1 through 9 or an R. This entry corresponds to the indicators L1 through L9, which you defined in columns 59 and 60 of the input specifications, and LR. For example, 4 in column 39 indicates the information is printed on the L4 total line, and an R in column 39 indicates the information is printed on the final total (LR) line.

#### 19.3.2.5.2. Heading (Columns 45 through 70)

You use this field to print a heading (constant) on a specific total line. You must enclose the heading within apostrophes.

When you specify a heading to appear next to a total, it is printed to the left of the first total. It is separated from the total by two spaces. If you specify two or more headings (or fields) to appear next to a total, they are printed in the order you specified:





**19.3.2.6. Field Description that Prints a Field next to a Total (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)**

You use this field description to print an alphanumeric or numeric field on a specific total line. You can only use this field description if you also used a field description with an A in column 39. The field you enter is printed on the total line that corresponds to the level (L1 through L9 or LR) you specified in column 39.

Figure 19-23 illustrates the D-\*AUTO field description that has a number from 1 through 9 or an R in column 39 and a field name in columns 32 through 37.

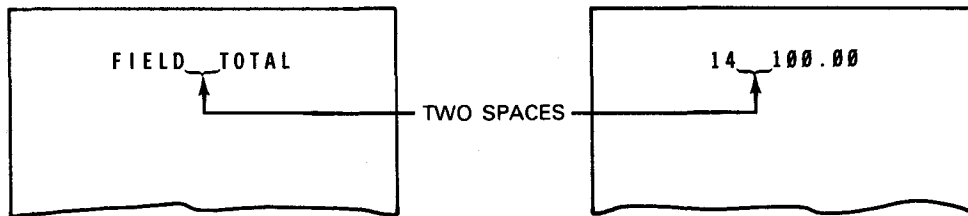
COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	FIELD NAME	BLANK OR EDIT CODE	1-9 OR R	BLANK	BLANK	BLANK OR EDIT WORD

Figure 19-23. D-\*AUTO Field Description (1 through 9 or R in Column 39 and a Field Name in Columns 32 through 37)

**19.3.2.6.1. Field Name (Columns 32 through 37)**

You use this field to specify the name of a field, an indexed array, or a table. The associated field, array, or table is printed on a specific total line.

When you specify a field to appear next to a total, it is printed to the left of the first total. It is separated from the total by two spaces:



If you specify two or more fields (or headings) to appear next to a total, they are printed from left to right on a line in the order you specified. Each field is separated by one space.

#### **19.3.2.6.2. Edit Codes (Column 38)**

You use this field to enter an edit code for a numeric field, indexed array, or table you named in columns 32 through 37.

If you specified an alphanumeric field, indexed array, or table in columns 32 through 37, you must leave this column blank. Also, leave this column blank if you don't want to use an edit code.

If you named a numeric field, indexed array, or table in columns 32 through 37, you can enter an edit code. When you specify an edit code, you must leave columns 45 through 70 blank.

#### **19.3.2.6.3. Specific Total Line (Column 39)**

You use this field to identify a specific total line on which the field you described in columns 32 through 37 is printed. The field is printed to the left of the leftmost total on the line.

If you didn't define any of the field descriptions with an A in column 39, you can't use an entry in this column.

If you want to print the associated field on a specific total line, enter a number from 1 through 9 or an R. This entry corresponds to the indicators L1 through L9 and LR that you defined in columns 59 and 60 of the input specifications. For example, 4 in column 39 indicates the information is printed on the L4 total line, and an R in column 39 indicates the information is printed on the final total (LR) line.

#### **19.3.2.6.4. Edit Word (Columns 45 through 70)**

You use this field to edit numeric fields that you want printed on a specific total line.

If you specified an edit code in column 38, you must leave these columns blank. Also, leave these columns blank if you don't want to specify an edit word.

If you want to edit the field, enter an edit word. See 8.3.8 for information about edit words.

#### **19.3.2.6.5. Examples of Entries on D-\*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)**

Figure 19-24 illustrates a report that contains a heading and field printed next to the lowest level total line.







### 19.4. T-\*AUTO TOTAL REPORTS

You use the T-\*AUTO specification on the output specifications form to print a report that contains only total lines. In other words, there are no detail lines. The data is summarized for a group of input records and only the totals are printed on the report. You can specify a report that has subtotals and final total or a report that has only a final total.

You can specify the T-\*AUTO specification alone or in combination with standard RPG II specifications. You can't use both a T-\*AUTO and D-\*AUTO specification in the same program. You can specify T-\*AUTO for only one printer file per program.

You can also use the T-\*AUTO specification to:

- Print up to three lines of column headings over a field
- Accumulate several levels of totals
- Print information next to a total
- Format the report

The standard RPG II output specifications form is used for two types of entries: output file identification (columns 7 through 31) and field description (columns 23 through 70). T-\*AUTO specifications have the same two types of entries, but their use is not the same.

#### 19.4.1. Output File Identification and Control Entries (Columns 7 through 37)

You must specify a T in column 15 and \*AUTO in columns 32 through 36 for an auto report output file identification, which indicates that the report contains total lines only. You must follow it with at least one field description that describes when, where, or how the lines are printed.

Figure 19-26 illustrates the T-\*AUTO output file identification.

COLS:	6	7-13	15	16	17-18	19-22	23-31	32-37	38-70
	O	FILENAME	T	BLANK OR FETCH OVERFLOW	BLANK OR SPACE	BLANK OR SKIP	BLANK OR INDICATORS	*AUTO	BLANK

Figure 19-26. T-\*AUTO Output File Identification

#### 19.4.1.1. File Name (Columns 7 through 13)

You use this field to specify the name of the printer file that prints the report.

Enter the name of the printer file. The name you specify for this file must be the same name you specified on the H-\*AUTO page heading if you used one. You only need to specify a file name on the first output file identification for a file.

The file name can be up to seven characters in length. See 5.2.1 for more explanation.

**NOTE:**

*The file name can be eight characters long.*

#### 19.4.1.2. Type (Column 15)

You use this field to indicate that you want a report that contains total lines only.

Enter a T in this column and \*AUTO in columns 32 through 36. Remember that you can't use both D-\*AUTO and T-\*AUTO specifications in the same program.

#### 19.4.1.3. Fetch Overflow (Column 16)

You use this field to indicate that you want fetch overflow processing. Fetch overflow causes overflow processing at this point in the program rather than at the completion of the total cycle.

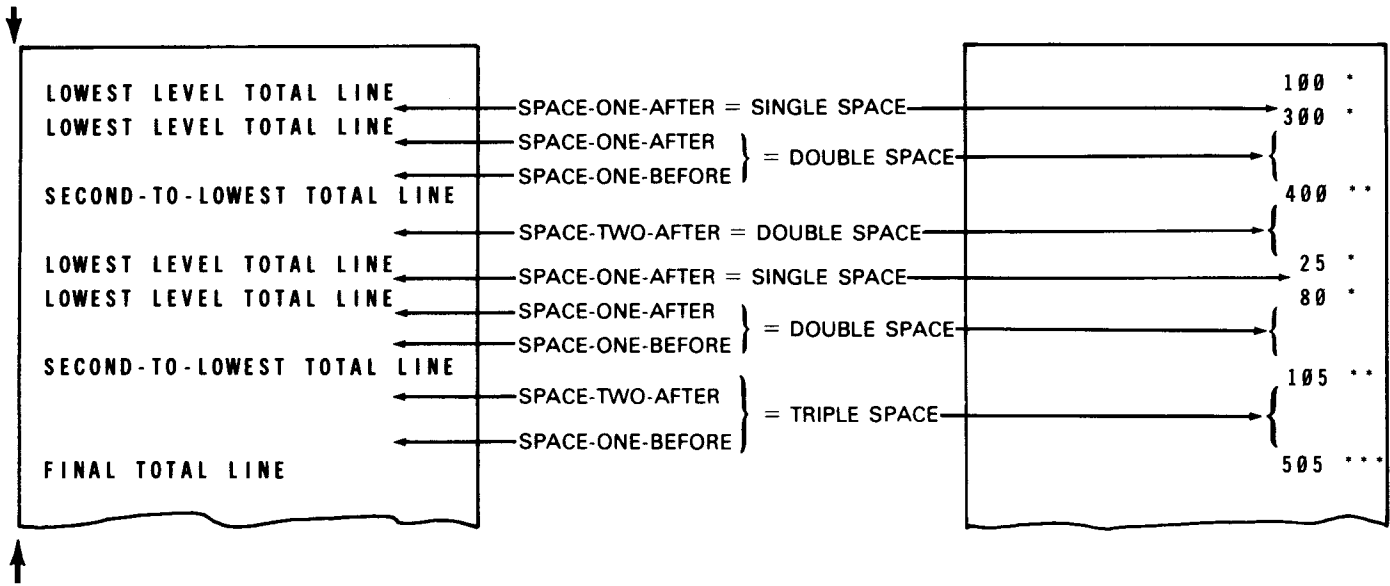
When you don't want fetch overflow processing, leave this column blank.

When you want fetch overflow, enter an F. It applies only to the lowest level total line. See 8.2.3 for further information about fetch overflow.

#### 19.4.1.4. Space (Columns 17 and 18)

You use this field to control the spacing of the lowest level total lines.

If you want auto report to automatically use a single space (space-one-after) after each lowest level total line and a double space (space-two-after) after each higher level total line, leave these columns blank. Auto report additionally generates one blank line (space-one-before) before the second-to-the-lowest level total line and before the final total line.



If you want to specify other spacing values for the lowest level total lines, enter them in these columns. You can't, however, specify any other spacing for higher level total lines. See 8.2.6 for rules on spacing.

If the total line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess information is printed on the overflow line in the order you specified and is right-justified. If you specified entries in the space field (columns 17 and 18), the spacing for the overflow line is as follows:

- The space-before entries you specified are for the original lowest level total line. Auto report uses a single space after the original line.
- The space-after entries you specified are for the overflow line. Auto report generates blanks for space-before for the overflow line.



The following example shows you the use of a space entry and the resulting printed report:

RPG II  
OUTPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES

PAGE NO.	LINE NO.	FORM TYPE	STACKER SELECT/ F-FETCH OVERFLOW TYPE H/D/T/E	SPACE		SKIP			OUTPUT INDICATORS			FIELD NAME	DATA FORMAT P/B/L/R	CODES				NOT USED	PROGRAM IDENTIFICATION			
				BEFORE	AFTER	BEFORE	AFTER	AND	AND	AND	NEGATIVE VALUE INDICATION			COMMAS INSERTED	ZERO BALANCE TO PRINT	CODES	ACTION					
1	2																					
0,1	0		R.P.R.T.																			
0,2	0																					

NO ENTRY

01/15/80	COMPANY REPORT	PAGE 1
DIVISION	DEPARTMENT	LEVEL                      SALES
2	12	500                      5,000.00 *
4	16	100                      10,000.00 *

THE LOWEST LEVEL TOTAL LINES ARE SINGLE-SPACED LIKE DETAIL LINES.

**19.4.1.5. Skip (Columns 19 through 22)**

You use this field to control skipping of the lowest level total lines. When you want no skipping by the printer, leave those columns blank. When you want to specify skipping values for total lines, enter them in these columns. These entries apply only to the lowest level total line. See 8.2.7 for rules on skipping.



If the total line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess information is printed on the overflow line in the order you specified and is right-justified. If you specified entries in the skip field (columns 19 through 22), the skipping for the overflow line is as follows:

- The skip-before entries you specified are for the lowest level total line. Auto report uses a single space after the original or lowest total line.
- The skip-after entries you specified are for the overflow line. Auto report generates blanks for skip-before for the overflow line.

#### **19.4.1.6. Output Indicators - Records (Columns 23 through 31)**

You use this field to specify output indicators that condition the printing of the lowest level total line generated by this T-\*AUTO specification. The lowest level total line is printed only when the conditions set by the indicators are met.

If you don't want to specify an output indicator, leave these columns blank. Auto report conditions the first generated total line by the lowest control level indicator you defined in the program.

If you want to specify an output indicator to condition the printing of the lowest level total line, enter it in these columns. Total lines are printed only for the output indicator you specify and higher numbered indicators. For example, if you define L1, L2, and L3 and then specify L2 in these columns, totals for L2, L3, and LR are printed but L1 isn't printed. In other words, when the lowest control level indicator you used on the T-\*AUTO output file identification is higher than the lowest control level indicator you defined on the input specification, auto report generates total lines that correspond to the lowest level indicator you used on the T-\*AUTO specification, the higher level indicators, and LR.

You can use AND or OR if you specify an output indicator on the first T-\*AUTO output file identification. See 8.2.8 for information about output indicators.

#### **19.4.1.7. \*AUTO (Columns 32 through 37)**

You use this field to indicate an auto report that contains total lines only.

Enter \*AUTO in these columns and a T in column 15.

#### **19.4.1.8. Examples of Entries on T-\*AUTO Output File Identification**

Figure 19-27 illustrates the most basic output file identification for a total report.



RPG II  
OUTPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES

PAGE NO.	LINE NO.	FORM TYPE	STACKER SELECT/ F-FETCH OVERFLOW	TYPE H/D/T/E	SPACE			SKIP			OUTPUT INDICATORS			FIELD NAME	DATA FORMAT P/B/L/R	CODES				NOT USED	PROGRAM IDENTIFICATION																
					BEFORE	AFTER	BEFORE	AFTER	BEFORE	AFTER	BEFORE	AFTER	BEFORE			AFTER	BEFORE	AFTER	NONE			CR	COMMAS INSERTED	ZERO BALANCE TO PRINT	X	REMOVE PLUS SIGN											
1	2	3	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45	70	71	74	75	80	
	0,1	0		REPORT																																	
	0,2	0																																			

CONSTANT OR EDIT WORD

OUTPUT FILE IDENTIFICATION

Figure 19-27. Examples of Entries on T-\*AUTO Output File Identification

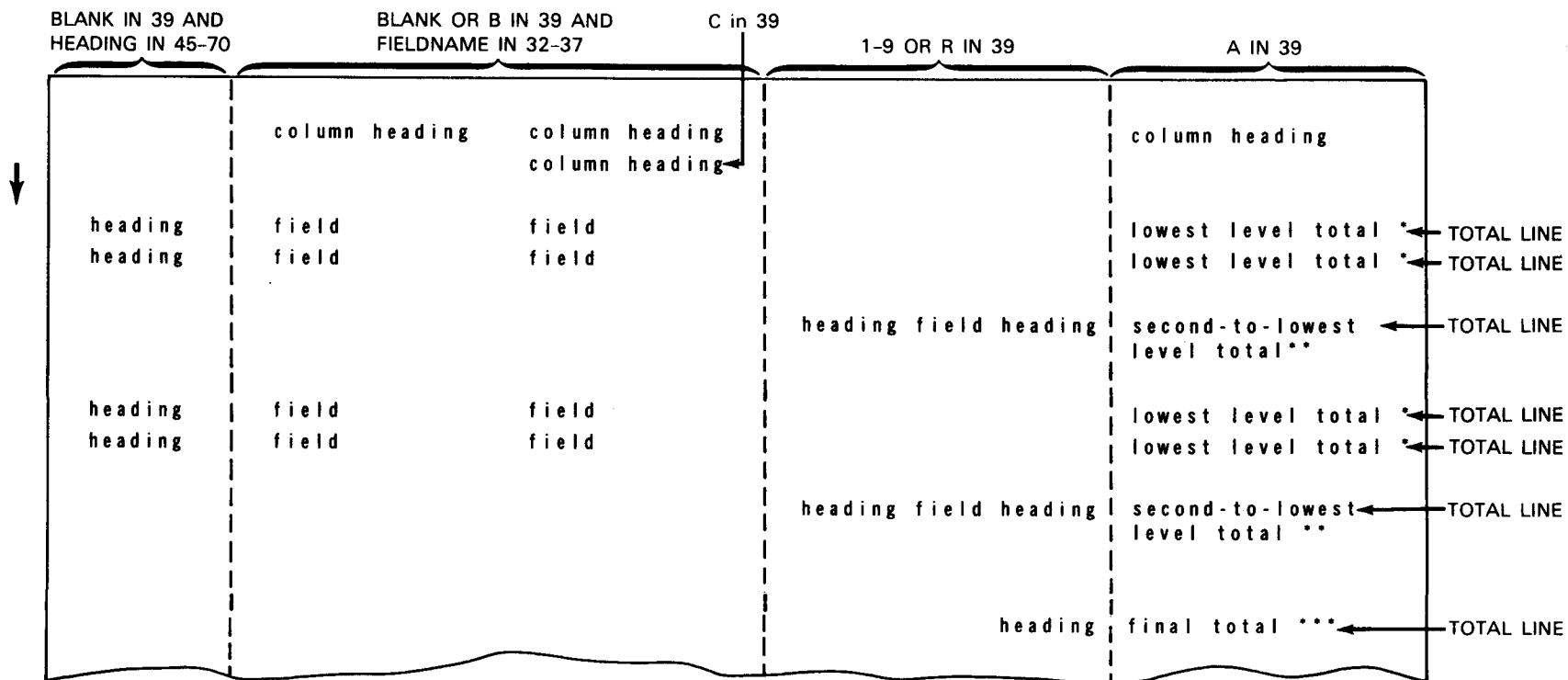
### 19.4.2. Field Description and Control Entries (Columns 23 through 70)

You must follow the T-\*AUTO output file identification with at least one field description. The entry you specify in column 39 determines how these field descriptions are used.

Each of the following field descriptions allow you to format your report in the way you want:

- A field description that prints an alphanumeric or numeric field on the lowest level total line and optionally prints a column heading over it (blank or B in column 39 and field name in columns 32 through 37). The B indicates that the field is reset to zero (numeric) or blank (alphanumeric).
- A field description that prints a heading (constant) on each total line (blank in column 39 and heading in columns 45 through 70)
- A field description that prints a numeric field on the lowest level total line and accumulates totals for it on higher level lines. You can optionally print a column heading over the numeric field (A in column 39).
- A field description that prints a second or third line of a column heading (C in column 39)
- A field description that prints an alphanumeric or numeric field next to a specific total line (1 through 9 or R in column 39 and a field name in columns 32 through 37)
- A field description that prints a heading (constant) next to a specific total line (1 through 9 or R in column 39 and a heading in columns 45 through 70)

The remaining entries on the field description have different meanings depending on the entry you specify in column 39. Figure 19-28 illustrates the functions of the T-\*AUTO field descriptions.



a. T-AUTO specifications

Figure 19-28. T-AUTO Field Descriptions (Part 1 of 2)

BLANK IN 39 AND HEADING IN 45-70	BLANK OR B IN 39 AND FIELDNAME IN 32-37	C in 39	1-9 OR R IN 39	A IN 39
	DEPARTMENT	ANNUAL REPORT		LEVEL
GROUP1	10	PA		15,000.00 * ← TOTAL LINE
GROUP2	11	PA		7,000.00 * ← TOTAL LINE
			DIVISION 5 SALES	22,000.00 ** ← TOTAL LINE
GROUP3	12	PA		4,000.00 * ← TOTAL LINE
GROUP4	13	PA		3,000.00 * ← TOTAL LINE
			DIVISION 6 SALES	7,000.00 ** ← TOTAL LINE
			GRAND TOTAL	29,000.00 *** ← TOTAL LINE

b. Example of printed report

Figure 19-28. T-AUTO Field Descriptions (Part 2 of 2)

**19.4.2.1. Field Description that Prints a Field and Column Heading (Blank or B in Column 39 and Field Name in Columns 32 through 37)**

You use this field description to print an alphanumeric or numeric field on the lowest level total line and optionally print a column heading over it.

Figure 19-29 illustrates the T-\*AUTO field description that has a blank or B in column 39 and a field name in columns 32 through 37.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK OR INDICATORS	FIELD NAME	BLANK OR EDIT CODE	BLANK OR B	BLANK OR END POSITION	BLANK	BLANK OR COLUMN HEADING

Figure 19-29. T-\*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)

**19.4.2.1.1. Output Indicators - Fields (Columns 23 through 31)**

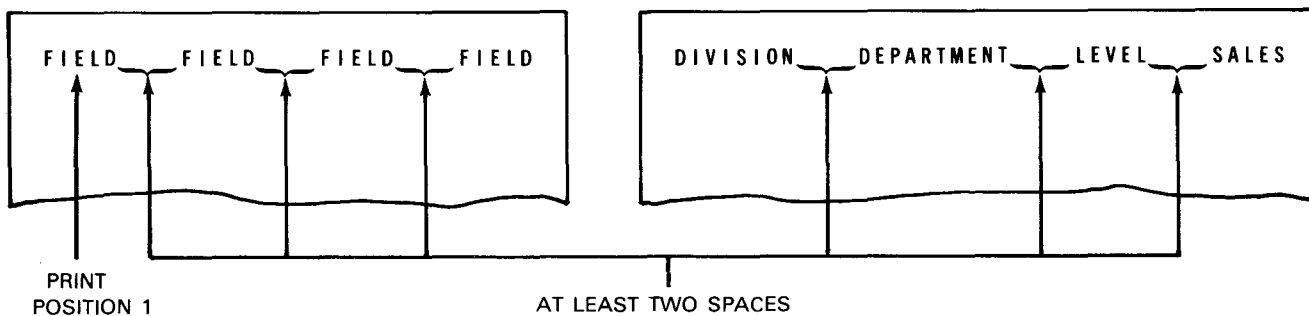
You use this field to specify output indicators that condition the printing of the field you specified in columns 32 through 37. The field is printed on the lowest level total line only when the conditions set by the indicators are met. If you don't want to specify an output indicator, leave these columns blank. Auto report prints the associated field each time the lowest level total line is printed.

If you want to specify an output indicator, enter it in these columns. By specifying an output indicator, you can suppress printing of common fields and thus reduce repetitive information. If you specify a column heading in columns 45 through 70 to be printed over the field you named in columns 32 through 37, the output indicators don't affect the column heading. See 8.2.8 for information about output indicators.

**19.4.2.1.2. Field Name (Columns 32 through 37)**

You use this field to specify the name of a field, an indexed array, or a table. The associated field, array, or table value is printed on the lowest level total line.

When you specify more than one name, the fields are printed from left to right on a line in the order you specify. At least two spaces are inserted between each field on a line. The first field, however, begins in print position 1:



**19.4.2.1.3. Edit Codes (Column 38)**

You use this field to enter an edit code for a numeric field, indexed array, or table you named in columns 32 through 37.

If you specified an alphanumeric field, table, or indexed array in columns 32 through 37, you must leave this column blank.

If you named a numeric field, indexed array, or table in columns 32 through 37, you can enter an edit code. If you choose not to enter an edit code, auto report generates a K edit code, which prints a numeric field or numeric element with commas and a decimal point where needed (for example, 2,123.77). The K edit code also suppresses zeros so that zero balances are not printed and negative balances are printed with a minus sign on the right.

**19.4.2.1.4. Blank After (Column 39)**

You use this field to indicate whether or not you want to reset an alphanumeric field to blanks and a numeric field to zeros after it is printed on the first total line.

If you don't want to reset an alphanumeric field to blanks or reset a numeric field (that isn't totaled) to zeros, leave this column blank.

If you want to reset the field to blank or zeros, enter a B.

**19.4.2.1.5. End Position in Output Record (Columns 40 through 43)**

You use this field to specify the end print position of the rightmost character of the field that is printed on the first total line.

If you want auto report to generate end positions for fields and to center column headings, leave these columns blank.

If you want to specify the end position of the rightmost character of the field, enter an end position. Specify end positions only to eliminate the automatic spacing between fields or to spread out a report on the page. The end position you specify may be changed a little by auto report when the line is centered or when the column heading and field are positioned in relation to each other. If the end position you specify causes an overlay with another field, auto report generates a new end position.

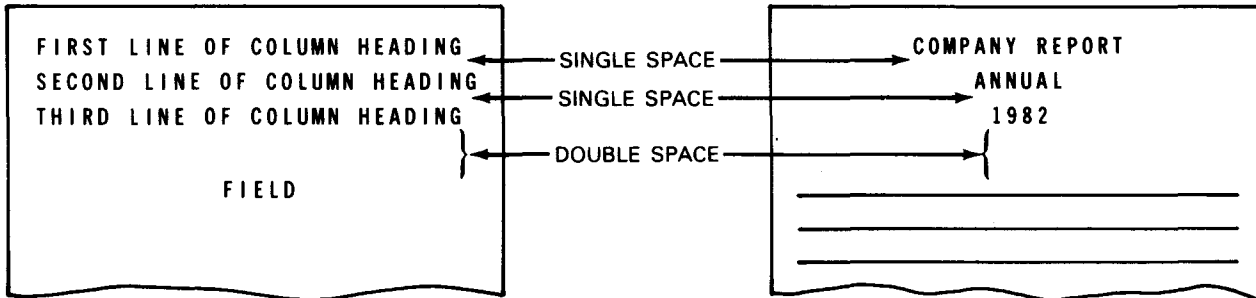
**19.4.2.1.6. First Column Heading Line (Columns 45 through 70)**

You use this field to print the first line of a column heading (constant) over the field you named in columns 32 through 37.

If you don't want a column heading, leave these columns blank.

If you want to print a column heading, enter it in these columns. You must enclose it within apostrophes. The value is printed over the field you specified in columns 32 through 37.

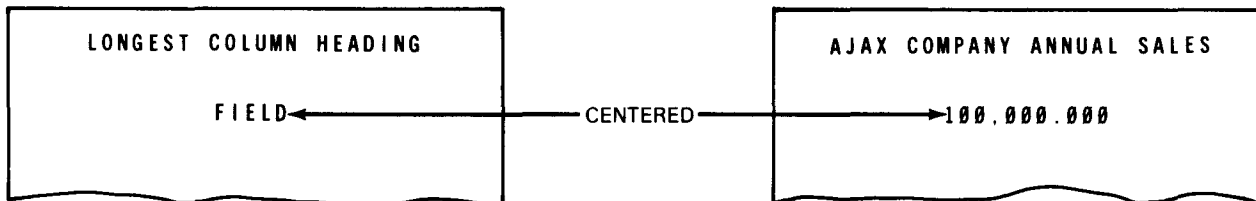
When you specify a column heading, auto report automatically uses a double space after each single column heading line. If you specify a second or third line of column heading (C in column 39), auto report uses a single space after each column heading line except the last, which is followed by a double space:



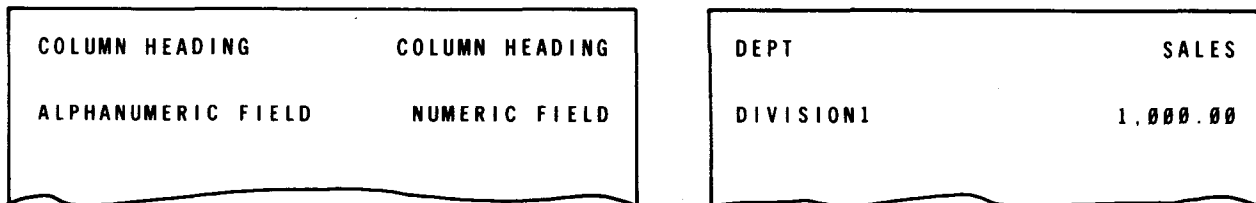
You can't specify any other spacing for column heading lines.

When you use a column heading, its placement above the field you specified in columns 32 through 37 depends on which is longer, the column heading or the field (including edit characters).

When the column heading is longer than its associated field, the field is centered under the longest column heading constant:



When the column heading is shorter than its associated field, the column heading is left-justified over an alphanumeric field and right-justified over a numeric field:



When you specify more than one column heading, they are printed from left to right on a line in the order you specify. No spaces are provided within a column heading - you must incorporate spaces within the column heading to provide for additional spacing.





	01/15/80		COMPANY REPORT		PAGE 1
	DIVISION	DEPARTMENT	LEVEL		SALES
LOWEST LEVEL TOTAL LINE	→13	GROUP 1	5000		17,000.00 *
LOWEST LEVEL TOTAL LINE	→14	GROUP 2	4000		15,000.00 *

Figure 19-30. Examples of Entries on T-\*AUTO Field Description (Blank or B in Column 39 and Field Name in Columns 32 through 37)

### 19.4.2.2. Field Description that Prints a Heading (Blank in Column 39 and Heading in Columns 45 through 70)

You use this field description to print a heading (constant) on the lowest level total line. There is no column heading over this heading.

Figure 19-31 illustrates the T-\*AUTO field description that has a blank in column 39 and a heading in columns 45 through 70.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK OR INDICATORS	BLANK	BLANK	BLANK	BLANK OR END POSITION	BLANK	HEADING

Figure 19-31. T-\*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)

#### 19.4.2.2.1. Output Indicators - Fields (Columns 23 through 31)

You use this field to specify output indicators that condition the printing of the heading (constant) you specified in columns 45 through 70. The heading is printed on the lowest level total line each time the line is printed.

If you don't want to specify an output indicator, leave these columns blank. Auto report prints the heading on the lowest level total line.

If you want to specify an output indicator to condition the printing of the heading, enter it in these columns. See 8.2.8 for information about output indicators.



	01/15/80	COMPANY REPORT			PAGE 1
		DIVISION	DEPARTMENT	LEVEL	SALES
LOWEST LEVEL TOTAL LINE	FIRST GROUP	_____	_____	_____	17,000.00 *
LOWEST LEVEL TOTAL LINE	FIRST GROUP	_____	_____	_____	15,000.00 *

Figure 19-32. Examples of Entries on T-\*AUTO Field Description (Blank in Column 39 and Heading in Columns 45 through 70)

**19.4.2.3. Field Description that Prints a Numeric Field and Column Heading and Accumulates Totals (A in Column 39)**

You use this field description to print a numeric field on the lowest level total line and to accumulate totals for it on higher level total lines. You can optionally print a column heading over the numeric field.

A total is printed for each control level (L1 through L9) you defined in columns 59 and 60 of the input specifications, and a final total is printed.

Figure 19-33 illustrates the T-\*AUTO field description that has an A in column 39.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	FIELD NAME	BLANK OR EDIT CODE	A	BLANK OR END POSITION	BLANK	BLANK OR COLUMN HEADING

Figure 19-33. T-\*AUTO Field Description (A in Column 39)

Auto report generates calculations to accumulate totals for the fields you specify in columns 32 through 37 and places the calculations in the following order:

1. EXSR statement for the subroutine named A\$\$\$SUM generated by auto report

The EXSR statement and all ADD statements are unconditioned.

## 2. Total calculations generated by auto report

They are sorted by level starting with L0. These total calculations accumulate totals for the fields you name on T-\*AUTO specifications that have an A in column 39. Auto report generates a Z-ADD calculation conditioned by L0 that resets each of the accumulated total fields on the lowest level total to zero after they are printed. This prevents the same value from being accumulated more than once. These calculations are the first total calculations in the generated source program.

3. Total calculations you specify
4. Subroutines you specify
5. RPG II subroutine named A\$\$\$SUM generated by auto report that accumulates the lowest level total

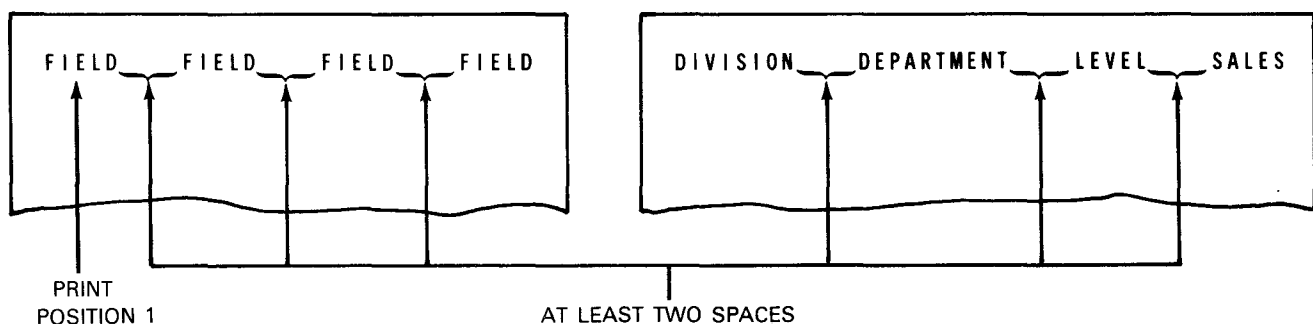
### 19.4.2.3.1. Output Indicators - Fields (Columns 23 through 31)

You can't use this field when column 39 contains an A.

### 19.4.2.3.2. Field Name (Columns 32 through 37)

You use this field to specify the name of a total numeric field that is accumulated and printed on the generated total line.

When you specify more than one name, the fields are printed from left to right on a line in the order you specify. At least two spaces are inserted between each field on the line. The first field, however, begins in print position 1:



Auto report generates an RPG II subroutine named A\$\$SUM that creates and names additional totaling fields based on the field name you specify to generate calculation and output specifications that accumulate and print the various levels of totals required.

If you want to accumulate a numeric field on the total line, enter the name of the field in columns 32 through 37 and an A in column 39. The associated field is printed on the lowest level total lines and totals for it are printed on higher level total lines. You can't identify an array, array element, or table.

You can specify totaling for any particular field only once in each program.

→ Pay attention to field names that end with a number from 1 through 9 or an R in an auto report program that accumulates totals, because auto report generates total fields that end in those characters.

If the field name you specify has fewer than six characters, any character from 1 through 9 or an R is added to the name to create a name for the total field. This added character corresponds to the total indicators L1 through L9 and LR, respectively. For example, if you specify a field name of GROUP and all nine control levels are defined, the generated field names are GROUP1, GROUP2,...GROUP9, and GROUPLR.

If the field name you specify has six characters, the last character is replaced by one of the characters from 1 through 9 or an R. For example, if you specify a field name of SUPPLY and all nine control levels are defined, the generated field names are SUPPL1, SUPPL2,..., SUPPL9, and SUPPLR.

You can't use any one field name more than once if you use an A in column 39. Also, if you specify a 5- or 6-character field name with an A in column 39, you can't specify a second 5- or 6-character field name that has the same first five characters with an A in column 39. For example, if you specify the following three field names with an A in column 39, all but the first are invalid:

GROUP	Valid
GROUPW	Invalid because the first five characters duplicate the first five characters of the first field
GROUP	Invalid because it is a duplicate of the first field

#### 19.4.2.3.3. Edit Codes (Column 38)

You use this field to enter an edit code for the numeric field you named in columns 32 through 37 as well as for all the generated total fields.

If you don't want to enter an edit code, leave this column blank. Auto report generates a K edit code, which prints a numeric field with commas and a decimal point where needed (for example, 2,123.77). The K edit code suppresses zeros so that zero balances are not printed and prints negative balances with a minus sign on the right.

If you want to enter an edit code, enter it in this column. See 8.3.3 for information about edit codes.

#### 19.4.2.3.4. Accumulate Totals (Column 39)

You use this field to indicate that you want to accumulate and print totals for the numeric field you named in columns 32 through 37. You can specify totaling only once for any particular field in each program.

If you want to accumulate and print totals for the numeric field you named in columns 32 through 37, enter an A. Auto report generates a subroutine named A\$\$SUM that accumulates the values from the fields into the lowest level total lines.

Auto report generates a B (blank after) in column 39 of all the detail and total field descriptions generated for the field name you specify. Thus, the value in the field you specified (and in any generated field) is reset to zero after the field value is printed.

Total fields are generated and named for all control level indicators you defined in columns 59 and 60 of the input specifications. Total calculations accumulate totals for the field you named in columns 32 through 37. Total calculations roll (add) the total from the lowest level total field to that of the next higher level field and end with a final total. This process is called total rolling.

For example, if you assign L1 and L3 to control fields on the input specification and specify the field PRD in columns 32 through 37 of the field description, three total fields (PRD1, PRD3, and PRDR) are generated and named by auto report. All total fields generated for the same level, such as PRD1 and SUPPL1 are printed on the same total line, and that line is conditioned by the corresponding control level indicator.

Generated total fields are two digits longer than the original field but the number of decimal positions remain the same in the generated fields. For example, if you use an edit code to define a field with a length of four characters (xx.xx), the length of each generated field is six characters (x,xxx.xx).

If the generated field is identical to a previously defined numeric field, the generated field is assigned the previously defined length and number of decimal positions (if the previously defined field is numeric).

Auto report prints asterisks to the right of generated total lines so you can tell they are total lines. One asterisk is printed to the right of the lowest level total line and two asterisks are printed to the right of the next level total and so on. For example, if you defined L1 and L3 control level indicators in a program, one asterisk is printed to the right of the L1 line, two asterisks are printed on the L3 line, and three asterisks are printed on the LR line. As many as 10 asterisks are printed on the LR line if you define all 9 control level indicators in the program.

To suppress the generation of asterisks on total lines, enter an N in column 28 of the auto report options specifications form.

Auto report generates a Z-ADD calculation conditioned by L0 that resets each of the accumulated total fields on the lowest level total line to zero after they are printed.

#### 19.4.2.3.5. End Position in Output Record (Columns 40 through 43)

You use this field to specify the end print position of the rightmost character of the field that is printed on the generated total line.

If you want auto report to generate end positions for fields and to center column headings, leave these columns blank.

If you want to specify the end position of the rightmost character of the field, enter an end position. Specify end positions only to eliminate the automatic spacing between fields or to spread out a report on the page. The end position you specify may be changed a little by auto report when the line is centered or when the column heading and field are positioned in relation to each other. If the end position you specify causes an overlay with another field, auto report generates a new end position.

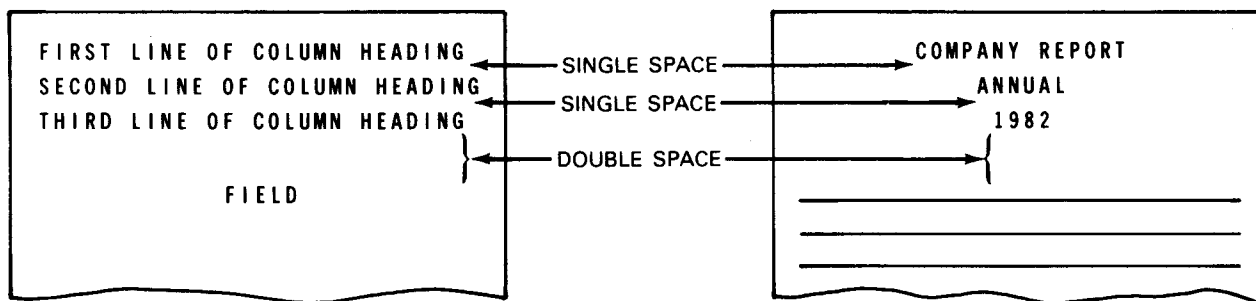
#### 19.4.2.3.6. First Column Heading Line (Columns 45 through 70)

You use this field to print the first line of a column heading (constant) over the generated total field you named in columns 32 through 37.

If you don't want a column heading, leave these columns blank.

If you want to print a column heading, enter it in these columns. You must enclose it within apostrophes. The value is printed over the accumulated total of the field you specified in columns 32 through 37.

When you specify a column heading, auto report automatically uses a double space after each single column heading line. If you specify a second or third line of column heading (C in column 39), auto report uses a single space after each column heading line except the last, which is followed by a double space:

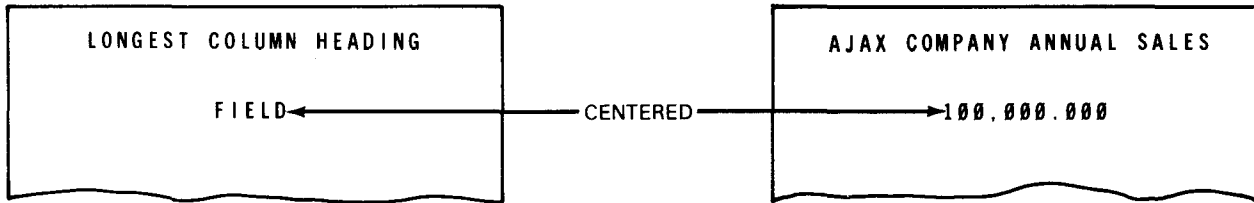


You can't specify any other spacing for column heading lines.

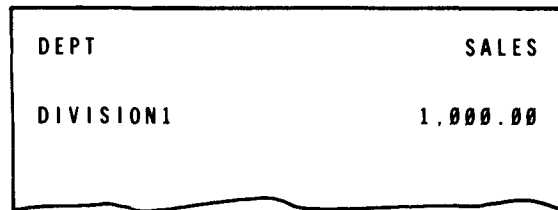
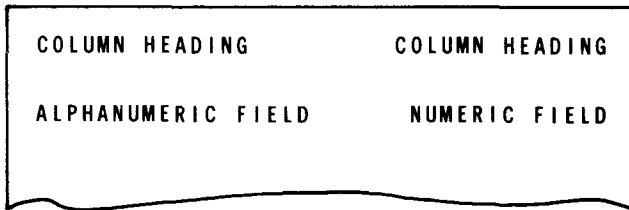
When you use a column heading, its placement above the field you specified in columns 32 through 37 depends on which is longer, the column heading or the field (including edit characters).

When the column heading is longer than its associated field, the field is centered under the longest column heading constant:



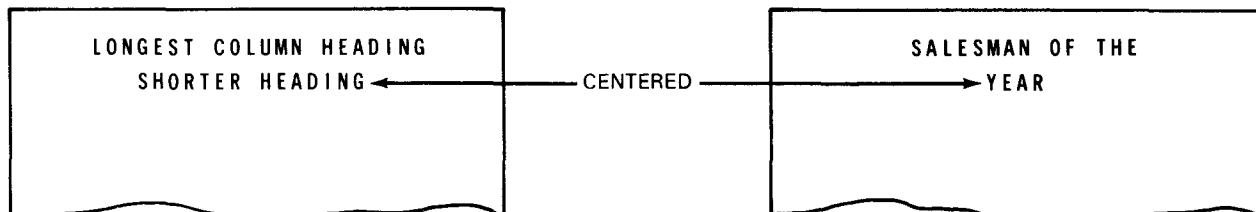


When the column heading is shorter than its associated field, the column heading is left-justified over an alphanumeric field and right-justified over a numeric field:



When you specify more than one column heading, these headings are printed from left to right on a line in the order you specify. No spaces are provided within a column heading – you must incorporate spaces within the column heading to provide for additional spacing.

You can use column heading continuation lines (C in column 39) to specify a second or third line of a column heading. When you use a multiple line column heading, the shorter column headings are centered under the longest column heading:



If the column heading line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess column headings are printed on the overflow line in the order you specified and are right-justified.

If you want to specify your own end position for column headings and their associated fields, enter the end position in columns 40 through 43 of the T-\*AUTO specification.

**19.4.2.3.7. Examples of Entries on T-\*AUTO Field Description (A in Column 39)**

Figure 19-34 illustrates a report that prints a numeric field on the lowest level total line and then accumulates totals for it on higher level total lines. A final total is also printed. A column heading is printed over the numeric field.



RPG II  
OUTPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES

PAGE NO.	LINE NO.	FORM TYPE	STACKER SELECT / F-FETCH OVERFLOW TYPE	SPACE			SKIP			OUTPUT INDICATORS			FIELD NAME	DATA FORMAT P/B/L/R	CODES			COMMAS			ZERO BALANCE TO PRINT		CODES ACTION			NOT USED	PROGRAM IDENTIFICATION
				BEFORE	AFTER	BEFORE	AFTER	BEFORE	AFTER	BEFORE	AFTER	N-NOT			N-NOT	N-NOT	1	2	3	4	CR	INSER	YES	NO	X		
1	0,1	O	RE.DPRT.	H									*AUTO														
	0,2	O																									
	0,3	O	RE.DPRT.	T									*AUTO														
	0,4	O											DIV.														
	0,5	O											DEPT.														
	0,6	O											LEV.														
	0,7	O											SAL.	A													
	0,8	O																									

	01/15/80		COMPANY REPORT		PAGE 1
	DIVISION	DEPARTMENT	LEVEL		SALES
LOWEST LEVEL TOTAL LINE	_____	_____	_____		17,000.00 *
LOWEST LEVEL TOTAL LINE	_____	_____	_____		15,000.00 *
FINAL TOTAL					32,000.00 **

Figure 19-34. Examples of Entries on T-\*AUTO Field Description (A in Column 39)

19.4.2.4. Field Description that Prints Second and Third Column Heading Line (C in Column 39)

You use this field description to print a second or third line of a column heading. You can use one or two C specifications following a field description that has an A, B, or blank in column 39 and a field name in columns 32 through 37.

Figure 19-35 illustrates the T-\*AUTO field description that has a C in column 39.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	BLANK	BLANK	C	BLANK	BLANK	COLUMN HEADING

Figure 19-35. T-\*AUTO Field Description (C in Column 39)

**19.4.2.4.1. Column Heading Continuation Lines (Column 39)**

You use this field to indicate that you want a second and third column heading line if there is more information in a column heading than can be contained on one line.

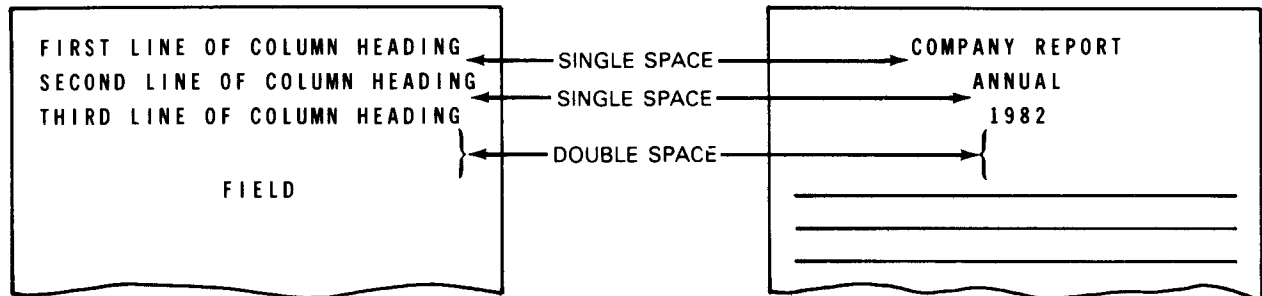
If you want to specify continuation lines for a column heading, enter a C. The column heading you specify in columns 45 through 70 is printed on the second or third line of the column heading.

**19.4.2.4.2. Second and Third Column Heading Line (Columns 45 through 70)**

You use this field to print a second or third column heading line. If you don't want to specify a second or third column heading line, leave these columns blank.

If you want to specify continuation lines for a column heading you specified on a field description with a B, blank, or A in column 39, enter them in these columns. If this is the first C specification, you are specifying the second line of the column heading. If this is the second C specification, you are specifying the third line of the column heading. You can't use more than two C specifications. You can specify column headings continuation lines that are up to 24 characters long, including blanks. You must enclose them within apostrophes. Short column headings are centered under the longest column headings.

Auto report uses a single space after each column heading line except the last, which is followed by a double space:



19.4.2.4.3. Examples of Entries on T-\*AUTO Field Description (C in Column 39)

Figure 19-36 illustrates a report that contains multiple-line column headings.

RPG II  
OUTPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW		SPACE	SKIP	OUTPUT INDICATORS						FIELD NAME	DATA FORMAT P/B/L/R	CODES				NOT USED	PROGRAM IDENTIFICATION																	
			TYPE	H/D/T/E			BEFORE	AFTER	BEFORE	AFTER	N.IND	N.IND			N.IND	N.IND	NEGATIVE VALUE INDICATION	COMMAS INSERTED			ZERO BALANCE TO PRINT	CODES	ACTION														
1	2	3	4	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45	70	71	74	75	80
0,1	O		R	E	P	O	R	T		H																											
0,2	O																																				
0,3	O		R	E	P	O	R	T		T																											
0,4	O																																				
0,5	O																																				
0,6	O																																				
0,7	O																																				
0,8	O																																				
0,9	O																																				
1,0	O																																				
1,1	O																																				

01/15/80	COMPANY REPORT			PAGE 1
DIVISION	DEPARTMENT	LEVEL	SALES	
	STORE		FOR THE	
			YEAR	
_____	_____	_____	17,000.00 *	
_____	_____	_____	15,000.00 *	

Figure 19-36. Examples of Entries on T-\*AUTO Field Description (C in Column 39)

19.4.2.5. Field Description that Prints a Heading next to a Total (1 through 9 or R in Column 39)

You use this field description to print a heading (constant) on a specific total line generated from an A in column 39. You can only use this field description if you used a field description with an A in column 39. The heading you enter is printed on the total line that corresponds to the level (L1 through L9 or LR) you specified in column 39.

Figure 19-37 illustrates the T-\*AUTO field description that has a number from 1 through 9 or an R in column 39.

COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	BLANK	BLANK	1-9, R	BLANK	BLANK	HEADING

Figure 19-37. T-\*AUTO Field Description (1 through 9 or R in Column 39)

#### 19.4.2.5.1. Specific Total Line (Column 39)

You use this field to identify a specific total line on which the heading (constant) is printed. The heading is printed to the left of the leftmost total on the line.

If you didn't define any of the field descriptions with an A in column 39, you can't use an entry in this column.

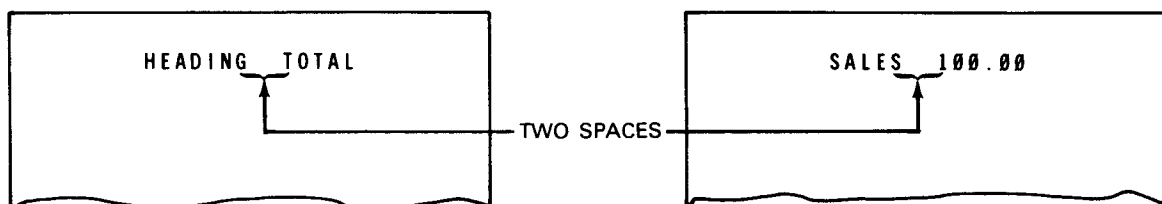
If you want to print the heading on a specific total line, enter a number from 1 through 9 or an R. This entry corresponds to the indicators L1 through L9, which you defined in columns 59 and 60 of the input specifications, and LR. For example, 4 in column 39 indicates the information is printed on the L4 total line and an R in column 39 indicates the information is printed on the final total (LR) line.

Your entry must be higher than the lowest control level indicator you used in columns 23 through 31 of the T-\*AUTO output file identification. If you didn't specify control level indicators on the T-\*AUTO output file identification, use only entries that are higher than the lowest control level indicators you defined in columns 59 and 60 of the input specifications.

#### 19.4.2.5.2. Heading (Columns 45 through 70)

You use this field to print a heading (constant) on a specific total line. You must enclose the heading within apostrophes.

The heading that will appear next to a total is printed to the left of the first total. It is separated from the total by two spaces. If you specify two or more headings (or fields) to appear next to a total, they are printed in the order you specified:



19.4.2.5.3. Examples of Entries on T-\*AUTO Field Description (1 through 9 or R in Column 39)

Figure 19-38 illustrates a report that contains a heading (constant) printed next to the final total.

RPG II  
OUTPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW TYPE H/D/T/E	SPACE		SKIP		OUTPUT INDICATORS				FIELD NAME	DATA FORMAT P/B/L/R	CODES			ACTION			NOT USED	PROGRAM IDENTIFICATION	
				BEFORE	AFTER	BEFORE	AFTER	AND	AND	AND	AND			AND	AND	1	2	3	4			5
1	0	0,1	R									*AUTO										
		0,2																				
		0,3	R									*AUTO										
		0,4										DIV.										
		0,5										DEPT.										
		0,6										LEV.										
		0,7										SAL.	A									
		0,8											R									
		0,9																				

	01/15/80	COMPANY REPORT	PAGE 1
	DIVISION	DEPARTMENT	LEVEL
			SALES
LOWEST LEVEL TOTAL LINE	_____	_____	17,000.00 *
LOWEST LEVEL TOTAL LINE	_____	_____	15,000.00 *
FINAL TOTAL	GROUP SALES		32,000.00 **

HEADING

Figure 19-38. Examples of Entries on T-\*AUTO Field Description (1 through 9 or R in Column 39)

**19.4.2.6. Field Description that Prints a Field next to a Total (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)**

You use this field description to print an alphanumeric or numeric field on a specific total line generated from an A in column 39. You can only use this field description if you used a field description with an A in column 39. The field you enter is printed on the total line that corresponds to the level (L1 through L9 or LR) you specified in column 39.

Figure 19-39 illustrates the T-\*AUTO field description that has a number from 1 through 9 or an R in column 39 and a field name in columns 32 through 37.

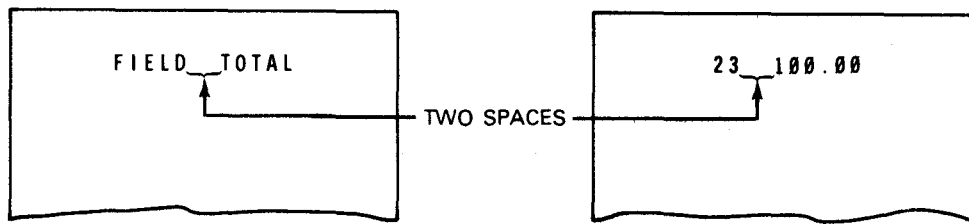
COLS:	7-22	23-31	32-37	38	39	40-43	44	45-70
	BLANK	BLANK	FIELD NAME	BLANK OR EDIT CODE	1-9 OR R	BLANK	BLANK	BLANK OR EDIT WORD

Figure 19-39. T-\*AUTO Field Description (1 through 9 or R in Column 39 and Field Name in Columns 32 through 37)

**19.4.2.6.1. Field Name (Columns 32 through 37)**

You use this field to specify the name of a field, an indexed array, or a table. The associated field, array, or table is printed on a specific total line.

When you specify a field to appear next to a total, it is printed to the left of the first total. It is separated from the total by two spaces:



If you specify two or more fields (or headings) to appear next to a total, they are printed in the order you specified. Each field is separated by one space.

#### 19.4.2.6.2. Edit Codes (Column 38)

You use this field to enter an edit code for a numeric field, indexed array, or table you named in columns 32 through 37.

If you specified an alphanumeric field, table, or indexed array in columns 32 through 37, you must leave this column blank. Also, leave this column blank if you don't want to use an edit code.

If you named a numeric field, indexed array, or table in columns 32 through 37, you can enter an edit code. When you specify an edit code, you must leave columns 45 through 70 blank.

#### 19.4.2.6.3. Specific Total Line (Column 39)

You use this field to identify a specific total line on which the field you described in columns 32 through 37 is printed. The field is printed to the left of the leftmost total on the line.

If you didn't define any of the field descriptions with an A in column 39, you can't use an entry in this column.

To print the associated field on a specific total line, enter a number from 1 through 9 or an R. This entry corresponds to the indicators L1 through L9, which you defined in columns 59 and 60 of the input specifications, and LR. For example, 4 in column 39 indicates the information is printed on the L4 total line, and R in column 39 indicates the information is printed on the final total (LR) line.

Your entry must be higher than the lowest control level indicator you used in columns 23 through 31 of the T-\*AUTO output file identification. If you didn't specify control level indicators on the T-\*AUTO output file identification, use only entries that are higher than the lowest control level indicators you defined in columns 59 and 60 of the input specifications.

#### 19.4.2.6.4. Edit Word (Columns 45 through 70)

You use this field to edit numeric fields that you want printed on a specific total line.

If you specified an edit code in column 38, you must leave these columns blank. Also, leave these columns blank if you don't want to specify an edit word.

If you want to edit the field, enter an edit word. See 8.3.8 for information about edit words.









## 19.5. AUTO REPORT OPTIONS SPECIFICATIONS FORM

You use the auto report options specifications form (Figure 19-42) to specify the options you want to use with the output from auto report.

These options include:

- Cataloging the generated source program into a library file
- Suppressing the generated date and page number so that they aren't printed
- Suppressing asterisks so that they aren't printed next to the generated totals
- ➔ ■ Suppressing listings or obtaining only diagnostics

The auto report options specifications form is not required. If you don't use this form, the options that correspond to the blank entries on the auto report options specifications form become the default values.

If you use this form, it must be the first form in the program.

You can't place auto report options specifications in a library file module that is copied by a /COPY statement.

If you didn't include control specifications (H in column 6) either in the auto report source program or in a library file module that is copied by a /COPY statement, auto report generates a control specification with blank entries that provides information about the RPG II program and describes the system requirements to the RPG II compiler. See Section 4 for an explanation of the blank entries for the control specification.

### 19.5.1. Form Entries

In the following subsections, you'll see what the fields on the form are used for and how the entries in these fields affect your program.

#### 19.5.1.1. Catalog Source Program in Library File (Column 7)

You use this field to indicate whether or not you want the generated source program to be written in a library file. In addition, the generated source program is also written in the system resident \$Y\$SRC disk file using the specified module name.

If you want to write the generated source program only into the system resident \$Y\$SRC disk file using the module name RPGSRC, leave this column blank.



If you want to catalog the generated source program in a library file on disk, enter a C. This cataloged source program becomes a permanent library file module. If, however, you catalog a new module with the same name as an old module, the new module replaces the old one. If your auto report specifications contain fatal errors, the generated source program isn't cataloged.

#### 19.5.1.2. Library File and Module Name (Columns 8 through 24)

You use this field if you decide to catalog the generated source program in a library file (C in column 7). You identify the disk on which the library file resides and the name under which the generated source program is cataloged.

To identify the disk on which the library file resides, enter F1 (system library) or the name you specified on the LFD job control statement followed by a comma. Then, specify the name under which the source program is cataloged.

[ LFDname ], module name

The LFD name and the module name can be up to eight characters in length and must begin with an alphabetic character from A through Z or one of these special characters: #, \$, or @. The remainder of the name may be any combination of alphabetic, numeric, or special characters.

If you don't use an LFD name, the default is F1. If you don't enter a module name, an error occurs.

If you catalog a new module with the same name as an old module, the new module replaces the old one.

If you don't use columns 8 through 24 on the auto report options specifications form, the generated source is always output to element RPGSRC and file \$Y\$SRC if there are no serious errors.

#### 19.5.1.3. Date Suppress (Column 27)

You use this field to suppress the generated date and page number so that they aren't printed on the first H-\*AUTO page heading line.

If you want the page number and date printed on the first H-\*AUTO page heading line, leave this column blank.

To suppress the page number and date printed on the first H-\*AUTO page heading line, enter an N. With the date and page number suppressed, you can have a title that takes up the entire line.

#### 19.5.1.4. Asterisks Suppress (Column 28)

You use this field to suppress asterisks so that they aren't printed next to totals generated by D-\*AUTO or T-\*AUTO (A in column 39).

If you want to print asterisks next to generated totals, leave this column blank. The asterisks are printed to the right of the highest end position on a generated total line as specified in the D-\*AUTO or T-\*AUTO specifications.

If you don't want to print asterisks next to generated totals, enter an N.

#### 19.5.1.5. List Options (Column 30)

You use this to control an auto report listing. A complete listing contains:

- Original source listing

Auto report prints a listing of your source program as it is retrieved. This listing is the base for the error message listing.

- Generated source listing

Auto report prints a listing of the source program that is generated by auto report and the specifications copied from a library file.

- Auto report error message listing

Auto report prints a listing of diagnosed errors from the original source listing. See Appendix G for an explanation of the auto report error messages.

If you want to print the complete source listing along with headings and diagnostics, leave this column blank.

If you don't want to print the complete source listing, enter a B. Use a B if you want to produce a source program that already has a listing.

If you want to print only a partial source listing, enter a P. This partial listing includes headings and diagnostics. Use P to see if small modifications to a program you already tested have produced any errors.

Auto report prints the original specifications and specifications copied from a library file in the following format:

- Columns 1 through 4

Sequence number of the specification. This number starts with 0001 on the RPG II control specification and is incremented by 1 on each specification that follows (e.g., 0001, 0002, 0003, etc). If more than 999 specifications are present in the program, the sequence is restarted at 0000.

↓  
■ Column 5

Code that identifies the specification

- Blank

Standard RPG II specifications present in the auto report program

- C

Specifications copied from a library file by the /COPY statement

## ■ Columns 6 through 80

Standard RPG II specifications

Compile-time tables and arrays are not changed by auto report; they remain in standard table/array record format.

Auto report prints the generated source listing in the following format:

## ■ Columns 1 through 4

Sequence number of the specification. This number starts with 0010 on the RPG II control specification and is incremented by 10 on each specification that follows (e.g., 0010, 0020, 0030, etc). If more than 999 specifications are present in the program, the sequence is restarted at 0000.

## ■ Column 5

Code that identifies the specification

- Blank

Standard RPG II specifications present in the auto report program

- C

Specifications copied from a library file by the /COPY statement

- E

Specifications generated by H-\*AUTO, D-\*AUTO, or T-\*AUTO specifications

- M

File description and input specifications copied from a library file and modified by /COPY modifier statements.

↑



- Columns 6 through 74

Standard RPG II specifications

- Columns 75 through 80

Same characters that are present in columns 75 through 80 of the RPG II control specification. (If these are blank on the control specification, they are also blank on all specifications in the generated RPG II program.)

Compile-time tables and arrays are not changed by auto report; they remain in standard table/array record format.

### 19.5.2. Examples of Entries on the Auto Report Options Specifications Form

Figure 19-43 catalogs the generated source program into the module named LIB on the F1 disk. A complete source listing is printed.

PAGE NO.	LINE NO.	FORM TYPE	CATALOG SOURCE	LIBRARY FILE AND MODULE NAME LFD name, module name	NOT USED	N OR BLANK	N OR BLANK	NOT USED	BIP OR BLANK	DATE SUPPRESS		NOT USED	PROGRAM IDENTIFICATION									
										ASTERISK SUPPRESS	LIST OPTIONS											
1	2	3	5	6	7	8	24	25	26	27	28	29	30	31	40	50	60	70	74	75	80	
			U	C,F1,1,LIB																		

Figure 19-43. Examples of Entries on Auto Report Options Specifications Form

### 19.6. /COPY STATEMENT

You use the /COPY statement to copy cataloged RPG II specifications from a library file (MIRAM or SAT) and insert them into an RPG II source program. This statement identifies the file that contains the copy module that contains the copied RPG II specifications.

By using the /COPY statement, you can copy source specifications and include them in several different programs. In this way, you don't need to repeat code specifications that are used in different programs.

You can copy auto report specifications and any valid RPG II specifications including tables and arrays in this manner. You can't, however, copy the auto report option specifications and other /COPY statements. You must place /COPY statements after the auto report options specifications, and they must precede source tables and arrays.

The specifications you include in an auto report program by the /COPY statement are initially placed in the program immediately after the /COPY statement. After all the specifications are copied from the library file, the entire auto report program is sorted into the following order:

1. Control specifications
2. File description specifications
3. Extension specifications
4. Line counter specifications
5. Telecommunications specifications
6. Input specifications
7. Calculation specifications in the order: detail, L0, L1 through L9, LR, and subroutines
8. Output specifications
9. Tables and arrays loaded at compilation time, which are placed last among the input statements to auto report.

After this sorting, auto report generates RPG II calculations and output specifications from the H-\*AUTO, D-\*AUTO, or T-\*AUTO specifications. These generated specifications are then sorted.

### **19.6.1. Form Entries**

In the following subsections, you'll see how the fields on the /COPY statement are used and how the entries in these fields affect your program.

#### **19.6.1.1. Form Type (Column 6)**

This field identifies the specification that contains the /COPY statement.

Use any specification except control (H) or auto report options (U).

#### **19.6.1.2. /COPY (Columns 7 through 11)**

You use this field to indicate that you want to copy cataloged RPG II specifications from a library file and insert them into an RPG II source program.

Enter /COPY in these columns.

#### **19.6.1.3. Library File and Module Name (Columns 13 through 29)**

→ You use this field to identify the file that contains the copy module that contains the cataloged RPG II specifications.



### 19.6.3.1. /COPY Modifier Statements for File Description Specifications

You can include modifier statements with the /COPY statement to modify, add, or delete entries on the file description specifications as they are copied from the library file.

#### 19.6.3.1.1. Form Type (Column 6)

You use this field to indicate that you want to modify a file description specification.

You enter your modifier statements on a file description specification (F) because the modifier statements will immediately follow it and can be put on the same specification.

#### 19.6.3.1.2. File Name (Columns 7 through 13)

You use this field to specify the name of the file description specification that you want to modify.

Enter the name of the file. You can use only one file description specification with a particular file name from the library file entries and you can use a particular file name only once in a modifier statement.

**NOTE:**

→ *The file name can be eight characters long.*

#### 19.6.3.1.3. Modifying File Description Entries (Columns 15 through 80)

You use this field to modify or add entries on a file description specification that is copied from a library file.

To make modifications, enter the file name in columns 7 through 13, and then in columns 15 through 80 make only those entries on the line that are to replace existing entries in the copied specification or that are to be included as new entries. Blank entries in the modifier statement don't affect the copied statement.

For example, assume that you want to copy the file description specifications for a frequently used file named ORDERS from the system library. The original specification contains an I in file type (column 15), which defines ORDERS as an input file. If you want to update the ORDERS file, change column 15 to U by including a modifier file description specification in the auto report source program. You must include the file name ORDERS and the new file type entry U. The result of the modifier statement is that the file type on the copied file description specification is changed from I to U (see Figure 19-45).







Enter the name of the input field. The modifier statement modifies all copied input field specifications that have the same field name. If there is no input field by the same name, the modifier statement is added to the program as a new input field specification. You can use modifier statements that have the same field names (length and number of decimal positions must also be the same), but only the first is used to modify a copied specification. Other field names are added as new input field specifications.

#### **19.6.3.2.3. Modifying Input Format Entries (Columns 43 through 70)**

You use this field to modify an input field on the input format specification that describes individual fields on the input record.

The fields that you can modify are:

- Column 43 (packed/binary)
- Columns 44 through 51 (field location)
- Column 52 (decimal positions)
- Columns 59 and 60 (control levels)
- Columns 61 and 62 (matching or chaining fields)
- Columns 63 and 64 (field record relation)
- Columns 65 through 70 (field indicators)

The method of replacing, adding, or blanking entries is similar to the method used to modify file description specifications.

If you want to modify an input field specification copied from a library file, enter the field name in columns 53 through 58 of an input field specification (I in column 6). Then you make only those entries on the line that are to replace existing entries in the copied specification or that are to be included as new entries. Blank entries in the modifier statement don't affect the copied statement. You must place modifier statements for input field specifications immediately after the /COPY statement in the auto report program that copies those specifications.

For example, assume that you want to copy an input field specification for a frequently used file named ORDERS from the system library. The original specification doesn't contain the entries you need. You must include a /COPY statement and the modifications you want to make. The results of the modifier statement are that the fields on the input field specification are changed to the new values. (See Figure 19-47.)





The first specification that follows the /COPY statement that is not an input field specification is considered the end of the input field modifier statements for the /COPY statement. (A comment statement with an I in position 6 is not considered the end of the input field modifier statements.)

You can use up to 20 input field modifier statements per /COPY statement. For best results, first use those statements that modify existing input field specifications, then use those that are to be added as new input field specifications. Thus, input field modifier statements that don't fit into the special main storage table for modifier statements are added to the RPG II source program as new input field specifications. This order of specifying modifier statements increases the likelihood that excess statements, if any, will be valid field descriptions.

**19.6.3.2.4. Setting Input Format Entries to Blanks (Columns 43 through 70)**

If you want to set an input field entry to blanks, enter an ampersand (&) in the first position of that entry on the modifier statement, and leave the remaining positions blank. For example, if you want to remove the minus field indicator entry (columns 67 and 68) from a cataloged specification, add an ampersand to the modifier statement in column 67 and leave column 68 blank. (See Figure 19-48.)

RPG II  
INPUT FORMAT SPECIFICATIONS

PROGRAM \_\_\_\_\_ PROGRAMMER \_\_\_\_\_ DATE \_\_\_\_\_ PAGE \_\_\_\_\_ OF \_\_\_\_\_ PAGES

PAGE NO.	FORM TYPE	LINE NO.	FILE NAME	RECORD IDENTIFICATION												FIELD DESCRIPTION										NOT USED	PROGRAM IDENTIFICATION																									
				RECORD IDENTIFICATION CODES												FIELD LOCATION		FIELD NAME	CONTROL LEVEL	MATCHING FIELDS OR CHAINING FIELDS	FIELD RECORD RELATION	FIELD INDICATORS																														
				1	2	3	4	5	6	7	8	9	10	11	12	FROM	TO					PLUS	MINUS	ZERO OR BLANK																												
1	2	3	5	6	7	14	15	16	17	18	19	20	21	24	25	26	27	28	31	32	33	34	35	38	39	40	41	42	43	44	47	48	51	52	53	58	59	60	61	62	63	64	65	66	67	68	69	70	71	74	75	80
		0,1	1	ORDERS	A	A	A	A																																												
		0,2	1																																																	
		0,3	1																																																	
		0,4	1																																																	
		0,5	1																																																	

a. Input specification for the ORDERS file as it is cataloged in a library file

Figure 19-48. Setting Input Field Entries to Blanks (Part 1 of 2)



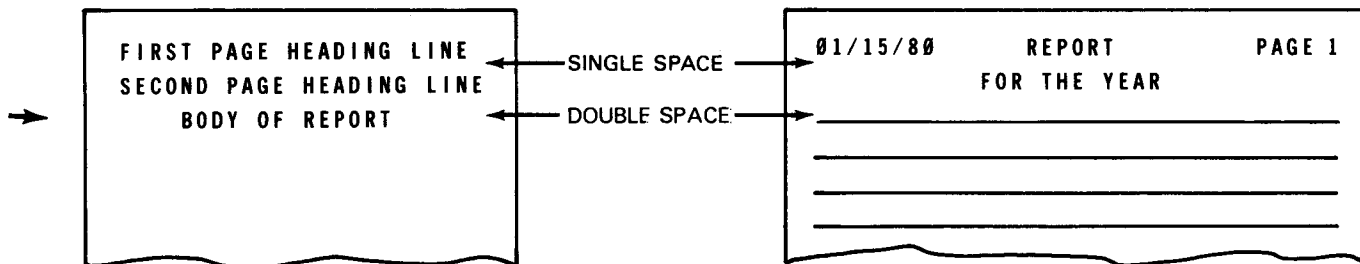
## 19.7. SUMMARY OF HOW AUTO REPORT FORMATS YOUR REPORTS

One of the advantages of auto report is that it frees you from specifying the format of your report on the output specifications. Auto report can automatically format your printed report by spacing, skipping, centering lines, and calculating end positions for fields and headings.

If you don't want auto report to automatically format your report, you must specify the appropriate entries on the H-\*AUTO, D-\*AUTO, or T-\*AUTO specifications.

### 19.7.1. Spacing for H-\*AUTO Page Heading Lines

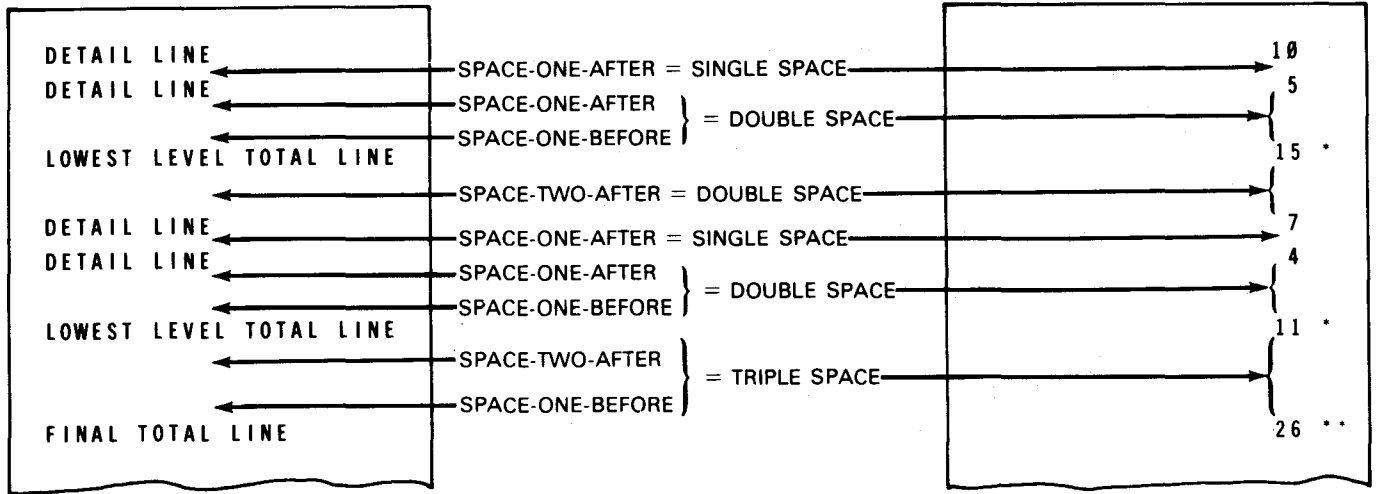
If you leave columns 17 and 18 blank on an H-\*AUTO output file identification, auto report uses a double space after each single page heading line. If you use more than one page heading line, auto report uses a single space after each page heading line except the last, which is followed by a double space.



If you want to specify other spacing values, enter them in columns 17 and 18 on the H-\*AUTO specification.

### 19.7.2. Spacing for D-\*AUTO Lines

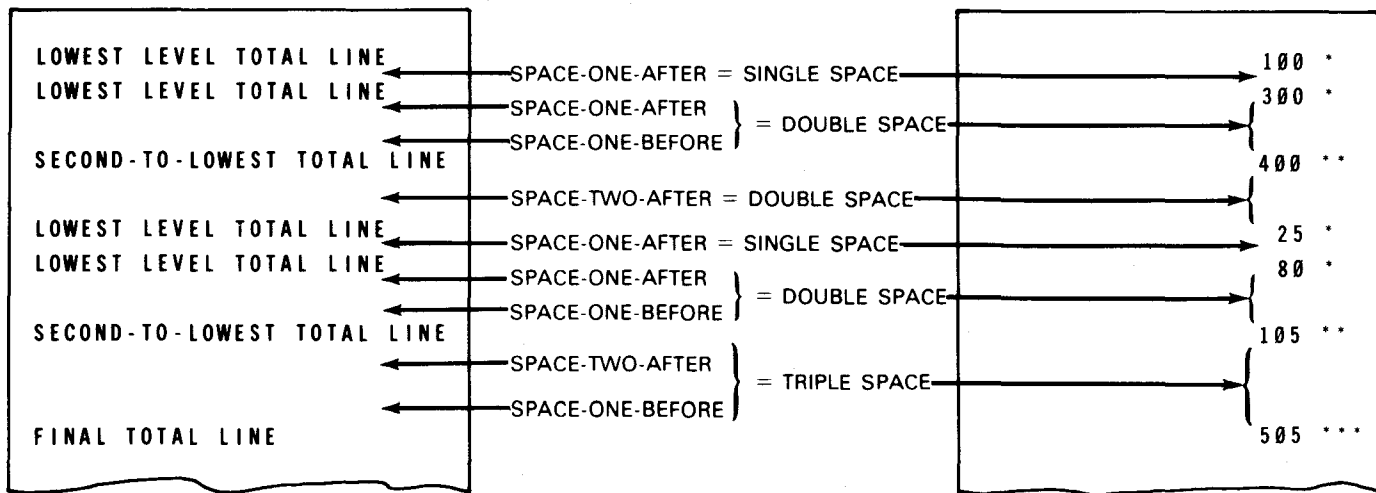
If you leave columns 17 and 18 blank on a D-\*AUTO output file identification, auto report uses a single space (space-one-after) after each detail line and a double space (space-two-after) after each total line. Auto report additionally generates one blank line before the lowest level total line and before the final total line (space-one-before).



If you want to specify other spacing values for detail lines, enter them in columns 17 and 18 on the D-\*AUTO specification. You can't specify any other spacing for total lines.

### 19.7.3. Spacing for T-\*AUTO Lines

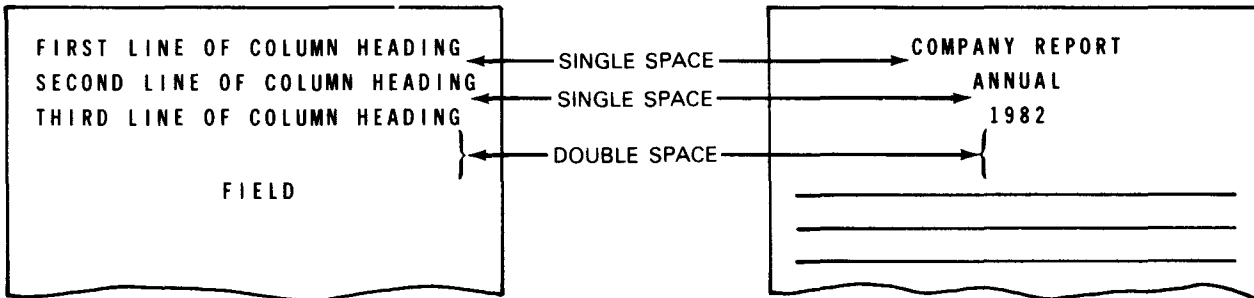
If you leave columns 17 and 18 blank on a T-\*AUTO output file identification, auto report uses a single space (space-one-after) after each lowest level total line and a double space (space-two-after) after each higher level total line. Auto report additionally generates one blank line before the second-to-the-lowest level total line and before the final total line (space-one-before).



If you want to specify other spacing values for the lowest level total lines, enter them in columns 17 and 18 on the T-\*AUTO specification. You can't specify any other spacing for higher level total lines.

**19.7.4. Spacing for D-\*AUTO or T-\*AUTO Column Heading Lines**

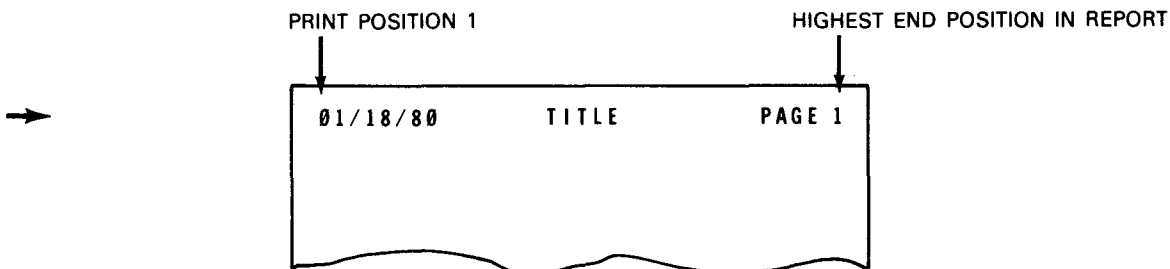
When you specify a column heading in columns 45 through 70 of a D-\*AUTO or T-\*AUTO field description, auto report uses a double space after each single column heading line. If you specify a second or third line of column heading, auto report uses a single space after each column heading line except the last, which is followed by a double space.



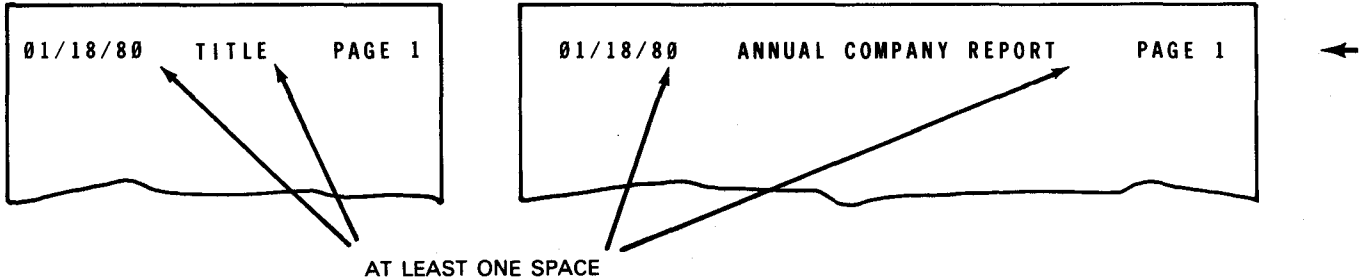
You can't specify any other spacing for column heading lines.

**19.7.5. Placement of H-\*AUTO Page Heading Lines**

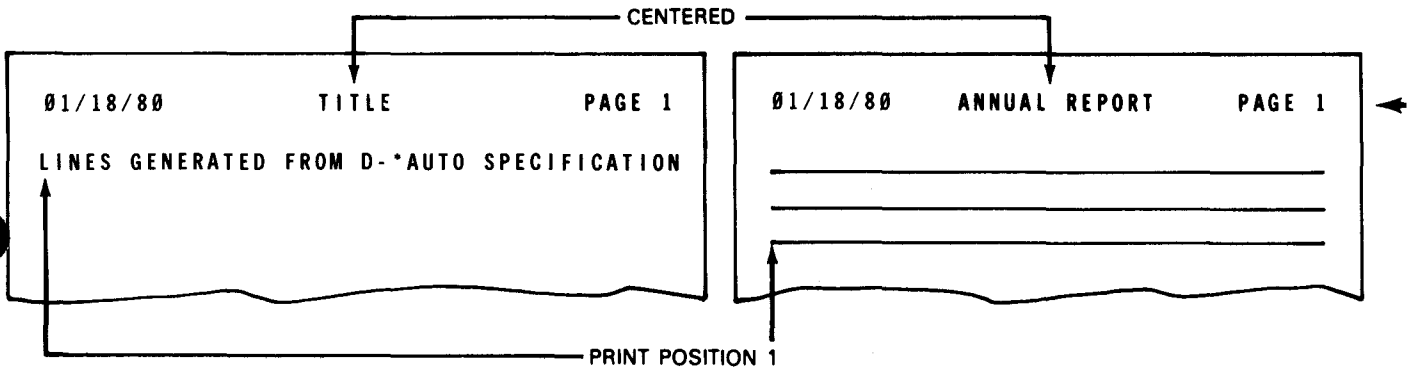
When you use H-\*AUTO to print a page heading, auto report automatically prints the date in print positions 1 through 8 and prints the page number with an end position equal to the highest end position of the longest line in the report:



The placement of the title depends on which is longer, the page heading or the lines generated by D-\*AUTO or T-\*AUTO specification. When the page heading is longer than the lines generated by D-\*AUTO or T-\*AUTO, one blank space separates the date from the title and the title from the word PAGE:



When the page heading is shorter than the lines generated by D-\*AUTO or T-\*AUTO, the D or T line begins in print position 1 and the title is centered above it:



When you specify more than one page heading, the shorter page headings are centered under the longest page heading.

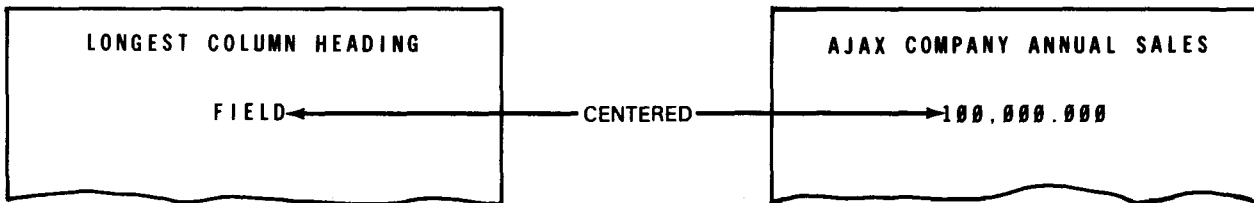
If the page heading line is longer than the record length you specified for the printer file, the characters on the right are truncated. There is no overflow line.

You can't specify any other placement for page heading lines.

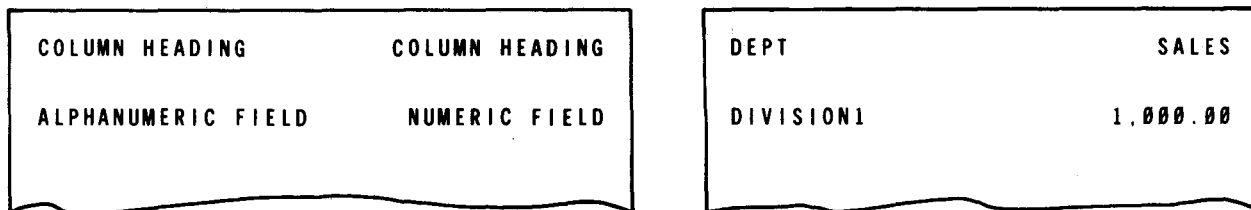
### 19.7.6. Placement of D-\*AUTO or T-\*AUTO Column Heading Lines over Fields

When you use D-\*AUTO, T-\*AUTO, or field description specifications to print a column heading over its associated field, its placement depends on whether the column heading or the field (including edit characters) is longer.

When the column heading is longer than its associated field, the field is centered under the longest column heading constant:

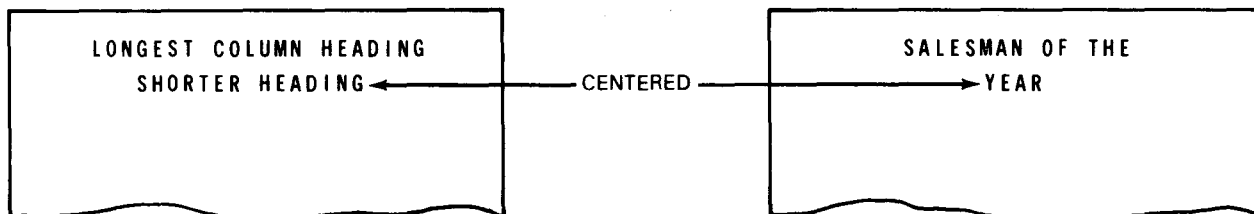


When the column heading is shorter than its associated field, the column heading is left-justified over an alphanumeric field and right-justified over a numeric field:



When you specify more than one column heading, they are printed from left to right on a line in the order you specify. No spaces are provided within a column heading - you must incorporate spaces within the column heading to provide for additional spacing.

You can use column heading continuation lines (C in column 39) to specify a second or third line of column heading. When you use a multiple line column heading, the shorter column headings are centered under the longest column heading:





If the column heading line is longer than the record length you specified for the printer file, an overflow print line is generated. The excess column headings are printed on the overflow line in the order you specified and are right-justified. (See Figure 19-49.)

If you want to specify your own end positions for the column headings and associated fields, enter the end position in columns 40 through 43 of the D-\*AUTO or T-\*AUTO specification.

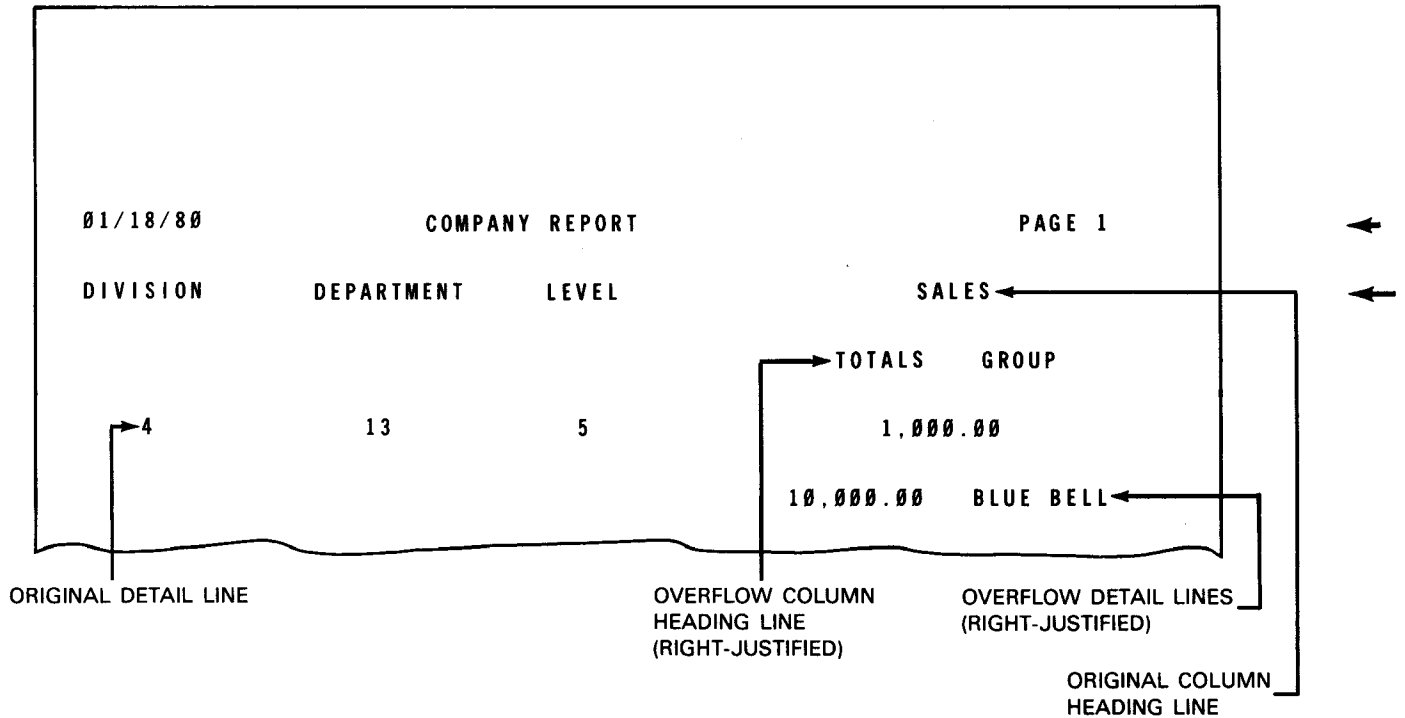
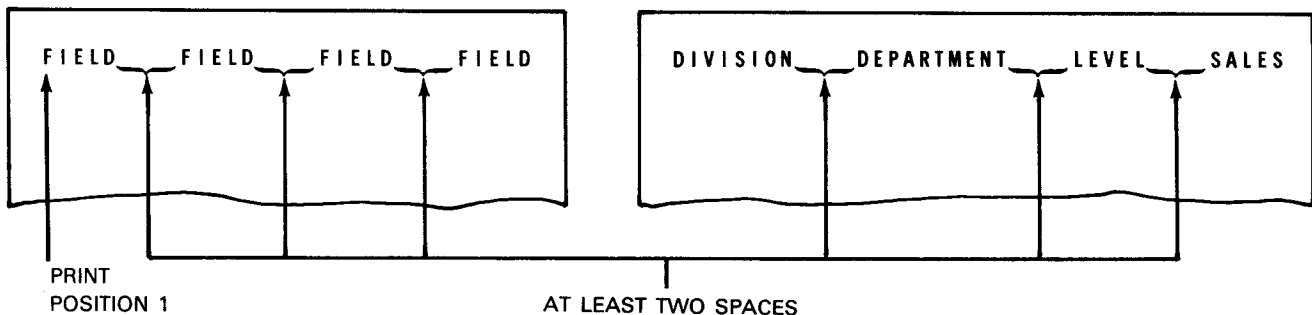


Figure 19-49. Overflow of D-\*AUTO or T-\*AUTO Lines

### 19.7.7. Placement of D-\*AUTO or T-\*AUTO Fields

When you use D-\*AUTO or T-\*AUTO to specify fields, they are printed in the order you specified. Auto report inserts at least two spaces between each field on the line. The first field, however, begins in print position 1:



If you want to specify your own end positions for fields, enter the end position in columns 40 through 43 of the D-\*AUTO or T-\*AUTO specification.

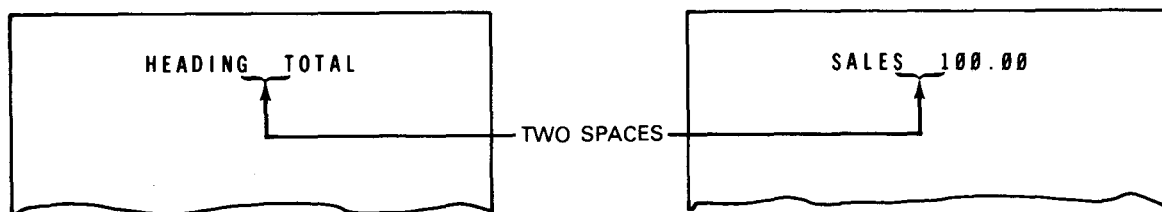
If the detail or total line is longer than the record length you specified for the printer file, an overflow print line is generated. (See Figure 19-49.) The excess information is printed on the overflow line in the order you specified and is right-justified. If you specified entries in the space fields (columns 17 and 18) and skip fields (columns 19 through 22), the spacing for the overflow line is as follows:

- The space-before and skip-before entries you specified are for the original line. Auto report uses a single space after the original line.
- The space-after and skip-after entries you specified are for the overflow line. Auto report generates blanks for space-before and skip-before for the overflow line.

#### 19.7.8. Placement of D-\*AUTO or T-\*AUTO Headings or Fields next to Totals

When you use D-\*AUTO or T-\*AUTO with a number from 1 through 9 or an R in column 39 to specify a heading to appear next to a total, it is printed to the left of the first total.

It is separated from the total by two spaces:



If you specify two or more headings or fields for a total, they are printed in the order you specified. Each field is separated by one space. Auto report doesn't provide any spacing within headings – you must incorporate spaces yourself.

#### 19.7.9. Skipping

If you leave columns 19 through 22 blank on an H-\*AUTO, D-\*AUTO, or T-\*AUTO output file identification, auto report skips the printer to home position before the first line is printed.

## 19.8. JOB CONTROL FOR AUTO REPORT

See 18.2.2 and 18.2.3 for information about job control statements used with auto report. ←

## 19.9. NON-ENGLISH LANGUAGE FEATURE

The non-English feature allows you to print diagnostics, listing headers, and log messages in a language other than English.

The diagnostic messages are printed on the auto report output listing when the errors occur during the processing of an auto report program. They are contained within the system message (canned) file. To establish a new version of the diagnostic messages, the auto report diagnostic message texts of the system message file may be changed.

The listing headers are printed at the top of every page of the auto report output listing. The listing headers are contained within the message module file. To establish a new version of the message module file, the message texts of the source code may be changed. The changed source code is then assembled, and the assembled message module is linked and placed in the same load library as the auto report AUTO# load file.

The log messages consist of the starting, ending, and internal error messages and are written to the log file. The log messages are contained within the message module file. To establish a new version of the message texts of the message module file, the message texts of the source code may be changed. The changed source code is then assembled, and the assembled message module is linked and placed in the same load library as the auto report AUTO# load file.

The format for message text in the messages module file is shown in Table 19-1.

The following rules apply to changing message texts:

- A message text is represented from the IDnnTEXT to the IDnnEND, and may be changed to a new message text (except for the IDnnINSx entries).
- The maximum length of a listing header is 120 bytes if the identification number is less than 40.
- The maximum length of a log message is 80 bytes if the identification number is equal to or more than 40.
- The order of the inserts may be changed, but the number and length cannot be changed.

Table 19—1. Message Text Format

Label	Attribute	Description
IDnnSTRT	H'nn'	Identification number of message
	Y	Total length of the current entry
	HL1	Number of inserts
	** { AL1	Length of insert
	{ HL1	Displacement of insert
	YL1	Length of message text
IDnnTEXT	C'text-1'	Text-1 of message
IDnnINS1	CL*insert1-length*	Area for insert-1
	C'text-2'	Text-2 of message
IDnnINS2	CL*insert2-length*	Area for insert-2
	C'text-3'	Text-3 of message
IDnnEND	C**'	End of message

## LEGEND:

- nn           Is the identification number of the message
- \*\*           Indicates that these entries repeat the number of times represented in the number of the inserts field
- IDnnINSx   Indicates that the message occurs the number of times represented by the number of the insert. The insert consists of a special constant that is set by auto report.

**PART 7. APPENDIXES**



## **Appendix A. RPG II Detail Logic Cycle**

### **A.1. GENERAL**

Each program that is generated by the RPG II compiler goes through the same logic cycle. In this appendix, the overall detail logic cycle is discussed first and then the individual subroutines are discussed.

### **A.2. OVERALL RPG II DETAIL LOGIC CYCLE**

Figure A-1 shows the overall RPG II detail logic cycle. Each step in the cycle is numbered and an explanation of it is given in the text that follows.

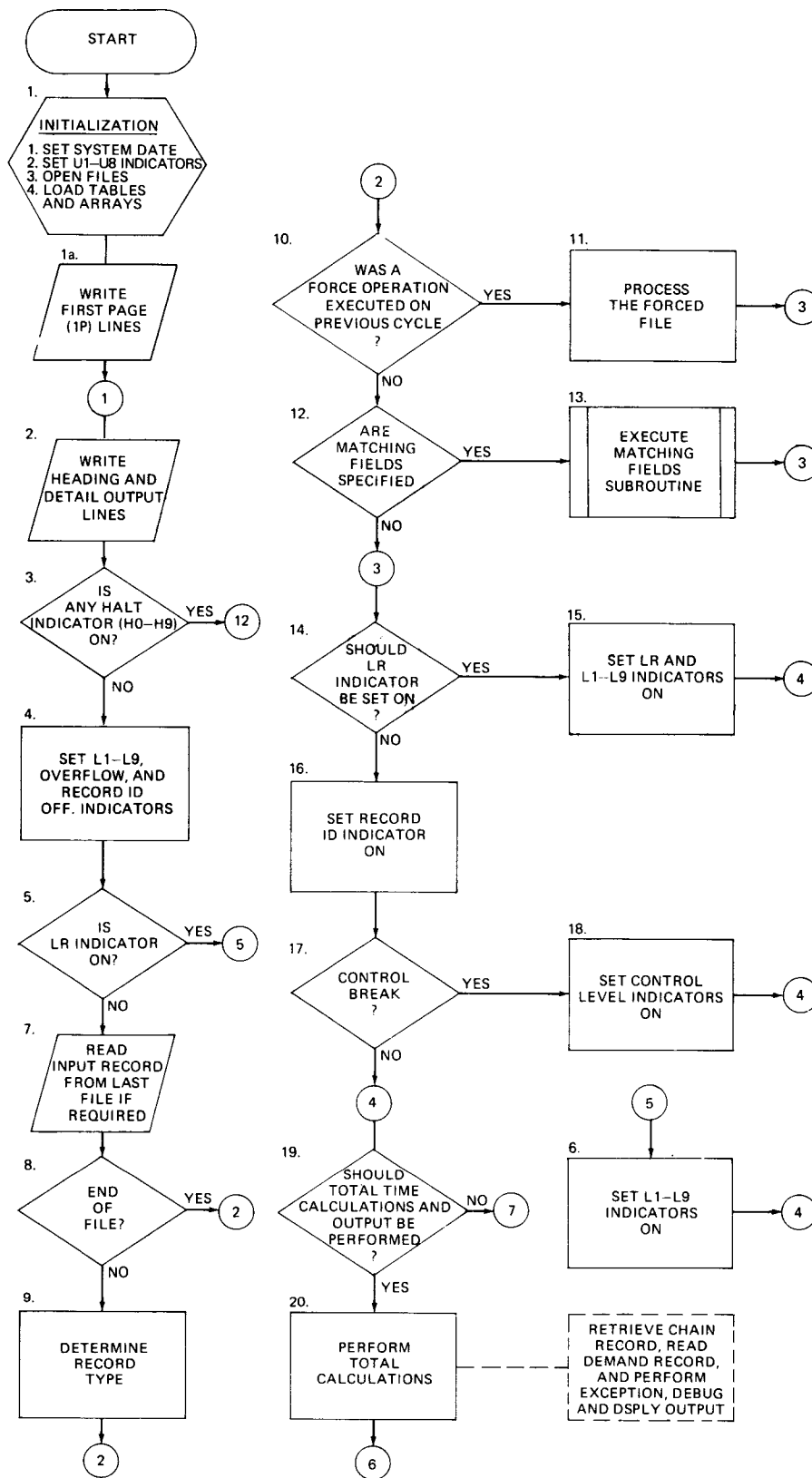


Figure A-1. Overall RPG II Detail Logic Cycle (Part 1 of 2)



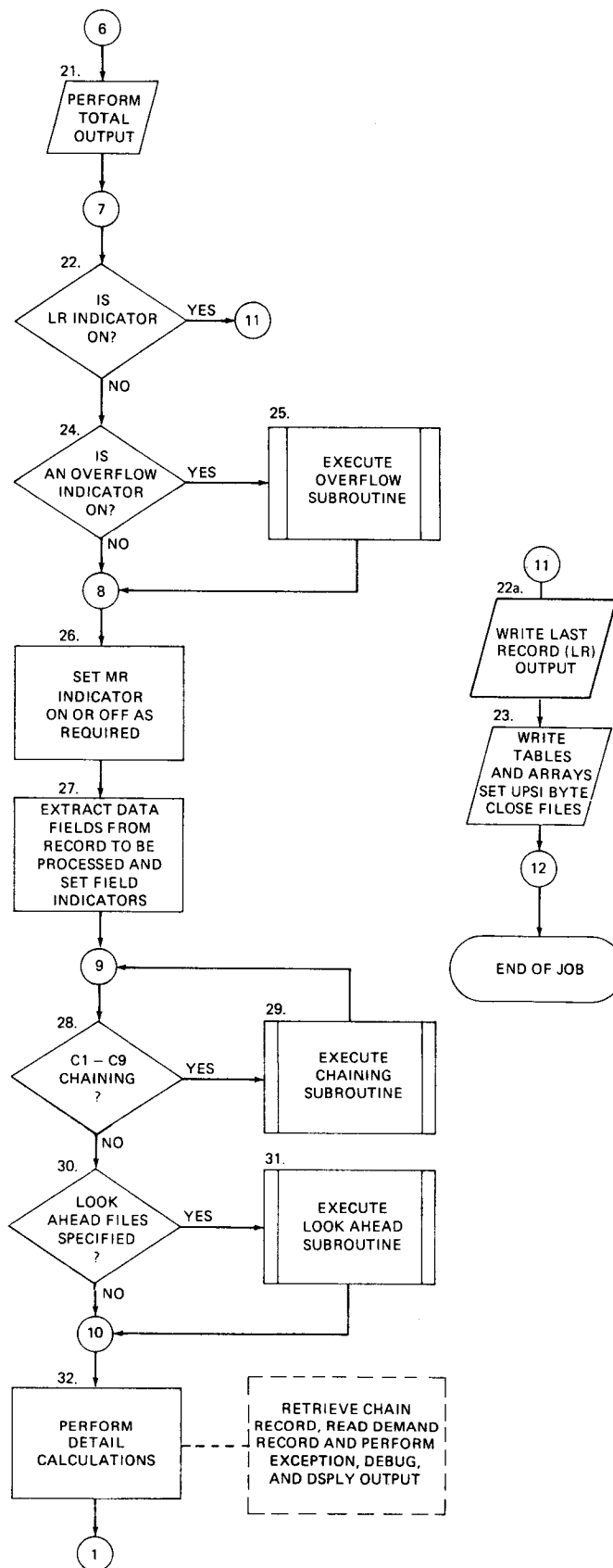


Figure A-1. Overall RPG II Detail Logic Cycle (Part 2 of 2)

Explanation of steps in Figure A-1:

1. This step initializes your program for execution. The initialization consists of setting the system date, setting the U1 through U8 indicators, opening the files, and loading the execution time tables and arrays.
  - 1a. Code for header and detail output lines conditioned by the first page indicator (1P) is executed prior to all other header/detail output lines. The 1P indicator, which was compiled on, is set off.
2. This step causes all heading and detail lines that meet the conditions specified on the output format specifications form to be printed. If fetch overflow is specified and the overflow indicator is on, the overflow lines are printed. This step is always executed at the beginning of the program so that the heading lines can be printed before processing begins.
3. The program terminates at this point if any of the halt indicators are on (HO through H9). If you do not want your program to terminate, you must set the halt indicators off before this step.
4. The control level indicators (L1 through L9) and record identifying indicators are set off. The overflow indicators are set off unless they were set on during the previous detail calculations or detail output. All other indicators that are on remain on.
5. The LR indicator is tested to see if it is on. If so, the program performs total calculations.
6. The control level indicators (L1 through L9) are set on.
7. A record is read from the last file that was processed. If a record address file is being used to retrieve records from this file, the record address defines the record to be retrieved. A record is not read from this file if look-ahead fields are specified in the last record that was processed because the record is already in main storage. A record is read from all primary and secondary input files during the first cycle through your program so that the matching fields subroutine can select the record for processing.
8. The file that was just read is tested for end-of-file. If end-of-file has occurred, step 9 is bypassed.
9. If a record is read from the file, the record type is determined and the record sequence is checked. If the record type cannot be determined or the record is not in the proper sequence, the HO indicator is set on.
10. If a FORCE operation was executed on the previous cycle, the forced file is selected for processing and step 12 is bypassed. Step 12 is bypassed because all records processed with the FORCE operation are processed with the MR indicator off.
11. The forced file is selected for processing after any matched fields are removed from the file that was just read. If a file is forced at end-of-file, the normal multifile logic selects the next record for processing.

12. If matching fields are specified, the matching fields are moved to the holding area for that file and the next record for processing is selected based on the value in the matching fields. If matching fields are not specified, the record read at step 7 is processed.
13. The matching fields subroutine is executed. It extracts the matching fields and performs sequence checking. It then determines which file is to be processed next. If the matching fields are not in sequence, the HO indicator is set on.
14. A test is made to see if the LR indicator should be set on. The LR indicator is set on when all of the records have been processed from the files that have an E specified in column 17 of the file description specifications and all matching secondary records have been processed.
15. Set the LR indicator on. When the LR indicator is set on, the L1 through L9 indicators are also set on. The L1 through L9 indicators can be used for conditioning only if they have been previously defined as conditioning indicators.
16. Set on the record identifying indicator for the record selected for processing.
17. The value in the control fields of the record being processed is compared with the value in the control fields of the last record processed. If they differ, a control break occurs.
18. If a control break occurs, the appropriate level indicator is set on, along with all lower level indicators, and the contents of the control field are saved.
19. A test is made to see if total time calculations and output should be performed. If control levels are not specified in columns 59 and 60 of the input format specifications form, total calculations and output are bypassed on the first cycle only. If control levels are specified, total calculations and output are bypassed only until after the first record with control fields is processed. When the LR indicator is on, total calculations and output are always performed.
20. All conditioned total time calculations and subroutines are performed at this time. The calculations can include CHAIN, DEBUG, DSPLY, EXCPT, and READ operations. The CHAIN operation immediately retrieves a record from an input file. The DEBUG operation causes the status of all indicators to be printed. The DSPLY operation allows a field to be displayed and a new value entered. The READ operation retrieves the next record from a demand file. The EXCPT operation causes the exception output lines to be written. The resulting indicators that are specified are set on or off at this time.
21. All conditioned total output lines are written. If the overflow indicator is on and fetch overflow is specified, the overflow lines are written.
22. If the LR indicator is on, steps 22a and 23 are performed. If the LR indicator is off, steps 22a and 23 are bypassed.
- 22a. Code for total output lines conditioned by the last record indicator (LR) is executed after all other total output lines.

23. All tables and arrays are written, U1 through U8 indicators are copied to the UPSI bytes, the files are closed, and the program is terminated.
24. If the overflow indicator is on, step 25 is performed. If the overflow indicator is off, step 25 is bypassed.
25. The overflow subroutine is executed. All lines conditioned by overflow indicators are executed. These lines are executed only if they were not executed by fetch overflow logic (step 21).
26. The MR indicator is set at this time if matching records are used. When the records match, the MR indicator is set on and remains on for the complete cycle that processes the matching record.
27. The data fields of the record to be processed are extracted and the field indicators for these fields are set.
28. If the record being processed has any fields specified with C1 through C9 chaining, step 29 is executed. If not, step 29 is bypassed.
29. The chaining subroutine is executed. It retrieves the selected record, sets the record identifying indicator, extracts the input fields, sets the corresponding field indicators, and returns to step 28.
30. If look-ahead fields are present, step 31 is executed. If not, step 32 is executed.
31. The look-ahead subroutine is executed. It retrieves the look-ahead record and then extracts the look-ahead fields.
32. All conditioned detail calculations and subroutines are performed. The calculations can include CHAIN, DEBUG, DSPLY, and READ operations.

### **A.3. RPG II DETAIL LOGIC SUBROUTINES**

The RPG II detail logic cycle contains four subroutines: the matching field subroutine, chaining subroutine, overflow subroutine, and look-ahead subroutine. These subroutines are shown in Figure A-2. Each step in each subroutine is numbered, and an explanation is provided in the subsections that follow.

#### **A.3.1. Matching Fields Subroutine**

The matching fields subroutine is executed when matching fields are specified in your program (Figure A-1, step 12). This subroutine consists of the following steps:

1. If multifile processing is being used, step 2 is executed. If not, step 2 is bypassed.
2. The values of the match fields in the hold area are tested to see which file is to be processed next.

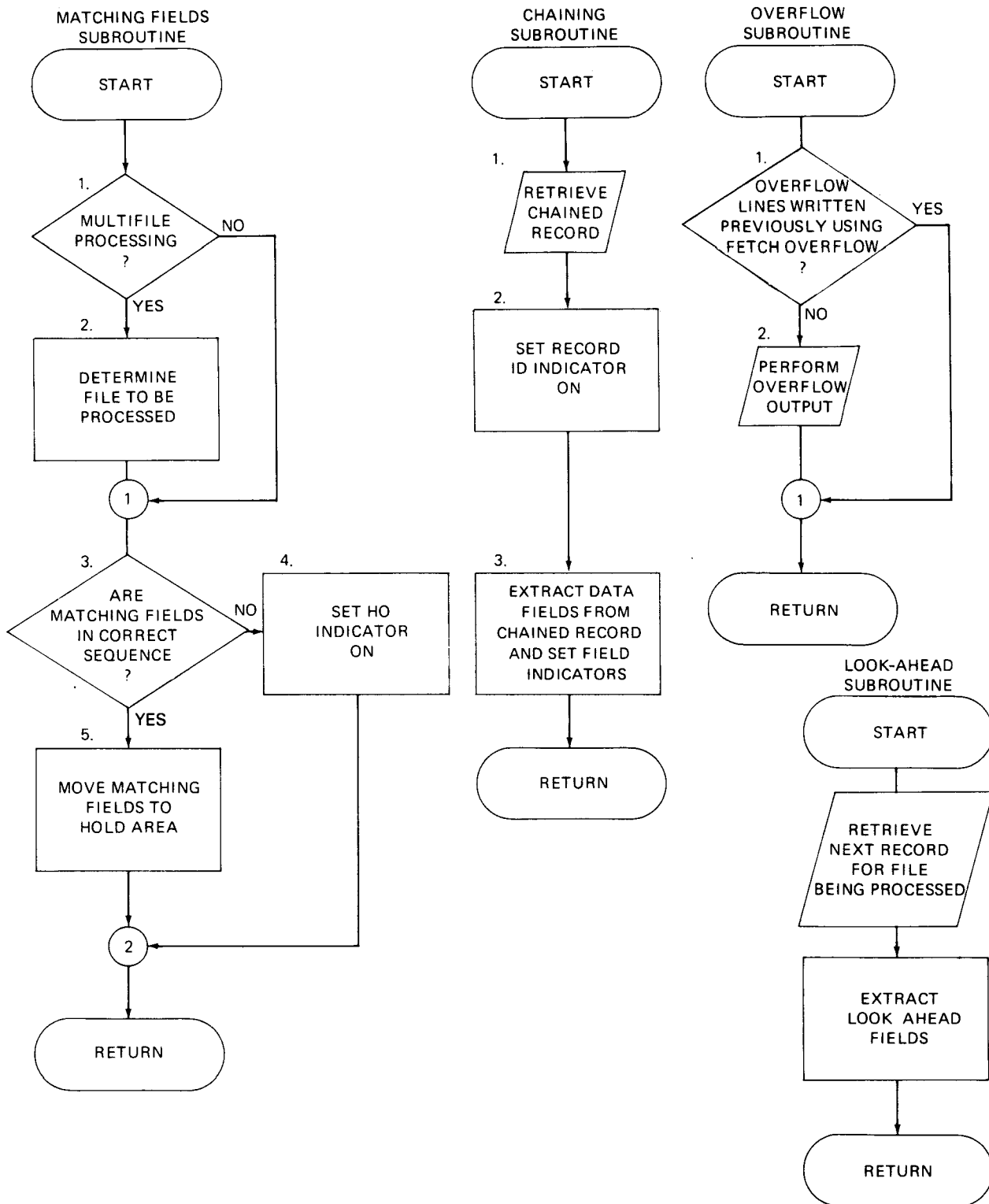


Figure A-2. RPG II Detail Logic Subroutines

3. The match fields are tested to see whether they are in the correct sequence. If they are, step 5 is executed. If not, step 5 is bypassed.
4. If the match fields are not in the correct sequence, the H0 indicator is set on.
5. The match fields are moved to a hold area.

### **A.3.2. Chaining Subroutine**

The chaining subroutine is executed when C1 through C9 chaining fields are specified in your program (Figure A-1, step 28). This subroutine consists of the following steps:

1. The chained record for the file is retrieved.
2. Set on the record identifying indicator for the chained record.
3. The input fields are extracted and the corresponding field indicators are set.

### **A.3.3. Overflow Subroutine**

The overflow subroutine is executed when an overflow indicator is on in your program (Figure A-1, step 24). This subroutine consists of the following steps:

1. A test is made to see whether the overflow lines were written previously by using the fetch overflow logic. If they were, step 2 is bypassed.
2. All output lines that are conditioned by the overflow indicator are written.

### **A.3.4. Look-Ahead Subroutine**

The look-ahead subroutine is executed when look-ahead fields are specified in your program (Figure A-1, step 30). This subroutine consists of the following steps:

1. The next record for the file being processed is retrieved. For combined or update files, the look-ahead fields are extracted from the record that is being processed.
2. The look-ahead fields are extracted.

## Appendix B. Programming Examples

### B.1. GENERAL

The programming examples that are contained in the subsections that follow are designed to show simple applications which the novice can relate to everyday use. Each example contains a flowchart, a description of the program, and the RPG II specifications forms with the required coding on them.

The examples that are shown could be combined into three programs and executed in steps; however, they are broken down into eight programs for clarity. This is done because RPG II allows you to organize, retrieve, and process files in many different ways and still accomplish the same result. The intent here is to show the use of as many operations and processing methods as possible; therefore, the entries on the specifications forms should not be interpreted as the most efficient or logical way to use RPG II for a specific type of application. The information required to compile, link-edit, and execute RPG II programs is provided in Section 18. Program testing aids are provided in Appendix C.

### B.2. CREATING A SEQUENTIAL FILE

This program reads records containing information about a company's sales staff from punched cards and creates a sequential output file on a tape device by writing these records in the order in which they are read. This program also provides a listing that shows the contents of the individual records on the output file and a count of the total number of records. The program processing steps are shown in Figure B-1.

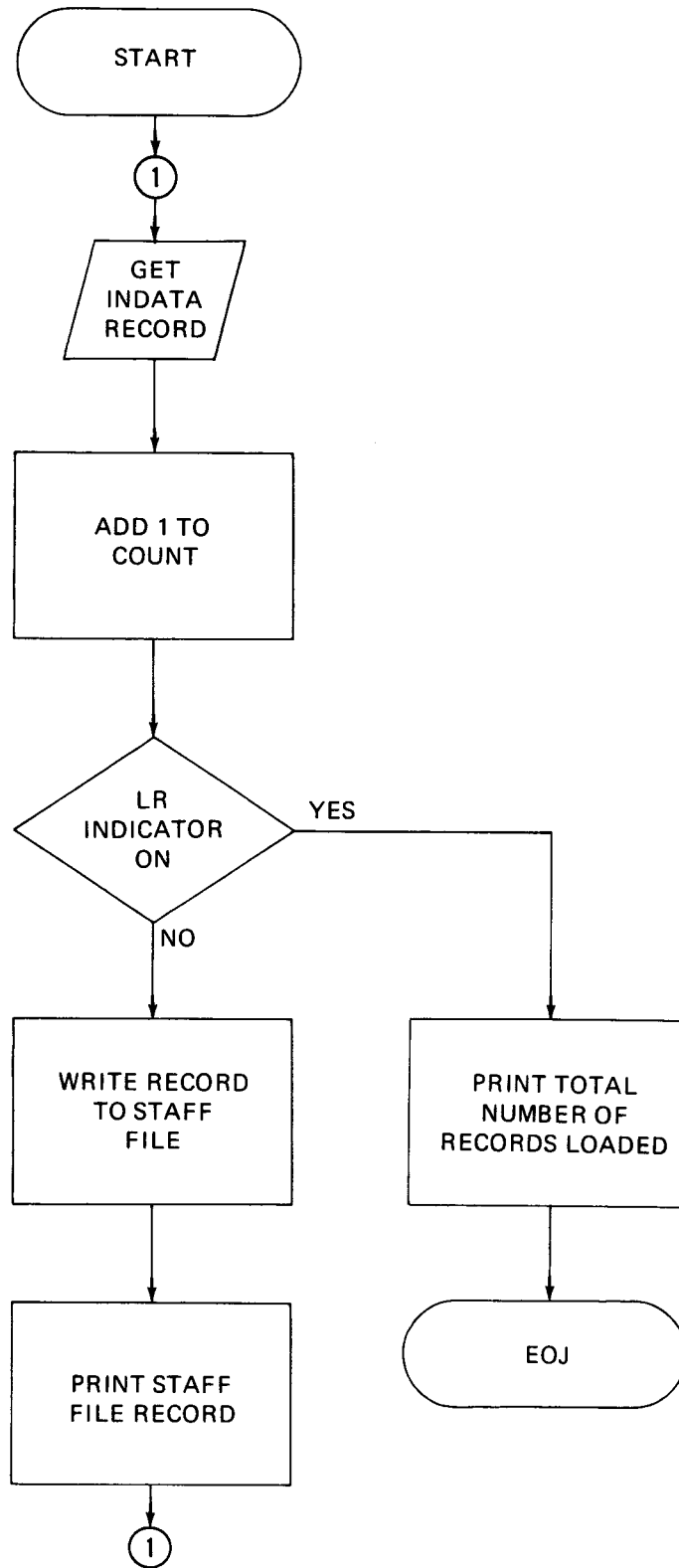


Figure B-1. Creating a Sequential File



## **B.2.1. Form Entries**

Figure B-2 shows the entries that are required for this program on the various specifications forms. These entries are discussed in the subsections that follow.

### **B.2.1.1. Control Card Specifications**

The S in column 40 specifies that numeric data is to be checked for a standard sign. CRSEQ in columns 75 through 80 specifies the name that is to be assigned to this program during compilation.

### **B.2.1.2. File Description Specifications**

INDATA is specified as the primary input and it is read from a card reader. The file is in ascending order (A in column 18) and it is to be processed to end-of-file. STAFF is an output file that is to contain standard labels (S in column 53) and is to be written on a tape device. LIST is an output file that provides a listing of the records on the STAFF file.

### **B.2.1.3. Input Format Specifications**

Records on the INDATA file are in no particular sequence (AA in columns 15 and 16). Indicator 01 is set on when a record is read.

### **B.2.1.4. Calculation Specifications**

As each INDATA record is read, COUNT is incremented by 1.

### **B.2.1.5. Output Format Specifications**

The heading line, STAFF FILE LISTING, is printed on the first page of the report and the COUNT is set to zero at the beginning of the program. Each time an INDATA record is read (indicator 01 is on), a record is written on the STAFF file and printed on the report (LIST file). When the INDATA file is exhausted (end-of-file), the LR indicator is set on and the constant TOTAL RECORDS LOADED= is printed followed by the contents of the COUNT field.

## **B.2.2. Report Format**

The format of the report produced by this program is shown in Figure B-3.







### **B.3. SEQUENTIAL FILE PROCESSING - MONTHLY SALES EARNINGS**

This program shows a common application using a sequential file. The STAFF file, created previously (B.2), is processed sequentially since each record in the file must be examined to determine whether a salesman is entitled to a commission check this month. This program also accumulates wage and commission totals for use in subsequent accounting programs and builds a line counter file for printing the commission checks offline. The program processing steps are shown in Figure B-4.

#### **B.3.1. Form Entries**

Figure B-5 shows the entries that are required for this program on the various specifications forms. These entries are discussed in the subsections that follow.

##### **B.3.1.1. Control Card Specifications**

SALES in columns 75 through 80 specifies the name that is to be assigned to this program during compilation.

##### **B.3.1.2. File Description Specifications**

STAFF is a primary input file in fixed-blocked format. It is a sequential tape file that is checked for end-of-file. COMMRPT is an output file that provides a monthly sales commission report. The CHECKS file is an output line counter file (L in column 39) which uses OV as its overflow indicator and stores data for commission checks on an intermediate tape file.

##### **B.3.1.3. Line Counter Specifications**

The line counter specifications form describes the internal carriage control tape when data is stored on an intermediate file. These entries relate a printed line to its corresponding punch in the carriage control tape. On line 010, channel 07 (columns 18 and 19) is associated with line 006 (first printing line on the form); therefore, channel 07 must contain a punch at the same relative position on the carriage control tape. The line number associated with channel 01 causes the setting of the overflow indicator as specified on the file description specifications form. When the internal line counter reaches or passes the specified number (060) with a space or print operation, the overflow indicator is set on.

##### **B.3.1.4. Input Format Specifications**

Indicator 01 is set on when a record from the STAFF file is read.

**B.3.1.5. Calculation Specifications**

When indicator 01 is set on, an individual salesman's salary and expenses are added to give monthly wages on line 010, monthly sales are multiplied by the commission rate to give monthly earned commission on line 020, total wages are accumulated on line 030, and total commissions are accumulated on line 040. Monthly commissions are then compared against monthly wages on line 050. If commissions (factor 1) are greater than wages (factor 2), indicator 06 is set on. If indicator 06 is on, wages are subtracted from commissions, giving the commission due to the salesman.

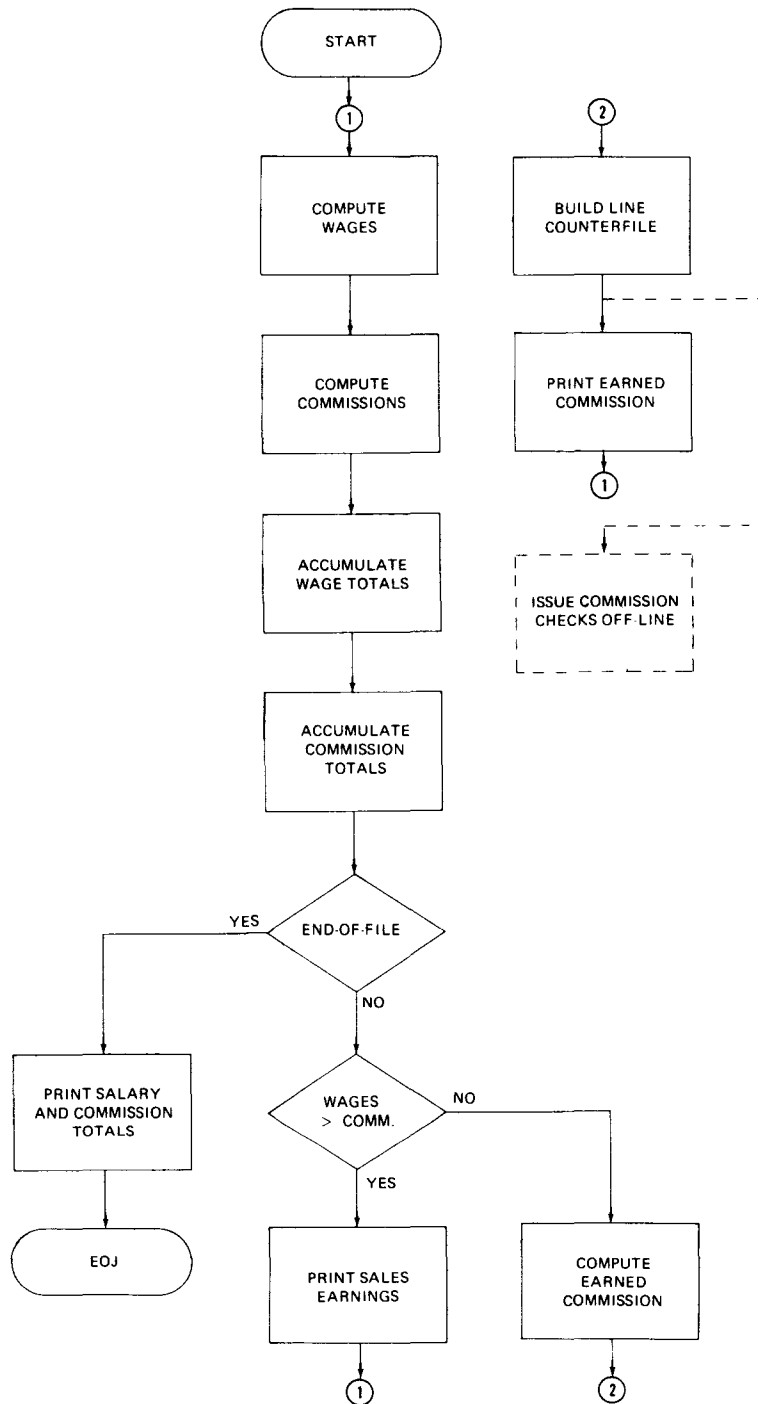


Figure B-4. Sequential File Processing — Monthly Sales Earnings













### **B.3.1.6. Output Format Specifications**

Page 05, lines 010 through 030 show that a line counter file (CHECKS) is built containing the data necessary to issue commission checks.

Page 06, lines 010 through 140 show that the monthly sales commission report and two heading lines are printed on the first page and on the top of a new page when forms overflow occurs.

Page 06, lines 150 through 220 show that these detail lines are printed whenever indicators 01 or 01 and 06 are on. The SALES, WAGES, and COMM fields are printed and edited each detail time.

Page 06, line 220 shows that the field CDUE is only printed when indicators 01 and 06 are on.

Page 07, lines 010 through 030 show that a single constant line is printed after spacing two lines at total output time (LR indicator is on).

Page 07, lines 040 through 050 show that the accumulated total wages are edited and printed, providing balance totals for later programs.

Page 07, lines 060 through 090 show that a single constant line is printed after spacing two lines. This line is followed by the editing and printing of the total commission.

### **B.3.2. Report Format**

The format of the report produced by this program is shown in Figure B-6.

## **B.4. CREATING AN INDEXED SEQUENTIAL FILE - PAYROLL MASTER FILE**

This program reads records containing employee information from punched cards and writes them on disk in ascending sequence by record key. This program also provides a listing that shows the contents of the individual records on the output file and a count of the total number of records. The program processing steps are shown in Figure B-7.

### **B.4.1. Form Entries**

Figure B-8 shows the entries that are required for this program on the various specifications forms. Other than specifying on line 020 of the file description specifications form that the output file PAYMAS is an indexed sequential file (I in column 32) and specifying an 8-character alphanumeric key field (8 in columns 29 and 30 and A in column 31) that is to be used to build an index for the PAYMAS file, the entries required for this program are the same as those described for creating a sequential file in B.2.



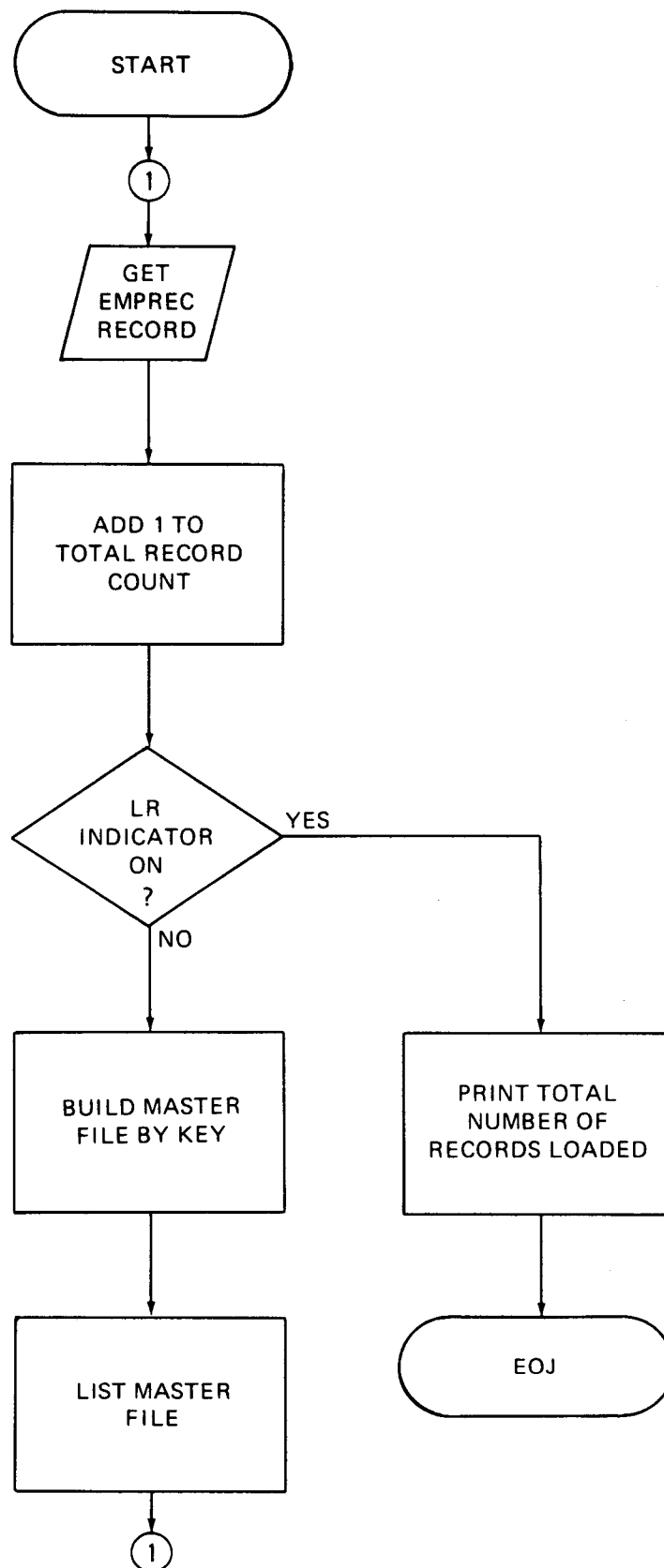


Figure B-7. Creating an Indexed Sequential File — Payroll Master File







## **B.4.2. Report Format**

The format of the report produced by this program is shown in Figure B-9.

## **B.5. PROCESSING AN INDEXED SEQUENTIAL FILE SEQUENTIALLY BY KEY - WAGE INCREASE**

This program shows an application using an indexed sequential file that is processed sequentially by key. An across-the-board pay raise for all employees has been authorized. The PAYMAS file, created previously (B.4), is processed sequentially because each record in the file must be examined to see by what percentage the employee's hourly rate is to be increased. If the hourly rate is less than \$4.00, the pay rate is increased by 10%. If greater than \$4.00, the pay rate is increased by 5%. The record with the lowest key is selected first and then records with successively higher keys are selected until the end-of-file is reached. The program processing steps are shown in Figure B-10.

### **B.5.1. Form Entries**

Figure B-11 shows the entries that are required for this program on the various specifications forms. These entries are discussed in the subsections that follow.

#### **B.5.1.1. Control Card Specifications**

SKEY in columns 75 through 80 specifies the name that is to be assigned to this program during compilation.

#### **B.5.1.2. File Description Specifications**

PAYMAS is a primary input file in fixed-blocked format containing 80-character records. The file is tested for end-of-file (E in column 17) and is in ascending sequence (A in column 18). This is an indexed sequential file (I in column 32) with a key length of 8 (8 in columns 29 and 30) which begins in record position 1 (1 in columns 35 through 38). The key is an unpacked alphanumeric key (A in column 31). The file is a disk file (DISC in columns 40 through 43).

NEWRATE is an output file that provides a listing of the hourly pay rate increases for each employee.

PRINTER FORMAT CHART  
180 PRINT POSITIONS

UNIVAC

TYPE OF PRINTER: \_\_\_\_\_ DATE: \_\_\_\_\_

PROGRAMMER: \_\_\_\_\_

APPLICATION: \_\_\_\_\_

10	20	30	40	50	60	70	80	90	100	110	120	130	140	150	160	170	180
MASTER FILE LISTING																	
TOTAL MASTERS LOADED .KXX																	

Figure B-9. Master File Listing Format

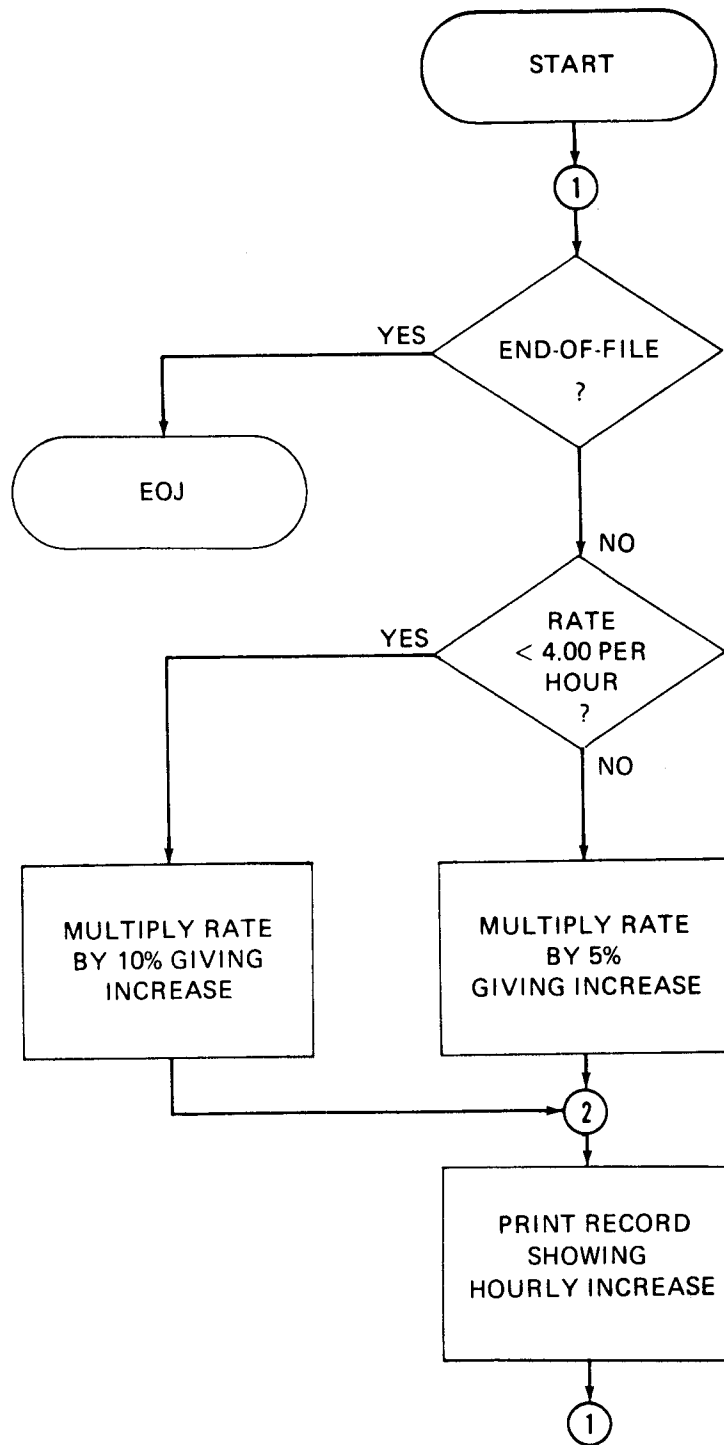


Figure B-10. Processing an Indexed Sequential File by Key — Wage Increase





# OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW				SPACE		SKIP		OUTPUT INDICATORS						FIELD NAME	DATA FORMAT P.B./L/R				CONSTANT OR EDIT WORD				NOT USED	PROGRAM IDENTIFICATION										
			TYPE H/D/T/E				BEFORE	AFTER	BEFORE	AFTER	AND	AND	N NOT	N NOT	N NOT	N NOT		END POSITION IN OUTPUT RECORD	EDIT CODES	COMMAS INSERTED	ZERO BALANCE TO PRINT	CODES	ACTION														
1	2	3	4	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45	70	71	74	75	80
04	0,1,0	0	N	E	W	R	A	T	E	H			2	0	7																						
04	0,2,0	0					O	R																													
04	0,3,0	0																																			
04	0,4,0	0																																			
04	0,5,0	0							H				1																								
04	0,6,0	0							O	R																											
04	0,7,0	0																																			
04	0,8,0	0																																			
04	0,9,0	0																																			
04	1,0,0	0							H				1																								
04	1,1,0	0							O	R																											
04	1,2,0	0																																			
04	1,3,0	0																																			
04	1,4,0	0																																			
04	1,5,0	0							D				1																								
04	1,6,0	0																																			
04	1,7,0	0																																			
04	1,8,0	0																																			
04	1,9,0	0																																			
04	2,0,0	0																																			

Figure B-11. Entries for Processing an Indexed Sequential File by Key — Wage Increase (Part 3 of 3)

### **B.5.1.3. Input Format Specifications**

The records in the PAYMAS file are in no special sequence (AA in columns 15 and 16) and indicator 01 is set on when a record is read. MKEY, NAME, and RATE are the fields that will be used during the execution of the program.

### **B.5.1.4. Calculation Specifications**

Line 010 shows that an employee's pay rate is compared against a base of \$4.00. If factor 1 is greater than factor 2, indicator 03 is set on.

Line 020 shows that if indicator 03 is on, the program branches to line 050.

Line 030 shows that if indicator 03 is off, the employee's hourly pay rate is increased by 5% and this increase is added to his current rate to establish his new rate on line 040.

Lines 060 through 070 show that for those employees whose hourly rate is less than \$4.00, the hourly rate is increased by 10%.

### **B.5.1.5. Output Format Specifications**

Lines 010 through 140 show that the NEWRATE file heading information is printed on the first page of the report or when forms overflow occurs.

Lines 150 through 180 show that the detail lines are printed in successive order from lowest employee number to highest.

### **B.5.2. Report Format**

The format of the report that is produced by this program is shown in Figure B-12.

## **B.6. PROCESSING AN INDEXED SEQUENTIAL FILE SEQUENTIALLY BETWEEN LIMITS - DEPARTMENT PAY RATES**

This program shows an application using a record address file to process an indexed sequential file sequentially between limits. A report is to be generated that lists the new pay rate of employees by department. The PAYMAS file is in sequence by employee number. The first three digits of the employee number represent the department the employee is in. The report is generated by defining a record address file, RAF, containing the lower and upper limits which allows all the records for a given department to be retrieved; that is, all the records on that segment of the file will be retrieved. Processing begins when a key is found that is greater than or equal to the lower limit. Processing ends when a key is found that is greater than the upper limit. The program processing steps are shown in Figure B-13.

APPLICATION: \_\_\_\_\_ TYPE OF PRINTER \_\_\_\_\_  
PROGRAMMER: \_\_\_\_\_ DATE: \_\_\_\_\_

LINE NO	10	20	30	40	50	60	70	80	90	100	110	120	130	132	140	144	150	160	LINE NO	
	HOURLY PAY RATE INCREASE																			
	EMPLOYEE	NAME		OLD PAY	NEW PAY															
	NUMBER			RATE	RATE															
	XXXXXXXX	XXXXXXXXXXXXXXXXXXXX		\$XX.XXX	\$XX.XXX															

Figure B-12. Hourly Pay Rate Increase Report Format



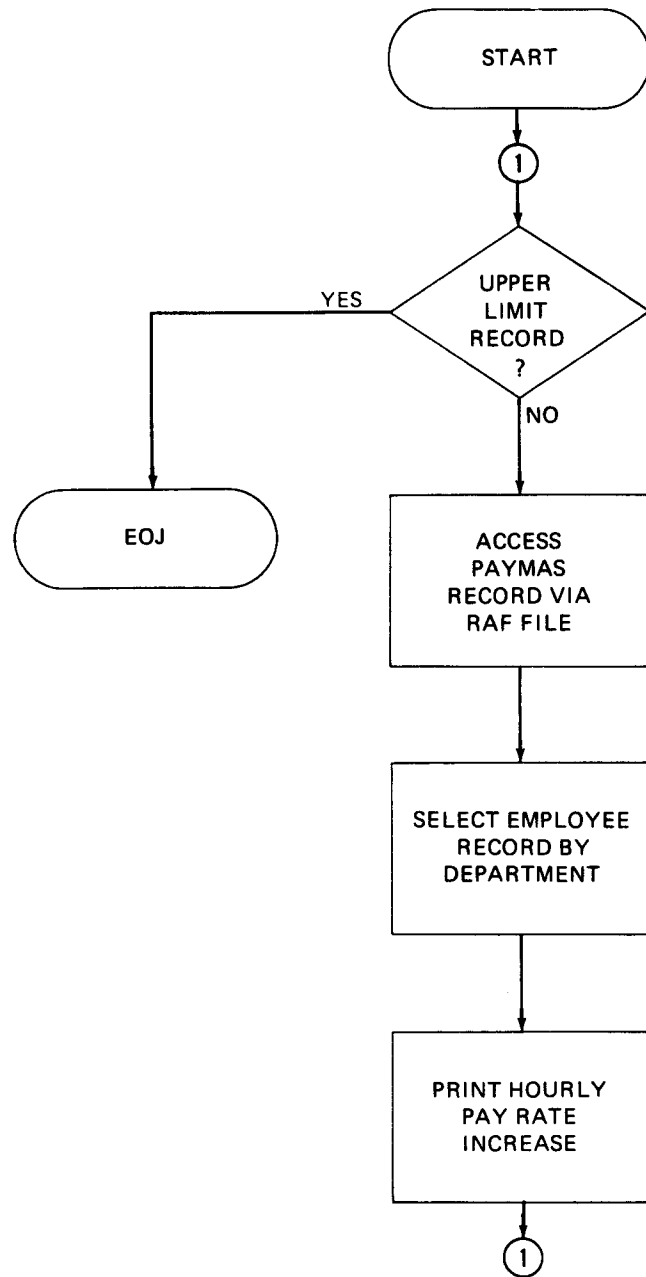


Figure B-13. Processing an Indexed Sequential File between Limits — Department Pay Rates

### **B.6.1. Form Entries**

Figure B-14 shows the entries that are required for this program on the various specifications forms. These entries are discussed in the subsections that follow.

#### **B.6.1.1. Control Card Specifications**

LIMITS in columns 75 through 80 specifies the name that is to be assigned to this program during compilation.

#### **B.6.1.2. File Description Specifications**

→ RAF is a record address file (R in column 16), in fixed format containing 80-character records, which supplies the lower and upper limits (keys) for processing the PAYMAS file. The key length is 8 and each record consists of two keys in the form: lower key in the first eight record positions and upper key in the last eight record positions. The E in column 39 indicates that further information about this file is required on the file extension specifications form.

PAYMAS is the primary input that contains the records to be processed. The file is an indexed sequential file (I in column 32) with a key length of 8 (8 in columns 29 and 30) which begins in record position 1 (1 in columns 35 through 38). The file is to be processed sequentially between limits (L in column 28).

PRINT is an output file that produces a listing of the department pay rates. The overflow indicator associated with this file is OF (OF in columns 33 and 34).

#### **B.6.1.3. File Extension Specifications**

The entries on the file extension specifications forms indicate that the RAF is being used to specify the records that are to be processed on the PAYMAS file.

#### **B.6.1.4. Input Format Specifications**

The appropriate fields on the PAYMAS file are specified and indicator 01 is used to indicate an active record.

#### **B.6.1.5. Output Format Specifications**

Lines 010 through 120 show that the heading lines for the report are conditioned by the 1P or OF indicator.

Lines 130 through 170 show that these detail lines provide the employee pay scale increases by department.

### **B.6.2. Report Format**

The format of the report that is produced by this program is shown in Figure B-15.







APPLICATION: \_\_\_\_\_ TYPE OF PRINTER: \_\_\_\_\_  
PROGRAMMER: \_\_\_\_\_ DATE: \_\_\_\_\_

LINE NO.	10	20	30	40	50	60	70	80	90	100	110	120	130	132	140	144	150	160	LINE NO.
	DEPARTMENT PAY RATES																		
	EMPLOYEE	NAME		RATE	DEPARTMENT														
	NUMBER			PER HOUR															
	XXXXXXXX	XXXXXXXXXXXXXXXXXXXX		\$.XX.XXX	XXXXXXXXXXXX														

Figure B-15. Department Pay Rates Report Format

## **B.7. UPDATING AN INDEXED SEQUENTIAL FILE BY CHAINING - MASTER FILE CHANGES**

This program randomly retrieves selected records from the master file, PAYMAS, by using the CHAIN operation. The master file is updated by making changes to employee records; the records from employees no longer employed and the changes made to active records are listed on a report.

Chaining is accomplished by using the CHAIN operation on the calculation specifications form. This operation allows a field value which is read from an input file (the chaining file named CHAIN) to act like a key to retrieve a master file record. The master file (the chained file), PAYMAS, is an indexed sequential file that is processed randomly. The key field in PAYMAS is the employee number and this field is also used in the CHAIN file. The program uses the employee number in the CHAIN file to locate a record with the same number in the PAYMAS file. When the record has been located, the new pay rate of an employee (contained in the CHAIN file) is used to update the PAYMAS file.

The CHAIN file contains a 1 in record position 9 for those employees that are no longer employed. When this condition occurs, the program does not print this record on the report.

The program processing steps are shown in Figure B-16.

### **B.7.1. Form Entries**

Figure B-17 shows the entries that are required for this program on the various specifications forms. These entries are discussed in the subsections that follow.

#### **B.7.1.1. Control Card Specifications**

UPDATE in columns 75 through 80 specifies that name that is to be assigned to this program during compilation.

#### **B.7.1.2. File Description Specifications**

CHAIN (the chaining file) is specified as the primary input file. It is in fixed format and contains 80-character records. ←

PAYMAS (the chained file) is specified as an update chained file (U in column 15 and C in column 16). It is an indexed sequential file (I in column 32) that is fixed-blocked format with a key field length of 8 (8 in column 30) that begins in record position 1 (1 in columns 35 through 38). MLIST is an output file that produces a listing of the master file changes. The overflow indicator is OF.

#### **B.7.1.3. Input Format Specifications**

The appropriate fields from the CHAIN file and the PAYMAS file are described. The CHAIN file has two valid record types. 01 records update the PAYMAS file with the new pay rate. 03 records indicate that the employee record should be deleted.

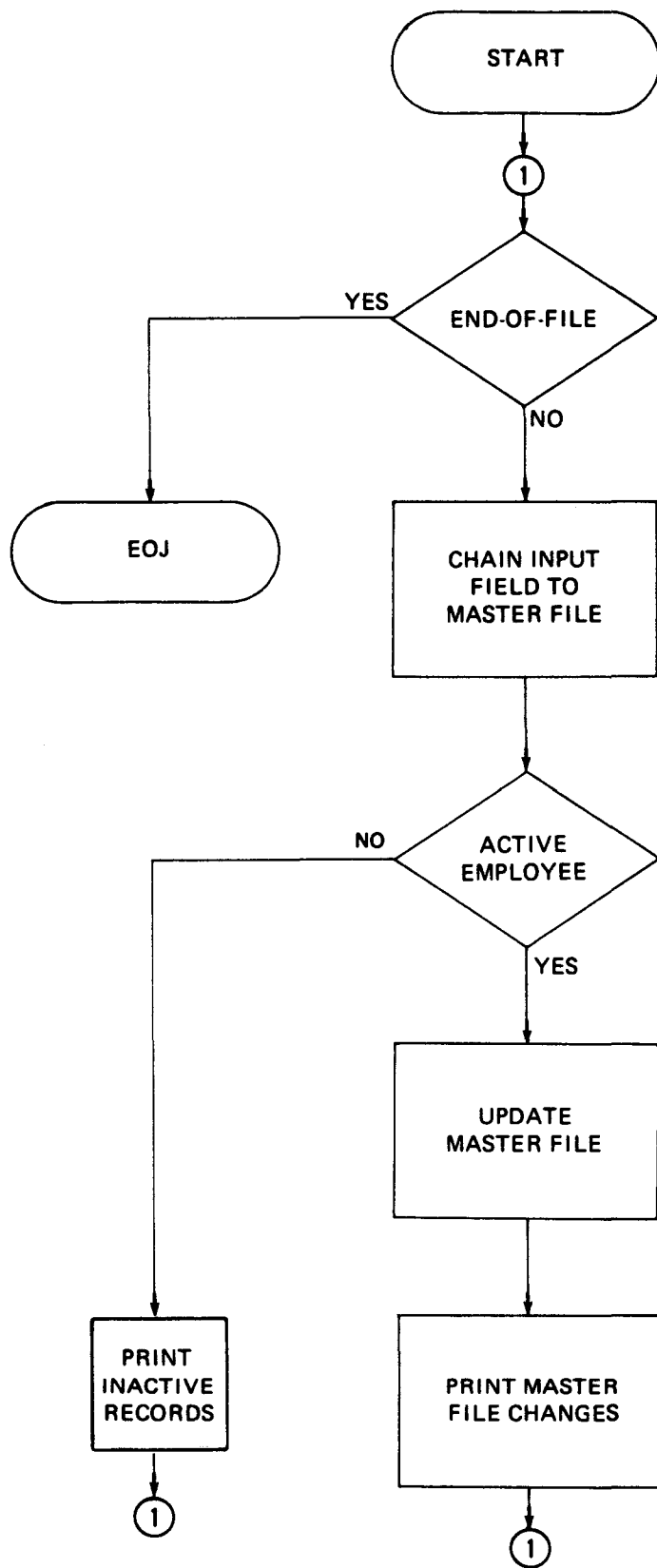


Figure B-16. Updating an Indexed Sequential File by Chaining — Master File Changes







# OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW	TYPE H/D/T/E	SPACE		SKIP		OUTPUT INDICATORS						FIELD NAME	DATA FORMAT P/B/L/R	END POSITION IN OUTPUT RECORD	CONSTANT OR EDIT WORD				NOT USED	PROGRAM IDENTIFICATION														
					BEFORE	AFTER	BEFORE	AFTER	N-NOT	N-NOT	N-NOT	N-NOT	N-NOT	N-NOT				N-NOT	N-NOT	N-NOT	N-NOT			CODES	ACTION												
1	2	3	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45	70	71	74	75	80	
04	0,1,0	0	PAYMAST	D											01	02																					
04	0,2,0	0																																			
04	0,3,0	0	MLIST	H			3								1P																						
04	0,4,0	0		OR											OF																						
04	0,5,0	0																																			
04	0,6,0	0		H			1								1P																						
04	0,7,0	0		OR											OF																						
04	0,8,0	0																																			
04	0,9,0	0																																			
04	1,0,0	0																																			
04	1,1,0	0		H			1								1P																						
04	1,2,0	0		OR											OF																						
04	1,3,0	0																																			
04	1,4,0	0																																			
04	1,5,0	0		D			2								01	02																					
04	1,6,0	0																																			
04	1,7,0	0																																			
04	1,8,0	0																																			
04	1,9,0	0																																			
04	1,10,0	0																																			
04	1,11,0	0																																			
04	1,12,0	0																																			
04	1,13,0	0																																			
04	1,14,0	0																																			
04	1,15,0	0																																			
04	1,16,0	0																																			
04	1,17,0	0																																			
04	1,18,0	0																																			
04	1,19,0	0																																			
04	1,20,0	0																																			
04	1,21,0	0																																			
04	1,22,0	0																																			
04	1,23,0	0																																			
04	1,24,0	0																																			
04	1,25,0	0																																			
04	1,26,0	0																																			
04	1,27,0	0																																			
04	1,28,0	0																																			
04	1,29,0	0																																			
04	1,30,0	0																																			
04	1,31,0	0																																			
04	1,32,0	0																																			
04	1,33,0	0																																			
04	1,34,0	0																																			
04	1,35,0	0																																			
04	1,36,0	0																																			
04	1,37,0	0																																			
04	1,38,0	0																																			
04	1,39,0	0																																			
04	1,40,0	0																																			
04	1,41,0	0																																			
04	1,42,0	0																																			
04	1,43,0	0																																			
04	1,44,0	0																																			
04	1,45,0	0																																			
04	1,46,0	0																																			
04	1,47,0	0																																			
04	1,48,0	0																																			
04	1,49,0	0																																			
04	1,50,0	0																																			
04	1,51,0	0																																			
04	1,52,0	0																																			
04	1,53,0	0																																			
04	1,54,0	0																																			
04	1,55,0	0																																			
04	1,56,0	0																																			
04	1,57,0	0																																			
04	1,58,0	0																																			
04	1,59,0	0																																			
04	1,60,0	0																																			

#### **B.7.1.4. Calculation Specifications**

Line 010 shows that the CKEY field of the CHAIN file is used to chain the PAYMAS file. This operation causes a record to be read from the PAYMAS file whose employee number is the same as that in the CKEY field.

Line 020 shows that the PAYMAS RATE field is updated with a new pay rate from the CHAIN file record.

Line 030 shows that when indicator 03 is on (a 1 is in record position 9 of the CHAIN file record), a message is printed indicating that the employee should be deleted from the file.

#### **B.7.1.5. Output Format Specifications**

Lines 010 through 020 show that when indicators 01 and 02 are on, the new pay rate is written on the PAYMAS; that is, the master file is updated with the new pay rate.

Lines 030 through 150 show that the heading lines for the report are conditioned by the 1P or OF indicator. The OF indicator controls the printing of the heading lines after the first page.

Lines 160 through 210 show that these detail lines are printed when indicator 01 is on.

Lines 220 through 240 show that when indicator 03 is on, an inactive record is present. If this occurs, the message NO LONGER EMPLOYED along with the employee number is printed.

#### **B.7.2. Report Format**

The format of the report produced by this program is shown in Figure B-18.

### **B.8. CREATING AN INDEXED SEQUENTIAL FILE - INVENTORY MASTER FILE**

This program reads records containing inventory information from punched cards and writes them on disk in ascending order by record key. This program also provides a listing that shows the contents of the individual records on the output files and a count of the total number of records.

The program processing steps are shown in Figure B-19.

#### **B.8.1. Form Entries**

Figure B-20 shows the entries that are required for this program on the various specifications forms. Other than the file and field names, the entries are essentially the same as those used in B.4 to create the indexed sequential payroll master file.

APPLICATION: \_\_\_\_\_ TYPE OF PRINTER \_\_\_\_\_  
PROGRAMMER: \_\_\_\_\_ DATE: \_\_\_\_\_

LINE	10	20	30	40	50	60	70	80	90	100	110	120	130	140	150	160
	MASTER FILE CHANGES															
	EMPLOYEE	NAME	CLASSIFICATION	NEW	DEPARTMENT											
	NUMBER			RATE												
	XXXXXXXX	XXXXXXXXXXXXXXXXXXXX	XXXXXXXXXXXXXXXXXXXX	\$.XX.XXX	XXXXXXXXXXXX											
	NO LONGER EMPLOYED	XXXXXXXX														

Figure B-18. Master File Changes Report Format

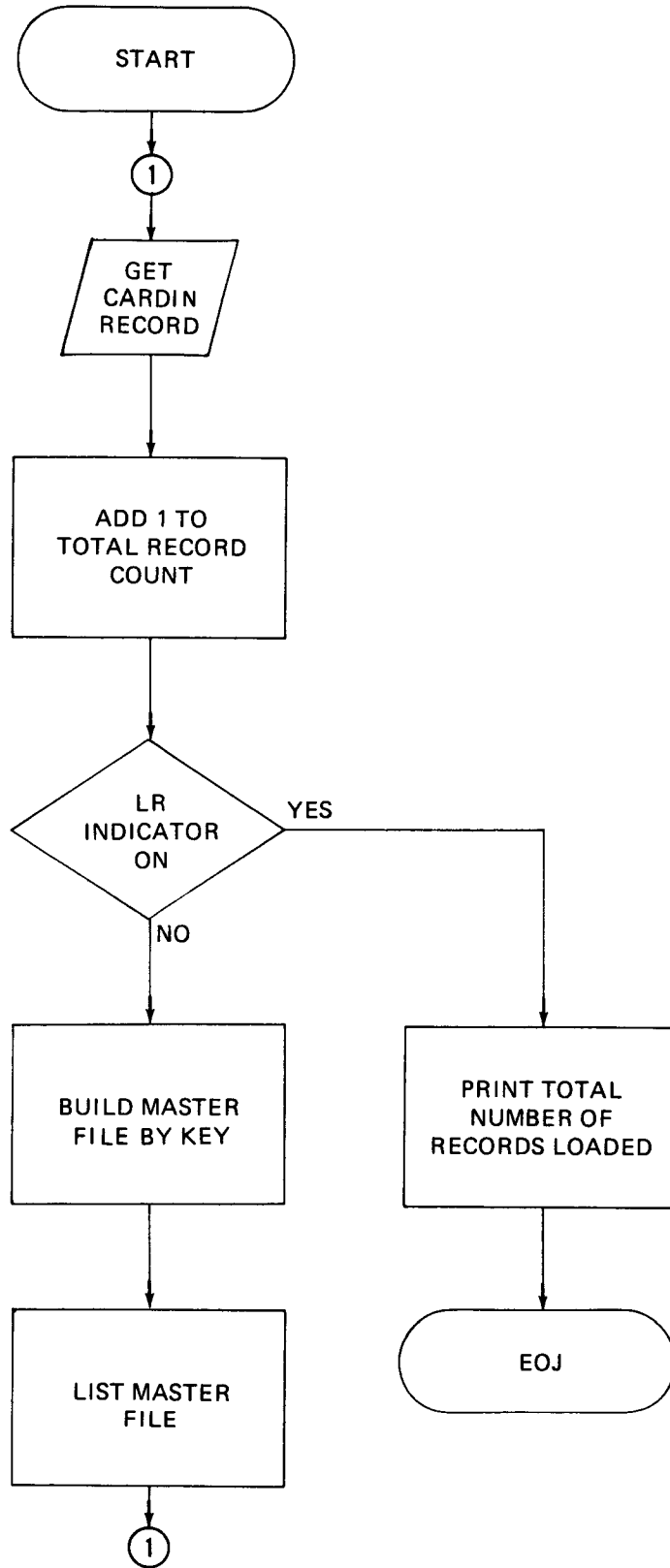


Figure B-19. Creating an Indexed Sequential File — Inventory Master File







### **B.8.2. Report Format**

The format of the report produced by this program is shown in Figure B-21.

## **B.9. PROCESSING WITH MATCHING RECORDS - INVENTORY CONTROL**

This program is designed to update inventory master file records, build a new master file, punch a card file of orders for those items whose quantity is below or equal to the reorder point, and to provide information on a daily activity report that reflects the stock status, usage, department charges for issues, and control totals for cost accounting.

The matching record technique is used to control the selection of records from multiple input files so that they are processed as if they were a single input file.

The file named OLDMAS is the primary or controlling file and the file named TRANS is the secondary file. The sequence in which the records are selected for processing is controlled by the values in the matching fields. The matching field values in the OLDMAS file are less than or equal to the matching field values in the TRANS file; therefore, the primary file is always processed first, since the record with the lowest value in its matching field is selected for processing first when ascending order is specified. The internal matching record indicator (MR) is used with the record identifying indicators and resulting indicators to control all calculation and output operations.

Initially, a record is read from each input and the matching field (part numbers in the OLDMAS and TRANS files) are compared to determine which file will be processed first. The OLDMAS file is updated if the part numbers are equal according to the type of transaction (order, receipt, or issue) and the updated data is written on the new master file, NEWMAS, and the daily activity report.

The NEWMAS file is built by writing all inactive records (OLDMAS exception records) on the file when a no match condition occurs in addition to the updated records. When an issue transaction is processed, the quantity-on-hand is checked to determine whether it is less than or equal to the reorder point. If it is, the program punches an order card.

If an invalid transaction code is detected, a message and the part number are printed on the printer.

The program processing steps are shown in Figure B-22.

### **B.9.1. Form Entries**

Figure B-23 show the entries that are required for this program on the various specifications forms.

#### **B.9.1.1. Control Card Specifications**

IVCTRL in columns 75 through 80 specifies the name that is to be assigned to this program during compilation.

APPLICATION: \_\_\_\_\_ TYPE OF PRINTER: \_\_\_\_\_  
PROGRAMMER: \_\_\_\_\_ DATE: \_\_\_\_\_

LINE NO.	10	20	30	40	50	60	70	80	90	100	110	120	130	132	140	144	150	160	LINE NO.	
1																				160
2																				159
3																				158
4																				157
5																				156
6																				155
7																				154
8																				153
9																				152
10																				151
11																				150
12																				149
13																				148
14																				147
15																				146
16																				145
17																				144
18																				143
19																				142
20																				141
21																				140
22																				139
23																				138
24																				137
25																				136
26																				135
27																				134
28																				133
29																				132
30																				131
31																				130
32																				129
33																				128
34																				127
35																				126
36																				125
37																				124
38																				123
39																				122
40																				121
41																				120
42																				119
43																				118
44																				117
45																				116
46																				115
47																				114
48																				113
49																				112
50																				111
51																				110
52																				109
53																				108
54																				107
55																				106
56																				105
57																				104
58																				103
59																				102
60																				101
61																				100
62																				99
63																				98
64																				97
65																				96
66																				95
67																				94
68																				93
69																				92
70																				91
71																				90
72																				89
73																				88
74																				87
75																				86
76																				85
77																				84
78																				83
79																				82
80																				81
81																				80
82																				79
83																				78
84																				77
85																				76
86																				75
87																				74
88																				73
89																				72
90																				71
91																				70
92																				69
93																				68
94																				67
95																				66
96																				65
97																				64
98																				63
99																				62
100																				61
101																				60
102																				59
103																				58
104																				57
105																				56
106																				55
107																				54
108																				53
109																				52
110																				51
111																				50
112																				49
113																				48
114																				47
115																				46
116																				45
117																				44
118																				43
119																				42
120																				41
121																				40
122																				39
123																				38
124																				37
125																				36
126																				35
127																				

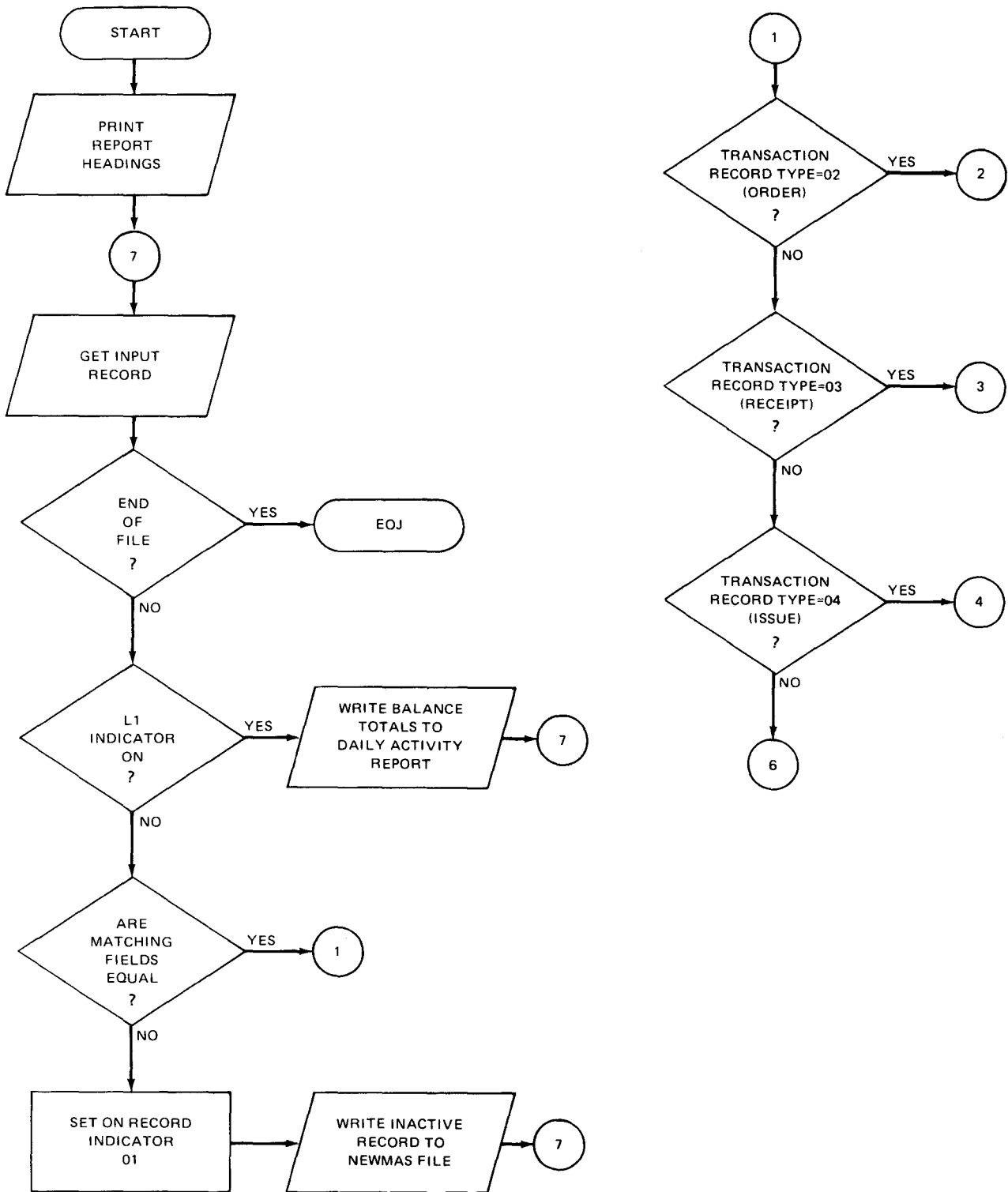


Figure B-22. Processing with Matching Records — Inventory Control (Part 1 of 2)

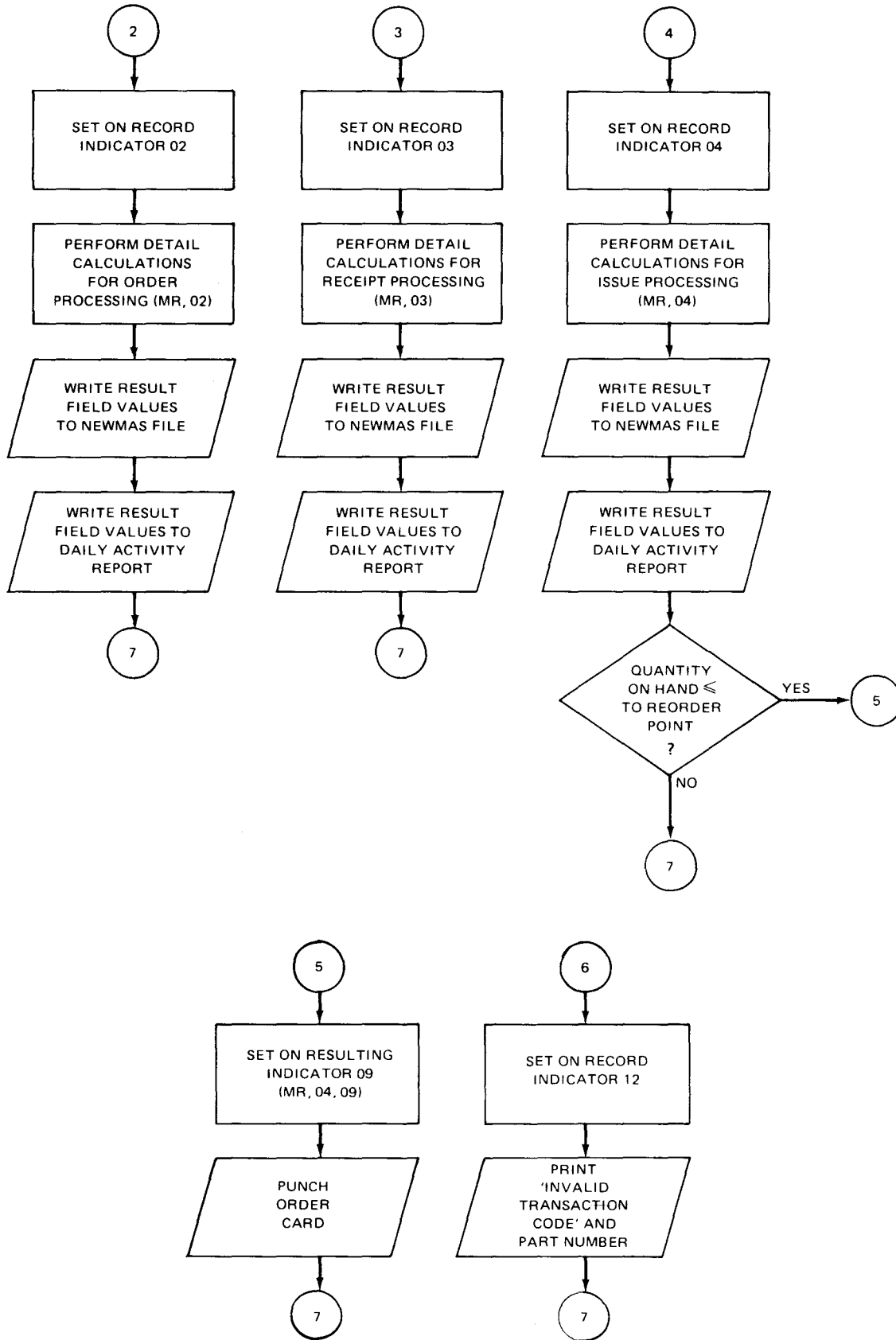


Figure B-22. Processing with Matching Records — Inventory Control (Part 2 of 2)











# OUTPUT FORMAT SPECIFICATIONS

PAGE NO.	FORM TYPE	LINE NO.	STACKER SELECT/ F-FETCH OVERFLOW		SPACE		SKIP		OUTPUT INDICATORS						FIELD NAME	DATA FORMAT P/B/L/R		CODES				ACTION		NOT USED	PROGRAM IDENTIFICATION													
			TYPE	H/D/T/E	BEFORE	AFTER	BEFORE	AFTER	AND	AND	N-NOT	N-NOT	N-NOT	N-NOT		N-NOT	N-NOT	END POSITION IN OUTPUT RECORD	EDIT CODES	B-BLANK AFTER	NEGATIVE VALUE INDICATION		COMMAS INSERTED			ZERO BALANCE TO PRINT	X	REMOVE PLUS SIGN	Y	EDIT DATE FIELD	Z	ZERO SUPPRESS						
1	2	3	5	6	7	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	37	38	39	40	43	44	45	70	71	74	75	80		
05	0.1	0	DAILYAC	H			2	0	7																													
05	0.2	0		OR																																		
05	0.3	0																										64										
05	0.4	0		H																																		
05	0.5	0		OR																																		
05	0.6	0																										17										
05	0.7	0																										40										
05	0.8	0																										52										
05	0.9	0																										70										
05	1.0	0																										88										
05	1.1	0																										110										
05	1.2	0																										121										
05	1.3	0																										128										
05	1.4	0		H																																		
05	1.5	0		OR																																		
05	1.6	0																										22										
05	1.7	0																										41										
05	1.8	0																										62										
05	1.9	0																										86										
05	2.0	0																										111										
05	2.1	0																										124										
05	2.2	0		D																																		
05	2.3	0																																				
05	2.4	0		D																																		
05	2.5	0																																				

UD1-1165 REV. 8-82

Figure B-23. Entries for Processing with Matching Records — Inventory Control (Part 5 of 7)





### **B.9.1.2. File Description Specifications**

OLDMAS, created previously (B.8), is specified as the primary input file (the controlling file). TRANS is specified as a secondary input file in fixed unblocked format that is read in from a card reader. NEWMAS is an output file that is to contain the updated master file information.

DAILYAC is used to produce a report that indicates the activity resulting from the day's transactions.

CRDPNCH is used to create a card file of orders for those items which have gone below the reorder point.

### **B.9.1.3. Input Format Specifications**

Records in the OLDMAS file are identified by indicator O1 and an M in record position 1. The entire record is defined as an 80-character field, INREC, so that the entire record can be written on the NEWMAS file at detail time by the EXCPT operation when the matching fields do not match. The individual fields are also defined so that they may be processed according to the type of transaction. L1 is the control level indicator for total time output and M1 identifies the matching fields. The type of transaction on the TRANS file is identified by a code in record position 1 (O = order, R = receipt, and I = issue) and record identifying indicators O2, O3, and O4, respectively. All other records with codes other than these (invalid records) will set indicator 12 on.

### **B.9.1.4. Calculation Specifications**

Line 010 shows that this operation is performed when there are matching records and an order transaction (MR and O2 indicators are on). The quantity from the TRANS file record (QUAN) is added to the quantity-on-order (QOO) from the OLDMAS file record and the result is placed in the new-quantity-on-order (NQOO) field for the NEWMAS file. The program then branches to line 050 where the cost of the order is extended and charged to the department that ordered the item. The extended price (AMT) will be printed in the Daily Activity Report under AMOUNT BILLED. The AMT field is then accumulated in QAMT on line 060 for the ORDERS portion of the BALANCE TOTALS on the Daily Activity Report.

Line 020 shows that this operation is performed when there are matching records and a receipt transaction (MR and O3 indicators are on). The quantity from the TRANS file (QUAN) is added to the quantity-on-hand (QOH) from the OLDMAS file record, and the result is placed in the new-quantity-on-hand (NQOH) field for the NEWMAS file. The program then branches to line 050 where the cost of the receipt is extended and charged the department that received the item. The extended price (AMT) is printed in the Daily Activity Report under AMOUNT BILLED. The AMT field is then accumulated in RAMT on line 070 for the RECEIPTS portion of the BALANCE TOTALS on the Daily Activity Report.

Line 030 shows that this operation is performed when there are matching records and an issue transaction (MR and O4 indicators are on). The quantity from the TRANS file (AMT) is subtracted from the quantity-on-hand (QOH) from the OLDMAS file record, and the result is placed in the new quantity-on-hand (NQOH) field for the NEWMAS file. On line 040,

QUAL is added to the year-to-date usage (USAGE) in the OLDMAS file. This field is written on the NEWMAS file and the Daily Activity Report under YEAR-TO-DATE USAGE. The cost of the issue is extended and charged to the appropriate department on line 050. The program then branches to line 080, where the AMT field is accumulated in IAMT for the ISSUES portion of the BALANCE TOTALS on the Daily Activity Report. On line 090, the new quantity-on-hand (NQOH) is compared to the reorder point (REOPNT) in the OLDMAS file. If NQOH is less than or equal to REOPNT, indicator 09 is set on. If indicator 09 is on, an O is placed in the OCODE field of the CRDPNCH file and a card is punched at detail output time that contains O in record position 1, the part number (MPNUM), the item description (ITEM), and the reorder quantity (REQTY).

#### **B.9.1.5. Output Format Specifications**

Page 04, lines 010 and 020 show that the inactive record (INREC) from the file OLDMAS is written on the NEWMAS file at detail time when there is a no-match condition.

Page 04, lines 040 through 120 show that these fields are written on the NEWMAS file when the MR indicator is on; that is, there is a transaction and the updated inventory record is written to the NEWMAS file. Page 04, lines 130 through 170 show that these fields are punched on an order card at detail time when the MR, 04, and 09 indicators are on; that is, there is an issue transaction and the new quantity-on-hand (NQOH) is less than or equal to the reorder point.

Page 05, lines 010 through 210 show that the Daily Activity Report heading information is printed on the first page of the report (controlled by 1P indicator) and on each succeeding page when forms overflow occurs (controlled by OF indicator).

Page 05, lines 220 through 250 show that this line is printed at detail time when a record is read from the TRANS file that does not contain a valid transaction code in record position 1. INVALID TRANSACTION CODE and the identifying part number (MPNUM) are printed.

Page 06, lines 010 through 130 show that these fields are printed on the Daily Activity Report at detail time when the MR indicator is on. This provides a printed record of each transaction that applies to a particular part number.

Page 06, lines 140 through 210, and page 07, lines 010 through 080, show that the L1 indicator controls the printing of the BALANCE TOTALS on the Daily Activity Report at total time. The amounts that are printed represent the sum of the order, receipt, and issue transactions for the day for a particular part.

#### **B.9.2. Report Format**

The format of the report produced by this program is shown in Figure B-24.

APPLICATION: \_\_\_\_\_ TYPE OF PRINTER: \_\_\_\_\_  
PROGRAMMER: \_\_\_\_\_ DATE: \_\_\_\_\_

LINE NO	10	20	30	40	50	60	70	80	90	100	110	120	130	132	140	144	150	160	LINE NO
	DAILY ACTIVITY REPORT																		
	PART NUMBER	ITEM DESCRIPTION	TRANS-ACTION	QUAN ON HAND	REORDER POINT	REORDER QUANTITY	NUMBER UNITS	UNIT PRICE	ASSEMBLY GROUP	DEPARTMENT BILLED	AMOUNT BILLED	YEAR-TO-DATE USAGE							
	XXXXXX	XXXXXXXXXXXX	XXXXXX	XXXX	XXXX	XXXX	XXXX	XXXXXX	XXXXXXXXXX	XXXXXXXXXXXX	\$XXXXXX.XX	XXXXXX	INVALID TRANSACTION CODE						
	BALANCE TOTALS												XXXXXX						
	ORDERS \$XXX,XXX.XX																		
	RECEIPTS \$XXX,XXX.XX																		
	ISSUES \$XXX,XXX.XX																		

Figure B-24. Daily Activity Report Format

### B.9.3. Entering and Compiling a Source Program from a Workstation

To enter a program from a workstation, you must follow a procedure to tie into the system. The first step of this procedure is to type LOGON followed on the same line by a series of characters that identifies you. A typical procedure is as follows:

- ① LOGON ACCNT
- ② EDT @RPG  
.  
Enter your RPG II program here  
.
- ③ @WRITE MO=PAYROL , FIL=LTFIL , SIZE=2 , VSN=PUBDSK , SAT=Y
- ④ @HALT
- ⑤ RV JC\$BLD  
.  
Enter your job control here  
.
- ⑥ RV PAYROL
- ⑦ LOGOFF

#### Explanation:

- ① The LOGON statement, plus the identification, connects your terminal to the host system. The identification in this example is your department name.
- ② This calls the general editor (EDT) and the RPG II editor, which gives you a choice of three screen formats. The format types are: formatted, for the novice programmer; positional; and free-form, for the experienced programmer. These formats act as a guide to help you write the program.
- ③ This writes the RPG II program into a file.
- ④ This indicates that you have ended programming, and terminates both the general editor and the RPG II editor.
- ⑤ This calls in a question and answer dialog that helps you build a job control stream. This stream is used to compile the program. You must give the stream a name; this name will be used in step 6.
- ⑥ This compiles the program. Next to the RUN (RV) command you must insert the name of the stream you gave in step 5.
- ⑦ This disconnects your workstation from the host system.

#### NOTE:

*For more information on the RPG II editor, see the OS/3 RPG II editor user guide/programmer reference.*





000330				SALCL	63
000340				RETNUM	70
000350				STDATE	76
000360				TITLE	96
000370	OPRNTR	H 307	OF		
000380		OR	1P		
000390					70 '***MULTIKEY'
000400					80 'FILE**'
000410		H 1	OF		
000420		OR	1P		
000430					6 'ID'
000440					24 'EMPLOYEE'
000450					54 'SOCIAL MARITAL'
000460					69 'EXEMP INS.'
000470					91 'PHONE DEPT SC'
000480					108 'RET. START'
000490					123 'TITLE'
000500		D 2	OF		
000510		OR	1P		
000520					7 'NUMBER'
000530					22 'NAME'
000540					54 'SECURITY STATUS'
000550					80 'NUMBER NUMBER'
000560					107 'NUMBER DATE'
000570		D 1	01		
000580				IDNUM	10
000590				NAME	32
000600				SSN	45 ' - - '
000610				MSTAT	51
000620				EXEMPT	59
000630				INSNUM	71
000640				PHONE	81 ' - '
000650				DEPT	87
000660				SALCL	91
000670				RETNUM	100
000680				STDATEY	110
000690				TITLE	132

Figure B-25. Creating a 5-Key MIRAM File Interactively (Part 2 of 2)

## Explanation of Figure B-25:

Line 1:

Identifies the program name as CREMLT.

Line 2:

Establishes INDATA as the interactive console (workstation) file. It is an input/primary file with a maximum record length of 97. CONSOLE is specified in columns 40 to 46 as the device.

**NOTE:**

*Record Length = Highest TO location specified in the input field description for any record of the console file.*



Lines 3 to 8:

Define MKFILE as an output file with five keys of reference stored on disk.

Lines 10 to 23:

Define INDATA file as a file with one record type referenced by function key 1 (01 in columns 19 and 20). The character 1 is placed in position 1 of each record and is referenced in the program as field IDKEY. Twelve fields will have prompts generated and displayed upon the workstation screen.

Lines 24 to 36:

Write input data from the prompt screen, positions 2 to 97, to the MKFILE disk file, positions 1 to 96.

Lines 37 to 69:

Generate a printed listing formatted for readability of user reference and audit trail.

Record 1 Data Input:

```

      1      1      1
IDNUM  N10.0 1111111111
NAME   A 20 smith,john_p.-----
SSN    N09.0 1111111111
MSTAT  A 01 m
EXEMPT N02.0 03
INSNUM N08.0 11111111
PHONE  N07.0 11111111
DEPT   N04.0 1086
SALCL  A 02 mt
RETNUM N07.0 11111111
STDATE N06.0 110482
TITLE  A 20 branch_manager-----

```

Record 2 Data Input:

```

      1      1      1
IDNUM  N10.0 2222222222
NAME   A 20 doe,john_t.-----
SSN    N09.0 2222222222
MSTAT  A 01 s
EXEMPT N02.0 01
INSNUM N08.0 22222222
PHONE  N07.0 22222222
DEPT   N04.0 1086
SALCL  A 02 b4
RETNUM N07.0 22222222
STDATE N06.0 110482
TITLE  A 20 sales_representative

```





```

00025I      AD  04  1 C4
00026I      1  10IDKEY
00027I      2  80PHONE
00028I      9  100FLDNUM
00029I     11  30 FLDATA
00030I      AE  05  1 C5
00031I      1  10IDKEY
00032I      2  80RETNUM
00033I      9  100FLDNUM
00034I     11  30 FLDATA
00035IMKFILE BB  07
00036I      1  100IDNUM
00037I     11  30 NAME
00038I     31  390SSN
00039I     40  40 MSTAT
00040I     41  420EXEMPT
00041I     43  500INSNUM
00042I     51  570PHONE
00043I     58  610DEPT
00044I     62  63 SALCL
00045I     64  700RETNUM
00046I     71  760STDATE
00047I     77  96 TITLE
00048C      SETOF
00049C      FLDNUM  COMP 0          99
00051C     99      GOTO END
00052C      SETOF          212223
00053C      SETOF          242526
00054C      SETOF          272850
00056C     01      1  SETK MKFILE
00057C     01      IDNUM  CHAINMKFILE          50
00058C     N50     EXSR UPDTE
00059C     01N50   GOTO END
00060C     02      2  SETK MKFILE
00061C     02      SSN    CHAINMKFILE          50
00062C     N50     EXSR UPDTE
00063C     01N50   GOTO END
00064C     03      3  SETK MKFILE
00065C     03      INSNUM CHAINMKFILE          50
00066C     N50     EXSR UPDTE
00067C     01N50   GOTO END
00068C     04      4  SETK MKFILE
00069C     04      PHONE CHAINMKFILE          50
00070C     N50     EXSR UPDTE
00071C     01N50   GOTO END
00072C     05      5  SETK MKFILE
00073C     05      RETNUM CHAINMKFILE          50
00074C     N50     EXSR UPDTE
00075C     01N50   GOTO END
00077C      END      TAG
00078CSR     UPDTE  BEGSR
00079CSR     FLDNUM COMP 2          21
00080CSR     21     MOVEFLDLATA  NAME  20
00081CSR     21     GOTO END2
00082CSR     FLDNUM COMP 4          22
00083CSR     22     MOVEFLDLATA  MSTAT  1
00084CSR     22     GOTO END2
00085CSR     FLDNUM COMP 5          23
00086CSR     23     MOVEFLDLATA  EXEMPT 20
00087CSR     23     GOTO END2
00088CSR     FLDNUM COMP 7          24
00089CSR     24     MOVEFLDLATA  PHONE  70
00090CSR     24     GOTO END2
00091CSR     FLDNUM COMP 8          25
00092CSR     25     MOVEFLDLATA  DEPT   40
00093CSR     25     GOTO END2
00094CSR     FLDNUM COMP 9          26

```

Figure B-26. Interactively Updating a MIRAM File Using a Console File and Automatic Screen Prompts (Part 2 of 3)

```

00095CSR 26          MOVEFLDATA  SALCL  2
00096CSR 26          GOTO END2
00097CSR          FLDNUM    COMP 11          27
00098CSR 27          MOVEFLDATA  STDATE0060
00099CSR 27          GOTO END2
00100CSR          FLDNUM    COMP 12          28
00101CSR 28          MOVEFLDATA  TITLE  20
00102CSR          END2      TAG
00103CSR          ENDSR
00104OMKFILE D      N50N99 07
001050             IDNUM     10
001060             NAME      30
001070             SSN       39
001080             MSTAT    40
001090             EXEMPT   42
001100             INSNUM   50
001110             PHONE    57
001120             DEPT     61
001130             SALCL    63
001140             RETNUM   70
001150             STDATE   76
001160             TITLE    96

```

Figure B-26. Interactively Updating a MIRAM File Using a Console File and Automatic Screen Prompts (Part 3 of 3)

#### Explanation of Figure B-26:

Line 1:

Identifies the program name as MKUPD.

Line 2:

Establishes INDATA as the interactive console file.

Lines 3 to 8:

Define MKFILE as an update/combined file with five keys of reference.

Line 9:

Defines PRNTR as the output printer file.

Lines 10 to 34:

Define the INDATA interactive console file as having five record types (01 through 05) identified by characters 1 through 5 in record position 1. Screen prompts for these record types are called and displayed on the workstation by entering function keys 1 to 5, respectively.

Lines 35 to 47:

Describe fields of MKFILE.

↓

Lines 48 to 134:

Update MKFILE and print a listing of updated records.

The following steps demonstrate how to use the sample program MKUPD to update records. (We'll update the title in record 1 and the phone number in record 3.)

1. The prompt PLEASE KEY IN FIRST COMMAND appears on the screen.
2. Press function key 1, and the prompt screen for key 1 (IDNUM) appears:

```

1      1      1,2,3,4,5      1,2,3,4,5
IDNUM  N10.0  _____
FLDNUM NO2.9  __
FLDATA  A  20  _____

```

3. Enter key 1 of the first record (1111111111), the number of the field you wish to change (12), and the new title (personnel director):

```

1      1      1,2,3,4,5      1,2,3,4,5
IDNUM  N10.0  1111111111
FLDNUM NO2.0  12
FLDATA  A  20  personnel_director__

```

4. Press function key 2, and the prompt screen for key 2 appears:

```

2      2      1,2,3,4,5      1,2,3,4,5
SSN    N09.0  _____
FLDNUM NO2.0  __
FLDATA  A  20  _____

```

5. Enter key 2 of the third record (333333333), the number of the field you wish to change (07), and the new phone number (2156666):

```

2      2      1,2,3,4,5      1,2,3,4,5
SSN    N09.0  333333333
FLDNUM NO2.0  07
FLDATA  A  20  2156666_____

```

↑



```

00023IMKFILE BB 07
00024I          1 100IDNUM
00025I          11 30 NAME
00026I          31 390SSN
00027I          40 40 MSTAT
00028I          41 420EXEMPT
00029I          43 500INSNUM
00030I          51 570PHONE
00031I          58 610DEPT
00032I          62 63 SALCL
00033I          64 700RETNUM
00034I          71 760STDATE
00035I          77 96 TITLE
00036C          SETOF          102021
00037C          SETOF          2223
00038C*** WRITE RECORD ENTERED TO PRINTER FILE ***
00039C 01          EXCPT
00040C*
00041C** CHECK FOR DUPLICATE IDENTIFICATION NUMBER **
00042C*
00043C 01          1          SETK MKFILE          SET KEY = IDNUM
00044C 01          IDNUM      CHAINMKFILE          20
00045C*
00046C** CHECK FOR DUPLICATE SOCIAL SECURITY NUMBER **
00047C*
00048C 01 20      2          SETK MKFILE          SET KEY = SSN
00049C 01 20      SSN        CHAINMKFILE          21
00050C*
00051C** CHECK FOR DUPLICATE INSURANCE NUMBER **
00052C*
00053C 01 21      3          SETK MKFILE          SET KEY = INSNUM
00054C 01 21      INSNUM     CHAINMKFILE          22
00055C*
00056C** CHECK FOR DUPLICATE RETIREMENT NUMBER **
00057C*
00058C 01 22      5          SETK MKFILE          SET KEY = RETNUM
00059C 01 22      RETNUM     CHAINMKFILE          23
00060C*
00061C** 23 ON - NO DUP KEYS - WRITE RECORD TO MKFILE **
00062C*
00063C 01 23          SETON          10
00064OMKFILE DADD      10
00065O          IDNUM          10
00066O          NAME           30
00067O          SSN            39
00068O          MSTAT          40
00069O          EXEMPT         42
00070O          INSNUM         50
00071O          PHONE          57
00072O          DEPT           61
00073O          SALCL          63
00074O          RETNUM         70
00075O          STDATE         76
00076O          TITLE          96
00077OPRNTR  H 307  OF
00078O      OR          1P
00079O          70 '**MULTIKEY'
00080O          80 'FILE**'
00081O      H 1      OF
00082O      OR          1P
00083O          6 'ID'
00084O          24 'EMPLOYEE'
00085O          54 'SOCIAL      MARITAL'
00086O          69 'EXEMP      INS.'
    
```

Figure B-27. Adding Records to a Keyed MIRAM File (Part 2 of 3)



000870				91	'PHONE	DEPT	SC'
000880				108	'RET.	START'	
000890				123	'TITLE'		
000900	D	2	OF				
000910	OR		1P				
000920				7	'NUMBER'		
000930				22	'NAME'		
000940				54	'SECURITY	STATUS'	
000950				80	'NUMBER	NUMBER'	
000960				107	'NUMBER	DATE'	
000970	E	1	01				
000980				IDNUM	10		
000990				NAME	32		
001000				SSN	45	' - - '	
001010				MSTAT	51		
001020				EXEMPT	59		
001030				INSNUM	71		
001040				PHONE	81	' - '	
001050				DEPT	87		
001060				SALCL	91		
001070				RETNUM	100		
001080				STDATEY	110		
001090				TITLE	132		
001100	D	1	N10 01				
001110					40	'INVALID RECORD - '	
001120					60	'DUPLICATE KEY VALUES'	

Figure B-27. Adding Records to a Keyed MIRAM File (Part 3 of 3)



## Appendix C. Program Testing Aids

### C.1. GENERAL

After you have written your program, you must test it to see whether it produces the results that you want. If it does not produce the required results, you must find out why and make the necessary corrections. To help you test your program, RPG II provides the following aids:

- **DEBUG Operation**

This operation allows you to write out the program indicators that are on and the contents of fields during the execution of your program.

- **DSPLY Operation**

This operation allows you to display data on or enter data via the system console during program execution.

- **EXCPT Operation**

The EXCPT operation allows you to write output records when calculations are performed.

- **Operator Control**

This option allows you to control program termination via system console type-ins when a halt indicator (H0 through H9) that you specified in your program is set on, or the H0 indicator is set on as a result of an error condition.

- **\*ERROR Field**

The \*ERROR field contains an error code when an error occurs during the execution of your program.

- **Error Analysis Dump**

A formatted error analysis dump can be provided when your program terminates during execution. This dump contains the information most likely to help you find the errors in your program.

## ■ Unformatted Dump

If you do not want an error analysis dump when your program terminates during execution, an unformatted dump can be provided instead. This dump contains the same information as the error analysis dump except that its format makes it slightly more difficult to use.

## C.2. DEBUG OPERATION

You can use the DEBUG operation to tell you what program indicators are on and what is contained in specified fields at particular points during the execution of your program. To use the DEBUG operation you must first enter a 1 in column 15 of the control card specifications form and then you include the DEBUG operation in your program on the calculation specifications form at the required point.

When the DEBUG operation is executed, an output record that shows the program indicators that are set on and the contents of factor 1, if specified, is written on the output file that you specified in factor 2. If a result field is specified, one or more records that show the contents of the result field are also written on the output file. The formats of these records are:

Type 1 Record - Factor 1 is or is not specified.

<u>Position</u>	<u>Contents</u>
1 through 6	DEBUG -
7	Blank
8 through 17	The contents of factor 1 if specified. If not, these positions contain the source program statement number for the DEBUG operation.
18	Blank
19 through 32	INDICATORS ON -
33 and 34, 36 and 37, etc.	The name of the indicators, each separated by a blank, that are set on.

Type 2 Record - Result field is specified.

<u>Position</u>	<u>Contents</u>
1 through 13	RESULT FIELD -
15 through n	The contents of the result field.

When these records are written, all numeric fields are unpacked and zero suppressed. A minus sign is written to the right of a negative field. If the result field is a table or array, an individual type 2 record is printed for each element.

Figure C-1 shows the source program listing and the resulting printed output for a program that uses the DEBUG operation. In this example, lines 007, 008, 009, 010, and 012 on the source program listing contain DEBUG operations. The records that these operations produce on the resulting printed output are indicated by the source program line number and a brace.

Line 007 in the source program shows a DEBUG operation where factor 1 and the result field are not specified. This produces one type 1 record that lists the source program line number and the indicators that are set on.

Line 008 shows a DEBUG operation, where factor 1 (the literal ABC) is specified and the result field is not specified. This produces one type 1 record that lists the contents of factor 1 and the indicators that are set on.

Line 009 shows a DEBUG operation, where factor 1 is not specified but the result field is. This produces a type 1 record and a type 2 record. The type 1 record lists the source program line number and the indicators that are set on. The type 2 record lists the contents of the result field.

Line 010 shows a DEBUG operation, where factor 1 and the result field are specified. This produces a type 1 record and a type 2 record. The type 1 record lists contents of factor 1 and the indicators that are set on. The type 2 record lists the contents of the result field.

Line 012 shows a DEBUG operation, where factor 1 is not specified and the result field is specified as the name of a 5-element array (ARRAY). This produces a type 1 record and five type 2 records. The type 1 record lists the source program line number and the indicators that are set on. Each type 2 record lists the contents of one element of the array.

### C.3. DSPLY OPERATION

You can use the DSPLY operation (7.3.2.8.5) to cause data to be displayed on the system console and also to allow you to change data in a field when your program is being executed. As you can see, you can use this operation to make "hands on" corrections to your program.

### C.4. EXCPT OPERATION

You can use the EXCPT operation (7.3.2.8.6) to cause output to be written during calculation. This operation is useful when you want to check your calculation logic and you need more information than is provided by the DEBUG operation.

```

001      01 01 H          1
002      01 02 FCARDIN 1P F 80 80          CTLDR
003      02 01 FPRNTR  0 F 120 120'      OF  PRINTER
004      03 01 E          ARRAY  5  5  6
005      03 02 ICARDIN  NS  01
006      03 03 I          1  72FIELD
007      04 01 C          8  9 ALPHA
008      04 02 C          'ABC'  DEBUGPRNTR
009      04 04 C          FIELD  DEBUGPRNTR
010      04 05 C          MOVEALPHA  ALPHA
011      04 06 C          DEBUGPRNTR  ARRAY
012      04 07 C          DEBUGPRNTR  ARRAY
013      05 01 OPRNTR  H  207  1P
014      05 02 0          100 'DEBUG TESTS'
015      05 03 0          T 1  LR
016      05 04 0          100 'END OF TEST'

```

TEST27

SOURCE PROGRAM LISTING

DEBUG TESTS

```

007( DEBUG- 007      INDICATORS ON-00 01 LO
008( DEBUG- ABC      INDICATORS ON-00 01 LO
009 { DEBUG- 009      INDICATORS ON-00 01 LO
    { RESULT FIELD-  12
010 { DEBUG- 12      INDICATORS ON-00 01 LO
    { RESULT FIELD-  Z2
012 { DEBUG- 012      INDICATORS ON-00 01 LO
    { RESULT FIELD-  Z2AAAA
    { RESULT FIELD-  BBBBbb
    { RESULT FIELD-  CCCCCC
    { RESULT FIELD-  DDDDDd
    { RESULT FIELD-  EEEEEe

```

DEBUG OUTPUT

END OF TEST

Figure C-1. Using the DEBUG Operation

## C.5. OPERATOR CONTROL

You can request the operator to control program termination if a halt occurs. The request specification can be made either at compilation time or at execution time. Whether you have specified the operator control option at compilation time or at execution time, the same sequence of messages is displayed (C.5.3), depending on the type of halt and mode of operation.

When you use operator control in OS/3 mode, 9200/9300 mode, or IBM 360/20 mode, nonfatal errors are displayed after detail output. In IBM System/3 mode, errors are displayed immediately. In all modes, the halt indicators that you set are displayed after detail output.

### C.5.1. Compilation Time Specification

You specify the operator control feature in your program by coding a blank in column 8 and a 1 in column 9 in the control card specifications form (4.2.3). This setting is then compiled with your program and remains set indefinitely.

### C.5.2. Execution Time Specification

The specification for operator control at execution time is entered as a parameter in the // SET COMREG job control statement which must be included in the control stream preceding the execution step of the RPG II generated program. The specified parameter is placed in the communications region (COMREG) of the job preamble. If an error halt occurs during execution, the RPG II generated program retrieves the COMREG setting which overrides any compilation time setting. There are three options that can be used.

- Set operator control feature on.

The format of the control statement is:

1	10	20	30
<code>// SET COMREG,C'RPGOP'</code>			

The operator control feature will be used at execution time. This statement overrides the specification in column 9 of the control card specifications form, thus permitting you to use the operator control feature at execution time even though your program had been compiled without the feature.

- Set operator control feature off.

The format of the control statement is:

```
// SET COMREG,C'RPGNOP'
```

The operator control feature will not be used at execution time. This statement overrides the specification in column 9 of the control card specifications form, thus permitting you to ignore the operator control feature at execution time even though your program had been compiled with the feature.

- Set automatic default processing.

The format of the control statement is:

```

1      10      20
// SET COMREG,C'RPGAUTOP'

```

This option bypasses the operator control feature by automatically selecting default processing. The operator control feature is used at execution time. Only those error messages for which no continue action is allowed are displayed. All other messages automatically default to the continue action and the message is not displayed.

### C.5.3. Displayed Messages

The messages that are displayed depend on whether the halt indicators (H0 through H9) specified in your program were set on or whether the H0 indicator was set on as a result of an error condition.

If a halt indicator specified in your program is set on, the following messages are displayed on the system console:

RPG028 USER SET HALT INDICATORS ARE : Hn, Hn,...,Hn

RPG031 RPGII OPERATOR CONTROL, TYPE IN AVAILABLE OPTION (0, 1, 2, 3)

If the H0 indicator is set on as a result of an error condition, the following messages are displayed on the system console.

RPG nnn explanatory text (execution time error message for particular error)

RPG031 RPGII OPERATOR CONTROL, TYPE IN AVAILABLE OPTION (0, 1, 2, 3)

In either case, the operator should type in the appropriate action; that is:

- 0  
Continue. Control is returned to the program and processing continues at the instruction immediately following the error condition.
- 1  
Bypass. The remainder of the program cycle is bypassed and the next record is read.
- 2  
Controlled termination. LR processing is performed.
- 3  
Immediate termination. Program terminates immediately.

Only options 0, 2, or 3 should be used in reply to the RPG028 message. The operator control options that can be used with the individual execution time error messages are shown in the system messages programmer/operator reference.





The error analysis dump provides you with a formatted listing of the contents of main storage when an error causes your program to terminate. This dump contains the information most likely to help you find the error in your program.

### C.7.1. Using the Error Analysis Dump

The following subsections describe the major sections of an error analysis dump and how you use them.

#### C.7.1.1. REG SAVE AREA

This section contains the contents of general registers 0 through 15 at the time your program was terminated. General register 3 contains the base address of your program.

#### C.7.1.2. \*ERROR

This is a 12-byte field that has the following format:

<u>Byte</u>	<u>Contents</u>
0	Error code
1-3	Status bytes
4-7	Address pointer 1
8-11	Address pointer 2

The error code is the code for the condition that caused the HO indicator to be set on and terminate the program. The error codes and the conditions that cause them are shown in the system messages programmer reference manual. The message that is associated with the error condition is printed to the right of the error code.

The status bytes contain the status code from the last input/output command for a DAM, SAM, or ISAM file error. The status codes for these status bytes are shown in the system messages programmer reference manual. The word STATUS is printed to the right of these bytes.

Address pointer 1 and address pointer 2 contain addresses that relate to a specific file, field, record, or instruction that caused your program to terminate. These addresses vary with the type of error. What each address points to is printed to the right of it. These address pointers are:

#### FIELD

Address of the field associated with the error condition.

**FILE DES**

Address of the file descriptor whose corresponding file caused the error condition. A file descriptor is generated for each input, update, or combined file. The format of the file descriptor is shown in the system messages programmer/operator reference, UP-8076 (current version). ←

**IORB**

Address of the input/output request block. An input/output request block is generated for each file. The format of the input/output request block is shown in the system messages programmer/operator reference, UP-8076 (current version). ←

**KEY**

Address of the key of the DAM or ISAM record that caused the error.

**NSI**

Address of the next sequential instruction that would have been executed.

**RECORD**

Address of the record that caused the error.

**RCB**

Address of the remote control block. A remote control block is generated for each remote (telecommunications) file. The format of the remote control block is shown in the system messages programmer/operator reference, UP-8076 (current version). ←

**TLF**

Address plus 16 of the table linkage field whose corresponding file caused the error condition. A table linkage field is generated for each table and array in your program. The format of the table linkage field is shown in the system messages programmer/operator reference, UP-8076 (current version). ←

**C.7.1.3. RECORD**

If a record causes an error, that record is listed in the record section of the formatted error analysis dump. ←

**C.7.1.4. LINKAGE VECTOR**

This section lists the main logic routine addresses, IORB locations, DTF address pointers, and the number of files in your program. The relative addresses of where the address of the routines are located are listed under the heading ACTUAL.

The displacement addresses from the beginning of your program are listed under the heading DISP.

The abbreviated names of the routines are listed under the heading ROUTINES. The relative addresses of the routines are listed under the heading ADDRESS.

### C.7.1.5. FLDS/CONSTS/TABLES/ARRAYS

This section contains the data fields and constants used in your program. The displacements for these entries are listed in the leftmost column. If tables or arrays are used in your program, the table linkage fields for them are also listed in this section.

### C.7.1.6. Error Analysis Dump Example

Figure C-3 is an annotated example of a typical error analysis dump.

## C.8. UNFORMATTED DUMP

An unformatted dump will be provided when your program terminates during execution and you include a `//△OPTION△DUMP` statement in the job control stream used to execute your program.

### C.8.1. Using the Unformatted Dump

If an unformatted dump is provided when your program terminates, the halt indicators show which one caused your program to terminate.

If the H0 indicator is on, check the \*ERROR field. If it contains zeros, the H0 indicator was set on by your program rather than by an error condition. If the first byte of the \*ERROR field contains an error code, refer to the system messages programmer reference manual for the meaning of the error code and possible corrective action.

If the H0 indicator is not set on, check halt indicators H1 through H9 to see which one caused the program to terminate.

#### C.8.1.1. Locating Information in an Unformatted Dump

The base address of your program is used to locate information in an unformatted dump. This address is added to the displacement address specified for the information you want to locate and the resulting sum gives you the address in dump where the information is. The contents of general register 3 contains the base (load) address of your program.

##### C.8.1.1.1. Locating an Indicator

To locate an indicator, add the displacement address in the RESULTING INDICATORS section of the source program listing to the base address of your program. If the value at the calculated address contains FO, this means the indicator is set on. If the value is 00, the indicator is off.

#### **C.8.1.1.2. Locating \*ERROR Field**

The \*ERROR field can be located by adding the displacement address for \*ERROR in the FIELD NAMES section of the source program listing to the base address of your program. The calculated address points to the beginning of the \*ERROR field. See C.7.1.2 for the format of this field.

#### **C.8.1.1.3. Locating a Data Field**

A data field can be located by adding the displacement address in the FIELD NAMES section of the source program to the base address of your program. The calculated address points to the beginning of the field.

#### **C.8.1.1.4. Locating the Table Linkage Field (TLF) for a Table or Array**

A table linkage field for a table or array can be located by adding the displacement address minus 16 in the FIELD NAMES section of the source program listing to the base address of your program. The calculated address points to the table linkage field. The format of the table linkage field is shown in the system messages programmer reference.

#### **C.8.1.2. Unformatted Dump Example**

Figure C-4 is an annotated example of an unformatted dump.

FILE WITH UNDEFINED RECORD TYPE

```

H D
001 010 FCARDIN IPEAF 80 80 CTLRDR
002 020 FPRINTER 0 F 120 120 OF PRINTER
003 S 010 ICARDIN 011 01 2 C1
004 020 I 1 80 DATA
005 S 010 OPRINTER D 1 01
006 020 0 DATA 90
007 030 0 D 07 OF
    
```

UNDEFRT

SYMBOL TABLES

RESULTING INDICATORS

ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI
000011 OF	000014 1P	000015 LR	000016 00	000017 01	00007A L0	000085 H0
000086 H1	000087 H2	000088 H3	000089 H4	00008A H5	00008B H6	00008C H7
00008D H8	00008E H9					

DISPLACEMENT FOR 01 INDICATOR (pointing to 000017 01)

DISPLACEMENT FOR H0 INDICATOR (pointing to 000085 H0)

FIELD NAMES

ADDRESS FIELD	ADDRESS FIELD	ADDRESS FIELD	ADDRESS FIELD	ADDRESS FIELD
0001C4 *ERROR	0001B8 P9\$DMP EXTRN	0001E4 DATA		

DISPLACEMENT FOR 'DATA' FIELD NAME (pointing to 0001E4 DATA)

PROGRAM POINTERS

TABLE INPUT/OUTPUT	000234
INPUT/OUTPUT INTERCEPT	000858
INPUT FIELD EXTRACTION	000238
DETERMINE RECORD TYPE	0002C8
GET INPUT RECORD	0003FC
GET LOOKAHEAD FIELDS	00050C
DETAIL CALCULATIONS	000510
TOTAL CALCULATIONS	000524
TOTAL OUTPUT	00053C
OVERFLOW OUTPUT	00055E
OVERFLOW BYPASS	00059C
HEADER/DETAIL OUTPUT	0005A8
OUTPUT FIELDS	000604
POINTER TO ADDRESSES OF DTFS	000788
PTR TO ADDRESSES OF I/O INTERFACE RTNS	000098
MAIN INPUT/OUTPUT	000858
INPUT/OUTPUT INTERFACE RTNS	000790
INPUT/OUTPUT REQUEST BLKS	000748
INITIALIZATION/WDRM AREA	000000
MASTER DRIVER	000F20

Figure C-3. Error Analysis Dump (Part 1 of 13)

UNIVAC OS/3 DISC RPG VER750728

UNDEFRT

08/01/75 08.10

PAGE 2

PROGRAM LENGTH 001078

NO ERROR NOTES IN THIS COMPILATION

COMPILATION COMPLETED ON 08/01/75 AT 08.11

UNIVAC SYSTEM OS/3 LINKAGE EDITOR  
DATE- 75/08/01 TIME- 08.11

VER750718

CONTROL STREAM ENCOUNTERED AND PROCESSED AS FOLLOWS-

/\*

LOADM UNDEFR  
INCLUDE UNDEFR

/\*

DPSCOME \*AUTO-INCLUDED\*  
P9SDMP \*AUTO-INCLUDED\*  
P\*SLVCT \*AUTO-INCLUDED\*

Figure C—3. Error Analysis Dump (Part 2 of 13)

\*DEFINITIONS DICTIONARY\*

SYMBOL.	TYPE.	PHASE.	ADDRESS.	SYMBOL.	TYPE.	PHASE.	ADDRESS.	SYMBOL.	TYPE.	PHASE.	ADDRESS.
DP\$COM0	ENTRY	ROOT	00300030	DP\$COM1	ENTRY	ROOT	00300000	DP\$COM2	ENTRY	ROOT	00000000
DP\$COM3	ENTRY	ROOT	00000000	DP\$COM4	ENTRY	ROOT	00000000	DP\$COM5	ENTRY	ROOT	00000000
DP\$COM6	ENTRY	ROOT	00300000	DP\$COM7	ENTRY	ROOT	00300000	KE\$ALP	ENTRY	ABS	00002A68
KE\$RES	ENTRY	ABS	00002A68	P*\$	CSECT	ROOT	00000480	P*\$DBGLT	ENTRY	ROOT	00001490
P*\$EXMT	ENTRY	ROOT	00301490	P*\$INSL	ENTRY	ROOT	00001450	P*\$LVCT	ENTRY	ROOT	000015A8
P*\$TAB	CSECT	ROOT	00001490	PR\$IOE	CSECT	ROOT	00300000	PRINTER	ENTRY	ROOT	000020E0
PRNTR	ENTRY	ROOT	00300730	PRNTRC	ENTRY	ROOT	00000762	P\$DMP	ENTRY	ROOT	00000480
UNDEFR	CSECT	ROOT	000019F0								

\*\*PHASE STRUCTURE\*\* HEX BYTES REPRESENTED BY EACH DASH - 70

I 00\*2A68\*

\*\* ALLOCATION MAP \*\*

PHASE NAME	TRANS	ADDR	FLAG	LABEL	TYPE	ESID	LNW	ORG	HIADDR	LENGTH	OBJ	ORG
UNDEFROD	MODE	-	ROOT				00000000		00002A67	00002A68		
*** START OF AUTO-INCLUDED ELEMENTS -												
- 03/26/74 01.09 -												
				PR\$IOE	OBJ							
				PR\$IOE	CSECT	02	00000000		000004A9	000004AA	00000000	
				DP\$COM7	ENTRY	02	00000000				00000000	
				DP\$COM0	ENTRY	02	00000000				00000000	
				DP\$COM1	ENTRY	02	00000000				00000000	
				DP\$COM6	ENTRY	02	00000000				00000000	
				DP\$COM2	ENTRY	02	00000000				00000000	
				DP\$COM5	ENTRY	02	00000000				00000000	
				DP\$COM4	ENTRY	02	00000000				00000000	
				DP\$COM3	ENTRY	02	00000000				00000000	
- 75/07/31 05.32 -												
				P*\$	OBJ							
				P*\$	CSECT	01	00000480		00001480	00000FDE	00000000	
				PRNTR	ENTRY	01	00000730				00000280	
				PRNTRC	ENTRY	01	00000762				00000282	
				P\$DMP	ENTRY	01	00000480				00000000	
- 75/07/23 23.17 -												
				P*\$TAB	OBJ							
				P*\$TAB	CSECT	01	00001490		000019E8	00000559	00000000	
				P*\$DBGLT	ENTRY	01	00001490				00000000	
				P*\$LVCT	ENTRY	01	000015A8				0000011E	

Figure C-3. Error Analysis Dump (Part 3 of 13)



	P*INST	ENTRY	01	00001490		00000000
	P*SEXMT	ENTRY	01	00001490		00000000
*** END OF AUTO-INCLUDED ELEMENTS - - 75/08/01 08.11 -	UNDEFR	OBJ				
	UNDEFR	CSECT	01	000019F0	00000000	00000000
	PRINTER	ENTRY	01	000020E0		000006F0
000019F0						

LOAD ADDRESS

FLAG CODES -

B - BLK DATA CSECT	D - AUTO-DELETED	E - EXCLUSIVE 'A' REF	G - GENERATED EXTRN	I - INCLUSIVE 'V' REF
M - MULTIPLY DEFINED	N - NOT INCLUDED	P - PROMOTED COMMON	U - UNDEFINED REF	V - VCON ITEM

\*ANY OTHER CODES REPRESENT PROCESS ERRORS\*

LINK EDIT OF 'UNDEFR' COMPLETED  
DATE- 75/08/01 TIME- 08.12  
ERRORS ENCOUNTERED- 0000

I THIS IS A VALID DATA RECORD

VALID DATA OUTPUT

RP61 ERROR ANALYSIS DUMP

REG SAVE AREA

GR 0-7 00000001 80001094 00002138 000019F0 030029F0 000039F0 000049F0 000019F0

GR 8-F 50001F02 000027C0 000027C0 000019F0 500004B2 000027C0 60002A2E 000004B0

GR-3 BASE ADDRESS OF RPG II OBJECT PROGRAM

Figure C-3. Error Analysis Dump (Part 4 of 13)

```

*ERROR
ACTUAL  DISP
0018B4  01C4      A      RP6001  UNDEFINED RECORD TYPE
0018B6  01C6      000000  STATUS
0018B8  01C8      00000000
0018BC  01CC      00002138      1CR3

```

```

INDICATORS ON ARE-
00 LO MO UJ

```

INVALID DATA RECORD

```

RECORD
002010                                E3C8C9E2 40C9E240 C1D540C9 D5E5C1D3      THIS IS AN INVAL
002020      C9C440C4 C1E3C14D D9C5C3D6 D9C44040 40404040 40404040 40404040 40404040  ID..DATA.RECORd.....
002040 TO 002060 SAME AS ABOVE

```

```

LINKAGE VECTOR
ACTUAL  DISP      ROUTINE      ADDRESS
001A8C  009C      TABLE-I/O  00001C24
001A90  00A0      I/O-INTERCEPT  00002548
001A98  00A8      MOVE-IN-DATA  00001C28
001AA0  00B0      RECORD-TYPE  00001C88
001AA8  00B8      GET-IN-RECORD  00001DEC
001AAC  00BC      LOOK-AHEAD  00001EFC
001AB4  00C4      DETAIL-CALCS  00001F00
001AB8  00C8      TOTAL-CALCS  00001F14
001ABC  00CC      TOTAL-OUTPUT  00001F2C
001AC4  00D4      OVRFLW-LINE-0  00001F46
001AC8  00D8      OVRFLW-BYPASS  00001F8C
001ACC  00DC      HJR/DET-LINES  00001F98
001AD0  00E0      OUT-LINE-TBL  00001FF4
001AD4  00E4      DTF-ADDR-PTR  00002178
001AD8  00E8      IO-INTERFACE  00002788
001ADC  00EC      MAIN-I/O  00002548
001AE0  00F0      REPORT-FILE  00002180
001AE4  00F4      I/O-REQ-BLKS  00002138
001B0C  011C      WORKAREA  000027C0

```

```

WORK AREA
0027C0
0000  000027C0 6000291A 000027C0 50001EF4 41546001 40001DEE 18DF4110 30984100
00F0  003E5821 00001A23 000029F0 000039F0 000049F0 000019F0 5010D294 960CD294
0E10  9801D290 0A115810 311C4100 00014120 000841E0 308E41F0 00F04200 005F9180

```

Figure C-3. Error Analysis Dump (Part 5 of 13)

RPG1

ERROR ANALYSIS DUMP

```

OE30      10004780 D06A42F2 E0008900 00014E20 035AD70B 31C431C4 4180D286 50B031A0
OE50      180058F0 309C05EF 18005870 315058F0 30EC05EF 47F0D150 40C5D5E3 09E840D6
OE70      C640E540 C9E240C1 E2E2E4D4 C5C44800 70F0F2F4 C2D3D6C3 0240D3C5 05C7E3C8
OE90      404DC3D6 03E240F2 F060F2F3 5D40C9E2 40C9D5E5 C1D3C9C4 4006D940 03C5E2E2
OE80      40E3C8C1 0540D9C5 C3D6D9C4 40D3C5D5 C7E3C848 40C2D3D6 C3D240D3 C5D5C7E3
OE00      C840C9E2 000022DC E4D4C5C4 700024E6 700024E6 00002061 000020E0 00002178
OEFO      000019F0 00002158 00000079 00002061 00002458 0000278E 00000001 0000235D
OF10      00002062 00001FF4 000027C0 600029F6 58703150 58F030DC 05EF9200 3014D609
OF30      30E53085 4770D242 0200D299 30E45870 315058F0 30B805EF 0200D29A 3084D200
OF50      3084D299 95F03015 4780D1CC 95F03078

```

FLODS/CONSTS/TABLES/ARRAYS

```

0184      00000000 00000480 00000000 00000000 00000000 C1000000 00000000 00002138
0100      00000000 00000000 00000000 00000000 00000000 E3C8C9E2 40C9E240 C1D540C9
01F0      05E5C1D3 C9C440C4 C1E3C140 09C5C3D6 09C44040 40404040 40404040 40404040
0210      40404040 40404040 40404040 40404040 40404040 40404040 40404040 40404040
0230      40404040

```

```

.....A.....
.....THIS IS AN I
NYALID DATA RECORD.....
.....

```

TABLE-I/O

```

001C24      07FE0000

```

I/O-INTERCEPT

```

002548      05F09042 F1AE187F 41D071EE 582030E4 59007236 47207096
002560      584030F4 D703723A 723A0501 723A3192 47807090 91404019 4710705A 95104019
002580      4780705A 0401401A 401A4780 705AD201 704C401A 95F00000 4780705A 96014019
0025A0      92224013 51014019 47107080 91404019 47107080 95104019 47807080 58102000
0025C0      45E0712A 05EF45E0 71644120 20044140 40209A01 723A47F0 70201BFF 47F070DA
0025E0      18409101 40194713 70089180 401647E0 70B69182 400E4780 70B658F0 312405EF
002600      18114310 400D0610 89100002 58121000 45E0712A 05EF45E0 71645040 31CC47F0
002620      70DA1BFF 12FF477D 71009180 401647E0 70F69500 400E477D 70F658F0 312405EF
002640      984271AE 07FE47F0 712A50F0 712692F0 30850201 31C57126 020031C7 712941E0
002660      71AE984D E00098F2 E02C58E0 31A007FE 00000000 12004770 71389220 400E47F0
002680      71445900 7236477D 71449240 400E9120 401947E0 715458F0 30F047F0 71621BFF
0026A0      43F0401E 588033E8 58FF8000 07FE1855 138B9120 401647E0 71AA58E0 31BC9240
0026C0      80009500 40084780 71944350 400892FF 71A14480 71A04188 01004650 71844350
0026E0      40090650 06504250 71A1D200 80018000 940F4016 07FE0000 000029F0 000039F0
002700      000049F0 000019F0 50001FD2 000027C0 000027C0 000019F0 00001FF4 000027C0
002720      50002A1E 5000254A 00000001 80001094 00002138 00000000 00000000 00000000
002740      00000000 700025C6 00002180 00000001 000020E0 0000217C 000019F0 00002158
002760      00000000 000049F0 5000254A 0000278E 000027C0 000027C0 00000000 00001FF4
002780      00000001 00020000

```

```

.....U.....
.....P.....N.....
.....M.....K.....
.....Q.....
.....E.....K.....G.....
.....2.....
.....O.....O.....
.....K.....
.....F.....
.....

```

Figure C-3. Error Analysis Dump (Part 6 of 13)

MOVE-IN-DATA

001C28		05805810	805E1211	078E5881	00041288	078E1A83	589030F4	.....4	
001C40	4A910002	58290000	50203194	18985A90	31AC05F8	9CABF038	43C0F04C	D201F010	.....8..0..D..K..0..
001C60	C0024110	00001883	433C0001	9502C000	4780F0E6	9580C008	4770F050	9500C00A	.....K.....0W.....0.....
001C80	4770F11E	47F0F050	80001094	06000000	00000000	07000700	00000000	00000000	..J..00.....
001CA0	00081C00	D24F31E4	200007FE	B0C46DE0	92003017	07FE0000			.....K..U.....

RPG I

ERROR ANALYSIS DUMP

RECORD-TYPE

001C88							50E0000C	58290000	.....
001CC0	50203194	D705F020	F020D703	10101010	9047D028	070005E1	00000000	00009847	....P..0..0..P.....
001CE0	0028188E	18909502	80044770	80320200	900F8004	47F08032	985631AC	1A541A64	.....K.....0.....
001D00	987831AC	1A761A8E	07F4960F	10005800	311C58ED	000C07FE	05A0D500	F0021016	.....4.....N..0..
001D20	4770A018	45C0A05A	D2001015	F00247F0	E3069108	F0034710	A02AD500	1015F002	.....K..0..0.....N..0..
001D40	4780E006	58B10010	41339001	18884381	03161A88	91108000	4780A04A	45C0A05A	.....
001D60	47F0A000	92F03085	92C331C4	501031CC	47F0E006	43B10016	418B0001	42810016	..0..0..C..D.....0.....
001D80	D5001016	1017070C	92011016	07FC0000	45110020	8F1C0000	03000284	00001A75	N.....
001DA0	00000000	0000028C	0E010101	00000000	454E0020	95C92001	4770402C	41A03017	.....I.....
001DC0	96F0A000	50A10008	47F0401E	800E0000	0284D203	1004401A	45FE0040	80040160	..0.....D.....K.....
001DE0	41A30085	50A10008	47F0E006						.....0..

GET-IN-RECORD

001DEC			058047FD	8310800C	00000000	000002C0	00E0181D		...0.....-..
001E00	5800311C	50100008	50E00004	D7083078	307847FD	809E58C0	30F44AC1	0002582C	.....P..N..#..0.....4..A..
001E20	00005020	31945890	809A1A93	58A10008	96F0A000	41F03085	19AF4770	805492C1	.....0..0.....A
001E40	31C458A0	30A84AA0	809E501A	000058F1	000C12FF	478C8078	1AF358C0	308458AC	..D.....1.....3.....
001E60	016C1A A3	05EF58ED	000458DD	000807FE	0700801A	000003A0	00000000	F0F0F0F0	..#.....0000
001E80	F0F0F0F0	F0F0F0F0	00000000	5810808E	1A135803	30F44A01	00061890	58F0800A	00000000.....4.....0..
001EA0	1AF305EF	9200900E	58F030A0	05EF95F0	30854770	800A58A0	30A84AA0	800E1BEE	..3.....0.....0.....
001EC0	50EA0000	47F08078	95229013	477080F8	41110004	92F01001	96F03015	D2083078	.....0.....8.....0..0..K..#
001EE0	808E47FD	80785083	001458FD	308041FF	000005EF	588D0014	47F08028		...0.....0.....0.....

LOOK-AHEAD

001EFC						07FE05B0			...N..
--------	--	--	--	--	--	----------	--	--	--------

Figure C-3. Error Analysis Dump (Part 7 of 13)

DETAIL-CALCS

001F00 05805890 311C90DE 914898AD 312098DE 914807FE

TOTAL-CALCS

001F14 05805890 311C90DE 914898AD

001F20 312098DE 9148419D 00FF07FE

TOTAL-OUTPUT

001F2C 58A0311C 90DEA000 582030F4 580030AD 58C030E0

001F40 98DEA000 07FE

OVRFLW-LINE-0

001F46 58AD 311C90DE A0005820 30F45800 30A058C0 30E058B0 30D80607

001F60 300C300C 07880580 95F03011 47708024 9207202F 589C0004 1A9305E9 9202202E

001F80 41020020 05E00200 30112033

OVRFLW-BYPASS

001F8C 05809200 203398DE A00007FE

RP61

ERROR ANALYSIS DUMP

HDR/DET-LINES

001F98 58A0311C 90DEA000

001FA0 582030F4 580030AD 58C030E0 92002033 058095F0 30174770 801E9201 2032589C

001FC0 00001A93 05E99202 202E4102 002005E0 058047F0 800E96F0 800198DE A00007FE

001FE0 02003011 203398DE A00007FE 98DEA000 07FED600

OUT-LINE-TBL

001FF4 0000060C 00000618 58820020

002000 024F800A 31E407FE 07FE4110 000545E0 E3C2C9E2 40C9E240 C10540C9 05E5C103

002020 C9C440C4 C1E3C140 09C5C3D6 09C44040 40404040 40404040 40404040 40404040

002040 TO 002060 SAME AS ABOVE

002060 45404040 40404040 40404040 40404040 40404040 40404040 40404040 40404040

002080 TO 0020C0 SAME AS ABOVE

0020C0 40404040 40404040 40404040 40404040 40404040 40404040 40404370 800245E0

0020E0 04008068 00000000 00000000 0000211C 00002106 00000000 00000400 0709C9D5

002100 E3C50940 00400E88 000024FA 000104FD F6200000 00000000 00002500 090020E2

002120 00000078 002C2400 84082004 00780009 00002062 45E004E4

DTF-ADDR-PTR

002178 000019F0 000020E0

.....  
.....  
.....4.....  
.....4.....00.  
.....0.....Z.....  
.....K.....  
.....  
.....  
.....0.....

.....  
.....4.....0.....  
.....Z.....0.....  
K.....0.....  
.....  
K....U.....N..THIS IS AN INV AL  
ID DATA RECORD.....  
.....  
.....  
.....PR IN  
TER.....0E.....  
.....MU  
.....0.....

Figure C-3. Error Analysis Dump (Part 8 of 13)

IO-INTERFACE										
002788		00002180	000019F0	00002308	000019F0	00002458	000019F0			.....0...0...0.....0
0027A0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	00002548	...0...0...0...0...0...0...0...
MAIN-1/0										
002548		05F09042	F1AE187F	41D071EE	582030E4	59007236	47207096			.0..1.. .....U.....
002560	584030F4	D703723A	723A0501	723A3192	47807090	91404019	4710705A	95104019		...4P.....N.....
002580	4780705A	D401401A	401A4780	705A0201	704C401A	95F00000	4780705A	96014019		... M..... H.....0.....
0025A0	92224013	91014019	47107080	91404019	47107080	95104019	47807080	58102000		.....
0025C0	45E0712A	05EF45E0	71644120	20044140	40209A01	723A47F0	702018FF	47F0700A		.....0.....0..
0025E0	18409101	40194710	70D89180	401647E0	70B69182	400E4780	70B658F0	312405EF		.....Q.....0.....
002600	1B114310	400D0610	89100002	58121000	45E0712A	05EF45E0	71645040	31CC47F0		.....0.....
002620	70DA18FF	12FF4770	71009180	401647E0	70F69500	400E4770	70F658F0	312405EF		.....6.....6..0.....
002640	984271AE	07FE47F0	712A50F0	712692F0	30850201	31C57126	020031C7	712941E0		.....0...0...0...K...E...K...6.....
002660	71AE984D	E00098F2	EC2C58E0	31A007FE	00000000	12004770	71389220	403E47F0		.....2.....0.....
002680	71445900	72364770	71449240	400E9120	401947E0	715458F0	30F047F0	716218FF		.....0...0...0.....
0026A0	43F04018	588030E8	58FF8000	07FE1855	19889120	401647E0	71AA5880	31BC9240		.0...Y.....
0026C0	80009500	40084780	71944350	400892FF	71A14480	71A04188	01004650	71844350		.....
0026E0	40090650	06504250	71A1D200	80018000	940DF401E	07FE0000	000029F0	000039F0		.....K.....0...0..
002700	000049F0	000019F0	50001F02	000027C0	000027C0	000019F0	00001FF4	000027C0		...0...0...K...0...4.....
002720	50002A18	5000254A	00000001	80001094	00002138	00000000	00000000	00000000		.....
002740	00000000	700025C6	00002180	00000001	000020E0	0000217C	000019F0	00002158		...F.....0.....
002760	00000000	000049F0	5000254A	00002788	000027C0	000027C0	00000000	00001FF4		.....0.....4.....
002780	00000001	00020000								.....
REPORT-FILE										
002180	90EC000C	5000F160	4100F15C	187F9502	400E4780	702A5880	30E818FF	43F04018		.....1-..1*.....Y...0..

RPG1 ERROR ANALYSIS DUMP										
0021A0	58F8F000	05EF47F0	71429511	400C4770	703841F0	725805EF	58004000	06005000		.80...0.....0.....
0021C0	40000201	71A44008	189995FF	400F4780	70784390	400F9511	400C4770	70E441F0		..K.....
0021E0	72A405EF	41A971AC	439A0000	41500001	45E071F0	92FF400F	95FF4010	478070A6		.....0.....
002200	43904010	9511400C	47707092	41F07208	05EF41A9	71BC439A	00004150	000145E0		.....0...Q.....
002220	71F092FF	401095FF	40114780	71024390	40119511	400C4770	70C041F0	72A405EF		.0.....0.....0.....
002240	41A971CC	439A0000	485071A4	41505001	45E071F0	95FF4012	478070FE	43904012		.....0.....
002260	9511400C	477070EE	41F07208	05EF41A9	71BC439A	00004150	000145E0	71F047F0		.....0...Q.....0...0..

Figure C-3. Error Analysis Dump (Part 9 of 13)

002280	713695FF	4012477D	71101899	47F07114	43904012	9511400C	47707122	41F072D8	.....0.....0..Q
0022A0	05EF41A9	71DC439A	00004850	71A44150	500145E0	71F0D203	400F71A8	02014008	.....OK.....M...
0022C0	71A44100	00015A0D	40005000	400058D0	716058E0	000C980C	001407FE	00000000	.....-.....
0022E0	00002738	0000000D	400021A6	00002458	00000001	000020E0	0000217C	000019F0	.....@.....0
002300	00002158	0000000D	000049F0	00002180	00002788	000027C0	000027C0	00000000	.....0.....
002320	00001FF4	00780001	FFFFFFF	27212223	24252627	28292A2B	2C2D2E2F	51515253	..4.....
002340	54555657	58595A59	5C5D5E5F	17111213	14151617	18191A1B	1C1D1E1F	10010203	.....59a.....
002360	04050607	08090A0B	0C0D0E0F	00000000	50E07250	40504008	58604000	42906000	.....-.....
002380	02037254	400058E0	30E818FF	43F04018	58F8F000	05EF12FF	4770714E	95FF400F	M.....Y.....0.....80.....
0023A0	4770722C	55FF4010	4780723C	58604000	58A07254	485071A4	44507248	58E07250	.....-.....
0023C0	07FE0000	0000000D	02006000	A0000000	9000228E	000020E1	90EC000C	187F58F0	.....K.....-...../.....0
0023E0	311C41F0	F10050D0	F00418DF	18FF9540	400E4780	705E9520	400E4770	702C0203	..01...D.....M.
002400	4000401C	4180706C	58104000	074F1000	13004100	00010A1E	12000748	12004770	.....?.....P.....
002420	70529222	401347F0	705E5900	707C4770	705E41F0	00FF5800	0004580C	000C980C	.....0.....0.....
002440	001407FE	18F041F0	F00092E5	31C447F0	705E902E	00FFFFFF	90EC000C	187F58F0	.....0.00..V..D..0.....0
002460	311C41F0	F10050D0	F00418DF	9520400E	47707036	410070A2	50001028	410070A8	..01...D.....
002480	50001038	0A2645E0	703E47F0	70529540	430E4770	70440A27	47F07092	9502400E	.....0.....0.....
0024A0	47707092	51021043	47E0707C	48504008	41505004	58604000	486070EC	07036000	.....-.....P.-.
0024C0	60004256	00019220	103158F0	133405EF	47F0708E	91041048	47E0708E	92201031	-.....0.....0.....
0024E0	58F01034	05EF45E0	703E18FF	58000004	58E0000C	980C0014	07FE96F0	401307FE	0.....0.....0.....
002500	18FF9584	10324780	709443F0	103292D8	31C447F0	70949102	104847E0	70CA4160	.....0.....Q..D..0.....
002520	60049240	60004850	40084850	70EE4740	73DE4450	70E45060	400007FE	020060D1	-.....-.....U.-.....K.-.
002540	60000600	00040002							-.....

I/O-REQ-BLKs

002138							00002010	00000000	.....
002140	00500000	000140FF	FFFFFF00	00000011	08000000	00002010	000020E3	00000000	.....
002160	00780000	000240FF	FFFFFF00	00000011	10200000	00000000			.....

JOB DUMP-USERS REGION

0019F0					05F058FD	F00607FF	000027C0	00000000	.....0.00.....
001A00	00000000	0000F000	00000000	00000000	03000000	30300000	00000000	00000000	.....0.....
001A20 TO 001A60	SAME AS ABOVE								
001A60	00000000	00000000	0000F000	00000000	03000000	30F00000	00000000	000000F0	.....0.....0.....0.....0
001A80	00000000	00000000	000019F0	00001C24	00002548	000019F0	00001C28	000019F0	.....0.....0.....0.....0
001AA0	00001C8E	000019F0	00001DEC	00001EFC	030019F0	00001F00	00001F14	00001F2C	.....0.....0.....
001AC0	000019F0	00001F4E	00001F8C	00001F98	00001FF4	00002178	00002788	00002548	..0.....4.....
001AE0	00002180	00002138	000019F0	000019F0	000019F0	300019FC	000019F0	000019F0	.....0.....0.....0.....0.....0
001B00	000019F0	000019F0	000019F0	000027C0	000019F0	000019F0	000019F0	000019F0	..0..0..0..0..0..0..0..0..0..0
001B20	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	..0..0..0..0..0..0..0..0..0..0
001B40	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	..0..0..0..0..0..0..0..0..0..0
001B60	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	..0..0..0..0..0..0..0..0..0..0

Figure C-3. Error Analysis Dump (Part 10 of 13)

001B80	00000002	00002010	00000000	00000000	00002A46	00000000	1C404040	00001000	.....
001BA0	00002000	00000003	00000480	00000000	00000000	C1000000	00000000	00002138	.....A.....
001BC0	00000000	00000003	00000000	00000000	00000000	E3C8C9E2	40C9E240	C10540C9	.....THIS IS AN I
001BE0	05E5C1D3	C9C440C4	C1E3C140	D9C5C3D6	D9C44040	40404040	40404040	40404040	.....
001C00	TO 001C20	SAME AS ABOVE							.....
001C20	40404040	07FE0000	05805810	805E1211	078E5881	00041288	078E1A83	589030F4	.....4
001C40	4A910002	58290000	50203194	18985A90	31AC05F8	90ABF038	43C0F04C	D201F010	.....8..O..O..K..O..
001C60	C0024110	00001888	438C0003	9502C000	4780F0E6	9580C008	4770F050	9500C00A	.....K...OW...O.....
001C80	4770F11E	47F0F053	80301094	00000003	00000000	37000700	00000000	00000000	..1..00.....
001CA0	00081C00	024F31E9	200007FE	8004E0E0	92003017	07FE0000	50E0000C	58290000	.....K..U.....
001CC0	50203194	0705F020	F0200703	10101010	93470028	070005E1	00000000	00009847	.....P..O..O..P.....
001CE0	0028188E	18909502	80044770	80320200	900F6004	47F08032	985631AC	1A541A64	.....M.....D.....
001D00	987831AC	1A761A86	07F4960F	100058D0	311C58E0	000C07FE	05A0D500	F002101E	.....4.....N..O...
001D20	4770A018	45C0A05A	02001015	F00247F0	E0069108	F0034710	A02A0500	1015F002	.....K...O..O.....N..O..
001D40	4780E006	58810010	41339001	192843E1	00161A88	91109000	4780A04A	45C0A05A	.....
001D60	47F0A000	52F03085	92C331C4	501031CC	47F0E006	43810016	41880001	4281001E	..O...O...C..D...O.....
001D80	0500101E	101707DC	9201101E	07FC0000	45110020	3F1C0000	00000284	00001A75	N.....
001DA0	00000000	000002EC	0E010101	00000000	45EE0020	95C92001	4770402C	41A03017	.....I.....
001DC0	96F0A000	50A10008	47F0401E	80060000	02840203	1094401A	45FE0040	80040160	..O.....D.....K.....
001DE0	41A30085	50A10008	47F0E00E	058047F0	8310600C	00000000	000002C0	00601810	.....O.....O.....
001E00	5800311C	50100008	50E00004	07083073	307847F0	809E58C0	30F44AC1	0002582C	.....P..N..N..O...4..A....
001E20	0000502C	31945890	809A1A93	58A10008	96F0A000	41F03085	19AF4770	805492C1	.....O...O...A.....
001E40	31C458A0	30A84AA0	800E501A	000058F1	000C12FF	47808078	1AF358C0	308458AC	..O.....1.....3.....
001E60	01EC1AA3	05EF58E0	000458D0	000807FE	0700601A	000003A0	00000000	F0F0F0F0	..2.....0000.....
001E80	F0F0F0F0	F0F0F0F0	00000000	5E108086	1A135800	30F44A01	000E1850	58F0800A	00000000.....4.....O..
001EA0	1AF305EF	9200900E	58F030A0	05EF95FD	30854770	80DA58A0	30A84AA0	800E18EE	..3.....O.....O.....
001EC0	50EA0000	47F08078	95229013	4770807E	41110004	92F01001	96F03015	02083078	.....O.....8.....O...O..K..N
001EE0	808E47F0	80785080	001458F0	306041FF	000005EF	58800014	47F08028	07FE0580	.....O.....O.....N.....
001F00	05805890	311C90DE	914898A0	312058DE	514807FE	05805890	311C90DE	914898A0	.....
001F20	312098DE	51484190	00FF07FE	58A0311C	90DEA000	582030F4	58D030A0	58C030E0	.....4.....4.....
001F40	98DEAG00	07FE58A0	311C90DE	A0005820	30F45800	30A058C0	30E058E0	30D8D607	.....4.....4.....QO..
001F60	300C300C	07880580	95F03011	47708024	9207202F	589C0004	1A9305E9	9202202E	.....O.....Z.....
001F80	41020020	05E0D200	30112033	05809200	203398DE	A00007FE	58A0311C	90DEA000	.....K.....
001FA0	582030F4	58D030A0	58C030E0	92002033	058055FD	30174770	801E9201	2032589C	.....4.....O.....
001FC0	00001A93	05E99202	202E4102	002005E0	058047F0	800E96F0	800198DE	A00007FE	.....2.....O...O.....
001FE0	02003011	203398DE	A00007FE	98DEA000	07FED600	0000060C	00000618	58B20020	K.....O.....

Figure C-3. Error Analysis Dump (Part 11 of 13)



002000	D24F300A	31E407FE	07FE4110	030545E0	E3C8C9E2	40C9E240	C10540C9	35E5C1D3		W.....U.....N..THIS.IS.AN.INVALID
002020	C9C440C4	C1E3C140	09C5C3D6	09C44040	40404040	40404040	40404040	40404040		ID.DATA.RECORD.....
002040	TO 002060	SAME AS ABOVE								
002060	45404040	40404040	40404040	40404040	40404040	40404040	40404040	40404040		.....
002080	TO 0020C0	SAME AS ABOVE								
0020C0	40404040	40404040	40404040	40404040	40404040	40404040	40404040	800245E0		.....
0020E0	040080E8	00000000	00000000	0000211C	00002106	00000000	00000400	0709C905		.....PRIN
002100	E3C53940	00400E88	000024FA	000104F0	F6200000	00000000	00002500	09002062		TER.....06.....
002120	00000078	002C2400	84082004	00780009	00002062	45E004E4	00002010	00000000		.....MU.....
002140	00503000	000140FF	FFFFFF00	00000011	08000000	00002010	00002063	00000000		.....
002160	00780000	000240FF	FFFFFF00	00000011	10200000	00000000	000019F0	000020E0		.....0.....
002180	90EC000C	5000F160	4100F15C	187F9502	400E4780	702A588C	30E818FF	43F0401E		.....1-1*.....
0021A0	58F8F000	05EF47F0	71429511	400C4770	733841F0	725805EF	58004000	06005000		..80.....0.....0.....
0021C0	40000201	71A4400E	189995FF	400F4780	73784390	400F9511	400C4770	706441F0		..K.....
0021E0	72A405EF	41A971AC	439A0000	41500001	45E071F0	92FF400F	95FF4010	478070AE		.....0.....
002200	43904010	9511400C	47707092	41F072D8	05EF41A9	713C439A	00004150	000145E0		.....0..Q.....
002220	71F092FF	401095FF	40114780	71024390	40119511	400C4770	70C041F0	72A405EF		..0.....0.....

RPGI

ERROR ANALYSIS DUMP

002240	41A971CC	439A0000	485071A4	41505001	45E071F0	95FF4012	478070FE	43904012		.....0.....
002260	9511400C	477070EE	41F072D8	05EF41A9	713C439A	00004150	000145E0	71F047F0		.....0..Q.....0..0
002280	713695FF	40124770	71101899	47F07114	43904012	9511400C	47707122	41F072D8		.....0.....0..Q
0022A0	05EF41A9	710C439A	00004850	71A44150	500145E0	71F00203	400F71A8	02014008		.....0K.....K..
0022C0	71A44100	00015A00	40005000	40005800	716058E0	000C980C	001407FE	00000000		.....
0022E0	00002138	00000000	400021AE	0000245E	00000001	000020E0	0000217C	000019F0		.....a.....0
002300	00002158	00000000	000049F0	00002180	00002788	000027C0	000027C0	00000000		.....0.....
002320	00001FF4	00780001	FFFFFFF0	27212223	24252627	28292A2B	2C2D2E2F	51515253		..4.....
002340	54555657	58595A5B	5C5D5E5F	17111213	14151617	18191A1B	1C1D1E1F	10010203		.....542.....
002360	04050607	08090A0B	0C0D0E0F	00000000	50E07250	40504008	58604000	42906000		.....
002380	02037254	4000588C	30E818FF	43F04018	58F8F000	05EF12FF	4770714E	95FF400F		K.....Y.....0..E0.....
0023A0	4770722C	95FF4010	4780723C	58604000	58A07254	485071A4	44507248	58E07250		.....
0023C0	07FE0000	00000000	02006000	A0000000	900022B6	00002061	90EC000C	187F52F0		.....K...../.....0
0023E0	311C41F0	F10050D0	F00418DF	18FF9540	400E4780	705E9520	400E4770	702C0203		..01.....0.....K..
002400	4000401C	4180706C	58104000	074F1000	10004100	00010A1E	12000748	12004770		.....2.....P.....
002420	70529222	401347F0	705E5930	707C4770	705E41F0	00FF5800	000458E0	000C980C		.....0.....a.....0.....
002440	001407FC	18F041F0	F00092E5	31C447F0	705E902E	00FFFFFF	90EC000C	187F52F0		.....0..00..V..D..0.....0
002460	311C41F0	F10050D0	F00418DF	9520400E	47707036	410070A2	50001028	410070AE		..01.....0.....
002480	50001038	0A2645E0	703E47F0	70929540	400E4770	70440A27	47F07092	9502400E		.....0.....0.....
0024A0	47707092	91021048	47E0707C	48504008	41505004	58604000	486070EC	07036000		.....a.....P.....
0024C0	60004256	00019220	103158F0	103405EF	47F0708E	91041048	47E0708E	92201031		.....0.....0.....
0024E0	58F01034	05EF45E0	708E18FF	58000004	58E0000C	980C0014	07FE96F0	401307FE		..0.....0.....0.....
002500	18FF9584	10324780	709443F0	103292D8	31C447F0	70949102	104B47E0	70CA4160		.....0.....Q..D..0.....
002520	60049240	60004850	40084850	70EE4740	70DE4450	70E45060	400007FE	02006001		.....-.....U.....K.....

Figure C-3. Error Analysis Dump (Part 12 of 13)

002540	60000600	00040002	05F09042	F1AE187F	41D071EE	5E2030E4	59007236	47207096	-.....0..1.. .....
002560	584030F4	0703723A	723AD501	723A3192	47807090	91404019	4710705A	95104019	..4P.....N.....U.....
002580	4780705A	0401401A	401A4780	705AD201	704C401A	95F00000	4780705A	96014019	..M.....K.....0.....
0025A0	92224013	91014019	47107080	91404019	47107080	95104019	47807080	58102000	.....0.....0.....
0025C0	45E0712A	05EF45E0	71E44120	20044140	40209A01	723A47F0	70201BFF	47F070DA	.....0.....0.....
0025E0	18409101	40194710	70089180	401E47E0	70B69182	400E4780	70B658F0	312405EF	.....Q.....0.....
002600	18114310	400D0610	89100002	58121000	45E0712A	05EF45E0	71645040	31CC47F0	.....0.....0.....
002620	70DA18FF	12FF4770	71009180	401E47E0	70F69500	400E4770	70F658F0	312405EF	.....E.....E.....
002640	984271AE	07FE47F0	712A50F0	712692F0	30E50201	31C57126	020031C7	712541E0	.....0.....0.....K..E..K..6...
002660	71AE984D	E00098F2	E02C58E0	31A007FE	00000000	12004770	71389220	400E47F0	.....2.....0.....
002680	71445900	72364770	71449240	400E9120	401947E0	715458F0	30F047F0	716218FF	.....0.....0.....
0026A0	43F04018	588030E8	58F80000	07FE1855	18B89120	401E47E0	71A55880	318C9240	..0...Y.....0.....
0026C0	80005500	40084780	71944350	400892FF	71A14480	71A04188	01004650	71844350	.....0.....0.....
0026E0	40090E50	06504250	71A1D200	80018000	94DF4016	07FE0000	000029F0	000039F0	.....K.....0.....
002700	000045F0	000019F0	50001FD2	000027C0	000027C0	000019F0	00001FF4	000027C0	.....0.....K.....C.....4...
002720	50002A18	5000254A	00000001	80001D94	00002138	00000000	00000000	00000000	.....0.....0.....
002740	00000000	700025C6	00002180	00000001	000020E0	0000217C	000019F0	00002158	.....F.....0.....
002760	00000000	000049F0	5000254A	00002788	000027C0	000027C0	00000000	00001FF4	.....0.....0.....4
002780	00000001	00020000	00002180	000019F0	00002308	300019F0	00002458	000019F0	.....0.....0.....Q.....0.....0
0027A0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	000019F0	00002548	.....0.....0.....0.....0.....0
0027C0	000027C0	6000291A	000027C0	50001EF4	41546001	40001DEE	180F4110	30984100	.....-.....A.....-.....
0027E0	003E5821	00001A23	000029F0	000039F0	000049F0	300019F0	50100294	960C0294	.....0.....0.....0.....K...K..
002800	98010290	0A115810	311C4100	00014120	000241E0	308E41F0	00F04200	005F9180	..M.....0.....0.....
002820	10004780	00E442F2	E0008900	00014620	005AD708	31C431C4	4180028E	508031A0	.....2.....P..D..K.....
002840	180058F0	309C05EF	18005870	315058F0	30EC05EF	47F0D150	40C505E3	09E840D6	.....0.....0.....0.....J..ENTRY..0
002860	C640E540	C09C240C1	E2E2E404	C5C44800	70F0F2F4	C2D3D6C3	D240D3C5	D5C7E3C8	F.V.IS.ASSUMED...D24BLOCK.LENGTH
002880	404DC3D6	D3E240F2	F060F2F3	5D40C9E2	40C9D5E5	C1D3C9C4	40D6D940	D3C5E2E2	..COLS.20-238.IS.INVALID.OR.LESS
0028A0	40E3C8C1	0540D9C5	C3D6D9C4	40D3C5D5	C7E3C848	40C2D3D6	C3D240D3	C5D5C7E3	.THAN.RECORD.LENGTH..BLOCK.LEN6T
0028C0	C840C9E2	000022DC	E4D4C5C4	700D24EE	700D24E6	00002061	000020E0	00002178	4.IS.....UMED...W...W.../.....

RPGI		ERROR ANALYSIS DUMP							
0028E0	000019F0	00002158	00000079	00002061	00002458	00002788	00000001	00002350	.....0...../.....0.....2
002900	000020E2	00001FF4	000027C0	600029FE	58703150	58F030DC	05EF9200	3014D609	.....4.....-.....E.....0.....0.....
002920	30853085	4770D242	D200D299	30845870	315058F0	308805EF	D200D29A	3084D200	.....K..K..K.....0.....K..M...M..
002940	3084D299	95F03015	4780D1CC	95F03078					..M..0.....J..0..#
END OF DUMP									

Figure C-3. Error Analysis Dump (Part 13 of 13)

UNDEFRT

H — NO ERROR ANALYSIS DUMP BLANK IN COL. 8

```

001      010 FCARDIN IPEAF 80 80          CTLRDR
002      020 FPRINTER 0 F 120 120      OF  PRINTER
003      010 ICARDIN  011 01  2 CI
004      020 I              1 80 DATA
005      010 OPRINTER D 1      01
006      020 0              DATA      90
007      030 0              D 07  OF
    
```

— FILE WITH UNDEFINED RECORD TYPE

SYMBOL TABLES

RESULTING INDICATORS

ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI	ADDRESS RI
000011 OF	000014 1P	000015 LR	000016 00	000017 01	00007A LO	000085 H0
000086 H1	000087 H2	000088 H3	000089 H4	00008A H5	00008B H6	00008C H7
00008D H8	00008E H9					

FIELD NAMES

ADDRESS FIELD	ADDRESS FIELD	ADDRESS FIELD	ADDRESS FIELD	ADDRESS FIELD
0001C4 *ERROR	0001E4 JATA			

DISPLACEMENT FOR \*ERROR

DISPLACEMENT FOR DATA

DISPLACEMENT FOR '01' INDICATOR

DISPLACEMENT FOR 'H0' INDICATOR

PROGRAM POINTERS

TABLE INPUT/OUTPUT	000234
INPUT/OUTPUT INTERCEPT	000858
INPUT FIELD EXTRACTION	000238
DETERMINE RECORD TYPE	0002C8
GET INPUT RECORD	0003FC
GET LOOKAHEAD FIELDS	00050C
DETAIL CALCULATIONS	000510
TOTAL CALCULATIONS	000524
TOTAL OUTPUT	00053C
OVERFLOW OUTPUT	000556

Figure C-4. Unformatted Dump (Part 1 of 31)

```

OVERFLOW BYPASS
HEADER/DETAIL OUTPUT
OUTPUT FIELDS
PTR TO ADDRESSES OF DTFS
PTR TO ADDRESSES OF I/O INTERFACE RTNS
MAIN INPUT/OUTPUT
INPUT/OUTPUT INTERFACE RTNS
INPUT/OUTPUT REQUEST BLKS
INITIALIZATION/WORK AREA
MASTER DRIVER

```

```

00059C
0005A8
000604
000788
000D98
000B58
000790
0007A8
000DD0
000F20

```

DISPLACEMENT FOR I/O REQUEST BLOCKS

UNIVAC OS/3 DISC RPG VER750728 UNDEFRT 08/05/75 19.16 PAGE 2

PROGRAM LENGTH 001078  
NO ERROR NOTES IN THIS COMPILATION  
COMPILATION COMPLETED ON 08/05/75 AT 19.17

UNIVAC SYSTEM OS/3 LINKAGE EDITOR  
DATE- 75/08/05 TIME- 19.17

VER031874

CONTROL STREAM ENCOUNTERED AND PROCESSED AS FOLLOWS-

```

/*
LOADM UNDEFRT
INCLUDE UNDEFRT
*/
DP$COME *AUTO-INCLUDED*

```

\*DEFINITIONS DICTIONARY\*

SYMBOL.	TYPE.	PHASE.	ADDRESS.	SYMBOL.	TYPE.	PHASE.	ADDRESS.	SYMBOL.	TYPE.	PHASE.	ADDRESS.
DP\$COM0	ENTRY	ROOT	00000000	DP\$COM1	ENTRY	ROOT	00300030	DP\$COM2	ENTRY	ROOT	00000000
DP\$COM3	ENTRY	ROOT	00300030	DP\$COM4	ENTRY	ROOT	00300000	DP\$COM5	ENTRY	ROOT	00000000
DP\$COM6	ENTRY	ROOT	00000000	DP\$COM7	ENTRY	ROOT	00300000	KE\$ALP	ENTRY	ABS	00001528
MESRES	ENTRY	ABS	00001528	PR\$10E	CSECT	ROOT	00300000	PRINTER	ENTRY	ROOT	000008A0
UNDEFRT	CSECT	ROOT	00303480								

Figure C-4. Unformatted Dump (Part 2 of 31)

\*\*\*PHASE STRUCTURE\*\*\* HEX BYTES REPRESENTED BY EACH DASH - 40

I 00\*1528\*

\*\*\* ALLOCATION MAP \*\*\*

LOAD MODULE - UNDEFR SIZE - 00001528

PHASE NAME	TRANS ADDR	FLAG	LABEL	TYPE	ESID	LNW ORG	HIADDR	LENGTH	OBJ ORG
UNDEFROO	MODE - ROOT					00000000	00001527	00001528	
*** START OF AUTO-INCLUDED ELEMENTS -									
- 03/26/74 01.09 -									
			PR\$IOE	OBJ					
			PR\$IOE	CSECT	02	00000000	000004A9	000004AA	00000000
			DP\$COM7	ENTRY	02	00000000			00000000
			DP\$COM0	ENTRY	02	00000000			00000000
			DP\$COM1	ENTRY	02	00000000			00000000
			DP\$COM6	ENTRY	02	00000000			00000000
			DP\$COM2	ENTRY	02	00000000			00000000
			DP\$COM5	ENTRY	02	00000000			00000000
			DP\$COM4	ENTRY	02	00000000			00000000
			DP\$COM3	ENTRY	02	00000000			00000000
*** END OF AUTO-INCLUDED ELEMENTS -									
- 75/08/05 19.16 -									
			UNDEFR	OBJ					
			UNDEFR	CSECT	01	00000480		00000000	00000000
			PRINTER	ENTRY	01	000008A0			000006F0
00000480									

LOAD ADDRESS

FLAG CODES -

B - BLK DATA CSECT    D - AUTO-DELETED    E - EXCLUSIVE "A" REF    G - GENERATED EXTRN    I - INCLUSIVE "V" RE

M - MULTIPLY DEFINED    N - NOT INCLUDED    P - PROMOTED COMMON    U - UNDEFINED REF    V - VCON ITEM

\*ANY OTHER CODES REPRESENT PROCESS ERRORS\*

LINK EDIT OF 'UNDEFR' COMPLETED  
 DATE- 75/08/05 TIME- 19.17  
 ERRORS ENCOUNTERED- 0000

Figure C-4. Unformatted Dump (Part 3 of 31)

```

      I THIS IS A VALID DATA RECORD
USER CANCEL DUMP      JOB NAME #UNDEFRT      SYSTEM VERSION 2.01 .00
PSW AT INTERRUPT #C016001C4000150C  ERROR CODE # 00000000  TCB ADDR # 0054E0
PROBLEM PROGRAM REGS
  REGS 0-7  00000000  00003430  00003BF8  00003480  00001480  00002480  00003480  00000480
  REGS 8-F  50000A92  00001280  00001280  00003430  00000AB4  00001280  600014E6  00000000
                                GENERAL REGISTER 3 (LOAD ADDRESS OF
                                RPG II OBJECT PROGRAMS)
JOB PREAMBLE
  FFFA00  E4D5C4C5  C6D9E340  000000B6  00006230  00005C00  00001528  00005A00  00000000  005400
  FFFA20  E4D5C4C5  C6D9F0F0  E4D5C4C5  C6D9F0F0  75D80500  00006294  00006648  00000000  005420
  FFFA40  00000000  00000000  00000000  00003000  00000080  00000000  00000000  00000000  005440
  FFFA60  00000000  0750905C  004800D9  40F7F5F2  F1F70000  00400000  05F20310  00000000  005460
  FFFA80  00000045  00042100  00040100  0001003C  0A180000  00000001  00690E00  0A180034  005480
  FFFAA0  00000001  00651201  00000000  00003030  00000000  00000000  00000000  00000000  0054A0
  FFFAC0  000001A8  00000000  00001000  04003030  00000000  00000000  00000100  00000000  0054C0
TCBS
  FFFAE0  100054E0  00003300  180054E0  00003030  00000000  00005400  1C000000  00000000  0054E0
  FFFB00  C016001C  4000150C  00003000  00003430  00000BF8  00000480  00001480  00002480  005500
  FFFB20  00003480  00003480  50000A92  00001280  00301280  00000480  00000AB4  00001280  005520
  FFFB40  600014E6  00003000  00003000  00003030  00003000  00000000  00000000  00000000  005540
  FFFB60  00000000  00003000  00000000  00003030  00000000  00000000  00042000  0C810000  005560
  FFFB80  00FF2000  00003000  00003000  30303030  00300000  00300000  00000000  00000000  005580
  FFFBA0  00000000  00000000
OPEN FILE TABLE
  FFFBA8  0000000A  00000000  00000000  00003000  00000000  00000000  00000000  00000000  0055A8
  FFFBC8  00000000  00003000  00000000  00003030  00000000  00000000  00000000  00000000  0055C8
      0055E8 TO 005888 SAME AS ABOVE
  FFFE88  00000000  00003000  0A180000  00003030  00000000  00000000  00690110  00000000  005888

```

Figure C-4. Unformatted Dump (Part 4 of 31)

```

FFFEA8 00000000 00000000 00000000 00000000 00000000 00000000 0100010C 2754003F 0058A8
FFFEC8 02000115 4793000B 03000116 679E001F EE000000 00000000 00000000 00000000 0058C8
FFFEE8 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0058E8
    
```

005908 TO 0059FF SAME AS ABOVE

PROGRAM REGION

```

000000 47F0F008 4004F0F7 91401048 4780F018 50001058 58001054 90EC000C 50E00010 005A00
000020 41A0F3BE 41B0F464 92001032 91401046 4710F03E 9213103E 96021032 07FB9102 005A20
000040 1046071B 95201031 4780F1CA 954B1031 4780F0F8 954C1031 4780F0F4 9180104A 005A40
000060 4710F0EE 92141038 96021032 07FB958B 10314780 F082958C 10314780 F08247F0 005A60
000080 F0649104 104A078E 47F0F08C 948F104A 958B1031 4770F09C 9610104A 12004780 005A80
0000A0 F0AA5000 102847F0 F0AE9640 104A9101 10494780 F08C9118 104A074B 91181046 005AA0
0000C0 4780F0C6 05EA9140 104A4710 F085820 10285020 001007FB 4320103C 4220104E 005AC0
0000E0 928F103C 96021047 0A009118 1046077B 05EA07FB 96101048 94F3104A 948F1047 005AE0
000100 1B229118 10464780 F10C05EA 95000016 4780F122 43200016 4570F168 0A009640 005B00
000120 10479540 00174770 F12E47F0 F38E910F 00164780 F13C948F 104705EA 9180103C 005B20
000140 4780F148 94F3104A 954C1031 4770F154 96101048 96201047 96201048 43200017 005B40
000160 4570F168 47F0F3A4 41300008 91101048 4780F190 19234740 F1904140 F1829107 005B60
000180 10494780 F1881A23 43224000 47F0F198 89200003 41202001 4220103C 91101048 005B80
0001A0 4780F1A8 9680103C 91201048 4710F134 960E103C 94CF1048 07F71109 19210929 005BA0
0001C0 39394149 51590969 39391850 94F3104A 58401050 4890102C 91181046 4780F1E2 005BC0
0001E0 05EA4830 104C9104 10484710 F24A9101 10484710 F20A9110 10464710 F2261864 005BE0
000200 18694330 600147F0 F22A9548 10444770 F21A5830 105847F0 F22A4330 10445833 005C00
000220 000047F0 F22A4330 50011839 4780F232 18334930 104C4700 F2464830 104C9602 005C20
    
```

Figure C-4. Unformatted Dump (Part 5 of 31)

000240	10489680	10324030	10429107	10494770	F2669101	10434780	F2664300	10254203	005C40
000260	40009A01	10429150	10474780	F27E948F	10474380	104F4280	103C47F0	F28A9120	005C60
000280	10474780	F28A3E10	10479102	10474780	F29E4300	104E4200	103C94F0	10479110	005C80
0002A0	10464780	F2841A59	12334780	F2840630	4430F388	91081046	4780F2CA	5800103C	005CA0
0002C0	0202103D	10515000	10509104	10484780	F3461822	91101046	4710F2E6	06404320	005CC0
0002E0	400047F0	F2EC0650	43205000	4170F346	4140F33E	4150F342	12220785	4220103C	005CE0
000300	4130000F	14239510	103C0784	910F103C	4780F334	94F0103C	9500103C	07849550	005D00
000320	103C0785	96101048	9510103C	07849520	103C0785	92431038	96401032	07F89620	005D20
000340	104847F0	F1689108	10494780	F3725850	10501859	95481045	4770F364	50501058	005D40
000360	47F0F372	18444340	10454144	00005050	400091FF	10434770	F38C9106	103C4710	005D60
000380	F38C9601	10329218	103807FB	0A009118	10464770	F39805EA	9180103C	4780F3A4	005D80
0003A0	94F3104A	9180104A	4780F3AE	07FB9104	104A078B	47F0F0AE	D2004000	50009180	005DA0
0003C0	10024710	F3C80A01	9102104A	4780F3E8	94FD104A	9604104A	9101104A	4780F3E8	005DC0
0003E0	94FE104A	9608104A	917F1002	078E9108	10024780	F43A91E0	104A4780	F43A94F7	005DE0
000400	10029118	10464780	F4129604	104A47F0	F4169602	104A9101	10494780	F43A9108	005E00
000420	10164780	F43A9118	10464780	F4369638	104A47F0	F43A9601	104A9120	10024780	005E20
000440	F4549104	10464710	F4549242	10389620	103207FB	91401002	078E9223	10389610	005E40
000460	103207FE	917F1032	4780F492	91801048	4780F48C	91401048	4780F486	98EC000C	005E60
000480	58001058	0A3D98EC	000C9A3D	41E0F492	0A3D9140	10484780	F4A498EC	000C5800	005E80
0004A0	105807FF	98EC000C	07FF0000	00003000	05F058F0	F00607FF	00001280	00000000	005EA0
0004C0	00000000	0000F000	00000000	00003000	00000000	00000000	00000000	00000000	005EC0
0004E0	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	005EE0
005F00 TO 005F20 SAME AS ABOVE					START OF RPG II OBJECT PROGRAM				
000520	00000000	00000000	0000F000	00003000	00000000	00F00000	00000000	000000F0	005F20

Figure C-4. Unformatted Dump (Part 6 of 31)



```

000540 00000000 00000000 00000480 000036E4 00001008 00000480 000006E8 00000480 005F40
000560 00000778 00000480 000008AC 0000093C 00000480 000009C0 000009D4 000009EC 005F60
000580 00000480 00000A0E 00000A4C 00000A58 00000AB4 00000C38 00001248 00001008 005F80
0005A0 00000C40 00000BF8 00000480 00000430 00000480 00000480 00000480 00000480 005FA0
0005C0 00000480 00000480 00000480 00001280 00000480 00000480 00000480 00000480 005FC0
0005E0 00000480 00000480 00000480 00000430 00000480 00000480 00000480 00000480 005FE0
006000 TO 006040 SAME AS ABOVE
000640 00000002 00000A00 00000000 00000000 00001506 00000000 1C404040 00001000 006040
000660 00002000 00000000 00000000 00000000 00000000 C1000000 00000000 00000BF8 006060
000680 00000000 00000000 00000000 00000000 00000000 E3C8C9E2 40C9E240 C10540C9 006080
0006A0 05E5C1D3 C9C440C4 C1E3C140 05C5C33E 09C44040 40404040 40404040 40404040 0060A0
0006C0 40404040 40404040 40404040 40404040 40404040 40404040 40404040 40404040 0060C0
0006E0 40404040 07FE0000 05805810 805E1211 078E5881 00041288 078E1A83 589030F4 0060E0
000700 4A910002 58290000 50203194 18985A90 31AC05F8 90ABF038 48C0F04C 0201F010 006100
000720 C0024110 00001838 438C0001 9502C000 4780F0E6 9580C008 4770F050 9500C00A 006120
000740 4770F11E 47F0F050 80000854 00000000 00000000 07000700 00000000 0000000C 006140
000760 00081C00 024F31E4 200007FE 80046DE0 92003017 07FE0000 50E0000C 58290000 006160
000780 50203194 0705F020 F020D703 10101010 9047D028 070005E1 00000000 00009847 006180
0007A0 0028188E 18909502 80044770 80320200 900F8004 47F08032 985631AC 1A541A64 0061A0
0007C0 987831AC 1A761A86 07F4960F 10005800 311C58ED 000C07FE 05A00500 F0021016 0061C0
0007E0 4770A018 45C0A05A 02001015 F00247F0 E0069108 F0034710 A02A0500 1015F002 0061E0
000800 4780E006 58810010 41933001 18884381 00161A88 91108000 4780A04A 45C0A05A 006200
000820 47F0A000 92F03085 92C331C4 501031CC 47F0E006 43310016 41380001 42810016 006220
000840 05001016 1017070C 92011016 07FC0000 45110020 8F1C0000 00000284 00000535 006240

```

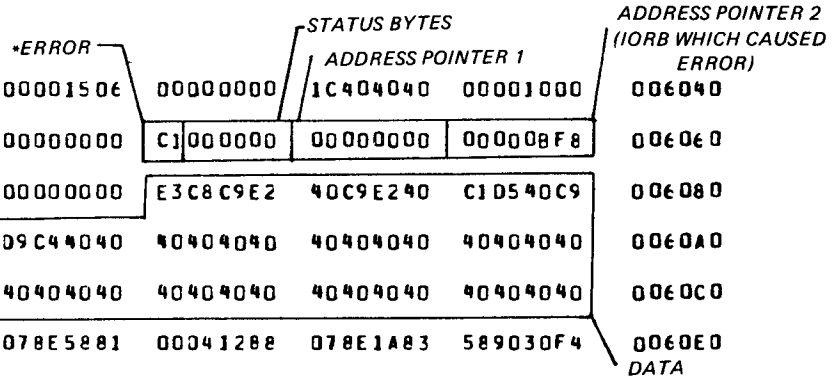


Figure C-4. Unformatted Dump (Part 7 of 31)

000860	00000000	0000029C	0E010101	00000000	454E0020	95C92001	4770402C	41A03017	006260
000880	96F0A000	50A1000E	47F0401E	300E0000	02B40203	1004401A	45FE0040	80040160	006280
0008A0	41A30085	50A10008	47F0E00E	058047F0	8010800C	00000000	000002C0	00601810	0062A0
0008C0	5800311C	5010000E	50ED0004	070E307B	307B47F0	809E58C0	30F44AC1	0002582C	0062C0
0008E0	00005020	31945890	809A1A93	58A10008	96F0A000	41F03085	19AF4770	805492C1	0062E0
000900	31C458A0	30A84AA0	800E501A	000058F1	000C12FF	4780807E	1AF358C0	30B458AC	006300
000920	016C1AA3	05EF58ED	00045800	000807FE	0700801A	000003A0	00000000	F0F0F0F0	006320
000940	F0F0F0F0	F0F0F0F0	00000000	5810808E	1A135800	30F44A01	000E1890	58F0800A	006340
000960	1AF305EF	9200900E	58F030A0	05EF95F0	30854770	80DA58A0	30A84AA0	800E18EE	006360
000980	50EA0000	47F0807E	95229013	477080FE	41110004	92F01001	96F03015	020E307B	006380
0009A0	808E47F0	80785080	001458F0	30B041FF	000005EF	58800014	47F08028	07FE0000	0063A0
0009C0	05805890	311C90DE	914898AD	312098DE	914807FE	05805890	311C90DE	914898AD	0063C0
0009E0	312098DE	91484190	00FF07FE	58A0311C	90DEA000	582030F4	580030A0	58C030E0	0063E0
000A00	98DEA000	07FE58A0	311C90DE	A0005820	30F45800	30A058C0	30E05880	300E0E07	006400
000A20	300C300C	07E80580	95F03011	47708024	9207202F	589C0004	1A9305E9	9202202E	006420
000A40	41020020	05E00200	30112033	05809200	203398DE	A00007FE	58A0311C	90DEA000	006440
000A60	582030F4	580030A0	58C030E0	92002033	058095F0	30174770	801E9201	2032589C	006460
000A80	00001A93	05E99202	202E4102	002005ED	058047F0	800E96F0	800198DE	A00007FE	006480
000AA0	02003011	203398DE	A00007FE	98DEA000	07FE0000	00000E0C	00000E18	58820020	0064A0
000AC0	024F800A	31E407FE	07FE0000	00000000	E3C8C9E2	40C9E240	C1D540C9	05E5C1D3	0064C0
000AE0	C9C440C4	C1E3C140	05C5C3D6	09C44040	40404040	40404040	40404040	40404040	0064E0
000B00	40404040	40404040	40404040	40404040	40404040	40404040	40404040	40404040	006500
000B20	00404040	40404040	40404040	40404040	40404040	40404040	40404040	40404040	006520
000B40	40404040	40404040	40404040	40404040	40404040	40404040	40404040	40404040	006540

RECORD IN ERROR  
(UNDEFINED RECORD)

Figure C-4. Unformatted Dump (Part 8 of 31)

006560 TO 006580 SAME AS ABOVE					RECORD ADDRESS	IORB IN ERROR			
000880	40404040	40404040	40404040	40404040	40404040	40404040	40400000	00000000	006580
0008A0	0400806E	00000000	00000000	0000080C	000008C6	00000000	00000400	07D9C9D5	0065A0
0008C0	E3C5D940	004008E4	00000F8A	000134F0	F6200000	00000000	00000FC0	09000B22	0065C0
0008E0	00000078	002C2400	84082004	00783009	00000B22	00000000	00000AD0	00000000	0065E0
000C00	00500000	000140FF	FFFFFFFF00	00003011	08000000	00000AD0	00000B23	00000000	006600
000C20	00780000	000240FF	FFFFFFFF00	00000011	10200000	00000000	00000480	000008A0	006620
000D40	713695FF	40124770	71101899	47F07114	43904012	9511400C	47707122	41F072D8	006740
000D60	05EF41A9	71DC439A	00004850	71A44150	500145E0	71F0D203	400F71A8	02014008	006760
000D80	71A44100	00015A00	40005000	40005800	716058E0	000C980C	001407FE	00000000	006780
000DA0	000011F8	00000000	40000C66	00003F18	00000001	000008A0	00000C3C	00000480	0067A0
000DC0	00000C18	00000000	00003480	00003C40	00001248	00001280	00001280	00000000	0067C0
000DE0	00000A94	00780001	FFFFFFFFFF	27212223	24252627	28292A2B	2C2D2E2F	51515253	0067E0
000E00	54555657	58595A5B	5C5D5E5F	17111213	14151617	18191A1B	1C1D1E1F	10010203	006800
000E20	04050607	08090A0B	0C0D0E0F	00003030	50E07250	40504008	58604000	42906000	006820
000E40	02037254	40005880	30E818FF	43F0401E	58F8F000	05EF12FF	4770714E	95FF400F	006840
000E60	4770722C	95FF4010	4780723C	58E04000	58A07254	485071A4	44507248	58E07250	006860
000E80	07FE0000	00000000	02006000	A0000000	90000D76	00000B21	90EC000C	187F58F0	006880
000EA0	311C41F0	F10050D0	F00418DF	18FF9540	400E4780	705E9520	400E4770	702C0203	0068A0
000EC0	4000401C	4180706C	58104000	074F1000	10004100	00010A1E	12000748	12004770	0068C0
000EE0	70529222	401347F0	705E5900	707C4770	705E41F0	00FF58D0	000458E0	000C980C	0068E0
000F00	001407FE	18F041F0	F00092E5	31C447F0	705E902E	00FFFFFF	90EC000C	187F58F0	006900
000F20	311C41F0	F10050D0	F00418DF	9520400E	47707036	410070A2	50001028	410070A8	006920
000F40	50001038	0A2645E0	70BE47F0	70929540	400E4770	70440A27	47F07092	9502400E	006940

Figure C-4. Unformatted Dump (Part 9 of 31)

000F60	47707092	91021048	47E0707C	48504008	41505004	58604000	486070EC	07036000	006960
000F80	60004256	00019220	103158F0	103405EF	47F0708E	91041048	47E0708E	92201031	006980
000FA0	58F01034	05EF45E0	70BE18FF	58000004	58E0000C	980C0014	07FE96F0	401307FE	0069A0
000FC0	18FF9584	10324780	709443F0	10323208	31C447F0	70949102	104B47E0	70CA4160	0069C0
000FE0	60049240	60004850	40084850	70EE4740	70DE4450	70E45060	400007FE	02006001	0069E0
001000	60000600	00040002	05F09042	F1AE187F	410071EE	582030E4	59007236	47207096	006A00
001020	584030F4	0703723A	723AD501	723A3132	47307090	91404019	4710705A	95104019	006A20
001040	4780705A	0401401A	401A4780	705A0201	704C401A	95F00000	4780705A	96014019	006A40
001060	92224013	91014019	47107080	91404019	47107080	95104019	47807080	58102000	006A60
001080	45E0712A	05EF45E0	71644120	20044140	40209A01	723A47F0	702018FF	47F070DA	006A80
0010A0	18409101	40194710	70089180	401647E0	70369182	400E4780	70B658F0	312405EF	006AA0
0010C0	18114310	40000610	89100002	58121000	45E0712A	05EF45E0	71645040	31CC47F0	006AC0
0010E0	70DA18FF	12FF4770	710C9180	401647E0	70F69500	400E4770	70F658F0	312405EF	006AE0
001100	984271AE	07FE47F0	712A50F0	712692F0	30850201	31C57126	020031C7	712941E0	006B00
001120	71AE9840	E00098F2	E02C58E0	31A037FE	00000000	12004770	71389220	400E47F0	006B20
001140	71445900	72364770	71449240	400E9120	401947E0	715458F0	30F047F0	716218FF	006B40
001160	43F04016	588030E8	58FF8000	07FE1855	183B9120	401647E0	71AA5880	31BC9240	006B60
001180	80009500	40084780	71944350	400892FF	71A14480	71A04188	01004650	71844350	006B80
0011A0	40090650	06504250	71A10200	80019000	94DF4016	07FE0000	00001480	00002480	006BA0
0011C0	00003480	00003480	50000A92	00001280	00001280	00000480	00000A84	00001280	006BC0
0011E0	50001408	5000100A	00000001	80000854	00000BF8	00000000	00000000	00000000	006BE0
001200	00000000	70001086	00000C40	00000001	00000BA0	00000C3C	00000480	00000C18	006C00
001220	00000000	00003480	5000100A	00001248	00001280	00001280	00000000	00000A84	006C20
001240	00000001	00020000	00000C40	00003430	00000E98	00000480	00000F18	00000480	006C40

Figure C-4. Unformatted Dump (Part 10 of 31)

001260	00000480	00000480	00000480	00000480	00000480	00000480	00000480	00001008	006C60
001280	00001280	600013DA	00001280	50000934	41546001	400008AE	180F4110	30984100	006C80
0012A0	003E5821	00001A23	00001480	00002480	00003480	00000480	50100294	960C0294	006CA0
0012C0	98010290	0A115810	311C4100	00014120	000841E0	308E41F0	00F04200	005F9180	006CC0
0012E0	10004780	006A42F2	E0008900	00014620	005A0708	31C431C4	41800286	50B031A0	006CE0
001300	180058F0	309C05EF	18005870	315058F0	30EC05EF	47F0D150	40C5D5E3	09E840D6	006D00
001320	C640E540	C9E240C1	E2E2E404	C5C44800	70F0F2F4	C2D3D6C3	D240D3C5	D5C7E3C8	006D20
001340	404DC3D6	D3E240F2	F060F2F3	5040C9E2	40C9D5E5	C1D3C9C4	40D6D940	D3C5E2E2	006D40
001360	40E3C8C1	D540D9C5	C3D6D9C4	40D3C5D5	C7E3C848	40C2D3D6	C3D240D3	C5D5C7E3	006D60
001380	C840C9E2	0000D9C	E4D4C5C4	7000DFA6	7000DFA6	00000821	000008A0	0000C3E	006D80
0013A0	00000480	0000C18	00000079	00000B21	0000D18	00001248	00000001	0000E1D	006DA0
0013C0	00000822	00000AB4	00001280	60001436	58703150	58F030DC	05EF9200	3014D609	006DC0
0013E0	30853085	4770D242	D200D299	30845870	315058F0	308805EF	D200D29A	3084D200	006DE0
001400	3084D299	95F03015	4780D1CC	95F03078	4780D1A0	91D1D298	4780D1B4	47F0D1CC	006E00
001420	91F0D298	4710D1CC	91D1D298	4710D13C	92F0D298	92D1D298	47F0D1F4	D2D7D29C	006E20
001440	307C92FF	D298D708	307B307B	58703150	58F030C8	05EF1B99	58703150	58F030CC	006E40
001460	05EF910F	D2984780	D1F492F0	D298D208	307BD298	95F03015	4780D242	91D2D298	006E60
001480	4780D20E	58703150	58F030D4	05EF9602	D2985870	315058F0	30A805EF	D2D03084	006E80
0014A0	D29A5870	315058F0	308C05EF	58703150	58F030C4	05EF4990	D28C4770	D15047F0	006EA0
0014C0	D1D695F0	30144710	D22C4100	00D15870	315058F0	30EC05EF	41000001	58703150	006EC0
0014E0	58F0309C	05EF1BFF	D6083085	30854770	D2860A1A	18131800	06000A3A	12004780	006EE0
001500	D24247F0	D1681800	18130A1C	00FF3700	00D10053	0C0D1280	03000000	00000000	006F00
001520	00000000	000000D0	000G0000	00000000	00000000	00000000	00000000	00000000	006F20
001540	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	006F40

Figure C-4. Unformatted Dump (Part 11 of 31)

```

006F60 TO 007800 SAME AS ABOVE
001E00 0000312E 0000312E 0000312E 00003664 00001280 0000312E 00003082 000000AB 007800
001E20 00000000 00000034 001FC422 00001F29 00001F25 00000000 00000000 00000664 007820
001E40 00000664 00000000 00000000 18100830 00000018 0F000000 F0000000 000F0000 007840
001E60 0F010700 07000700 9510804E 4780C018 9104D270 4780C014 45E0D4EC 47F0D2F6 007860
001E80 94FE0601 47F0C008 9680D601 05C0D207 9018D660 9180D601 4780C02A 9104D270 007880
001EA0 4780C01A 45E0D4EC D20105AC 804AD200 0580804C 47F0C038 1878D201 05AC7003 0078A0
001EC0 0200D5B0 700F9120 D2704780 C12E0701 05F0D5F0 9108D5B0 4780C052 4530C132 0078C0
001EE0 9102D5B0 4710C148 9140D5B0 4710C154 4810D5AC 45E0D28C 91802002 4780C076 0078E0
001F00 4530C132 91202002 4780C096 4530C132 91102002 4780C000 48102008 45E0D28C 007900
001F20 47F0C076 D203D5AC 20069180 20024780 C0A89200 05AC45E0 04040207 9044D650 007920
001F40 9180D601 4710C0FE D200D5AF 700245E0 04040201 903AD656 D2049030 C1809101 007940
001F60 02704780 C1E4D203 05AC7005 45E0D4D4 02079068 0650D205 05AC7009 4100D5AC 007960
001F80 41109024 45E0D272 47F0C122 18334140 80034130 30019540 40004780 C1184140 007980
001FA0 400147F0 C1D44430 C17AD202 9031C185 45E0D292 947FD601 94AFD600 47F0D2F6 0079A0
001FC0 9110D600 4710C146 9610D600 4110D0C4 45E0D4AC 07F34110 00D645E0 04AC47F0 0079C0
001FE0 C05A4110 00D545E0 04AC9640 060047F0 C0829150 06004780 C0D89240 902CD21E 0079E0
002000 902D902C 47F0C126 D2009024 8002C5D5 E3D9E8C5 08E4D701 05F0D5F0 9200D659 007A00
002020 02079018 06609140 80134710 C0FC0201 05808003 4810D580 45E0D28C 91108013 007A20
002040 4780C032 4530C14C 91088013 4780C03E 4530C0DE 948FD659 91202002 4780C05E 007A40
002060 91102002 4780C140 48102008 45E0D28C 47F0C03E 91802002 4710C176 91042002 007A60
002080 4710C166 1774370 800245E0 04FC0200 7000D659 02007001 2000D200 70022003 007A80
0020A0 0203D5AC 20069180 20024780 C09C9200 05AC5840 05ACD203 05AC8005 5B40D5AC 007AA0
0020C0 5040D5AC 02027003 05AD9101 D2704790 C1829120 D2704780 C0CED204 902EC18E 007AC0

```

Figure C-4. Unformatted Dump (Part 12 of 31)

0020E0	45E0D4E4	9101D602	4780C10C	948F3E00	47F0D2F6	9110D600	4710C0FA	9210D5AC	007AE0
002100	9610D600	9640D659	411000C4	45E0D4AC	07F34110	00E845E0	04AC9620	060047F0	007B00
002120	C0B49104	06014710	C0D69180	20024710	C0D69170	06304770	C0D6D203	05DC2006	007B20
002140	43302000	06304230	05E09203	05E39601	060247F0	C0D64530	C14C9640	065947F0	007B40
002160	C06E9140	06004710	C1644110	00D545E0	04AC9640	06599670	060007F3	411000E4	007B60
002180	45E0D4AC	9640D659	47F0C06E	411000C2	45E0D4AC	47F0C06E	9150D600	4770C0CE	007B80
0021A0	47F0C0BC	C3E2C5C3	E3009160	804C4710	C0769200	D5ACD202	D5AD804F	5830D5AC	007BA0
0021C0	9140804C	4710C086	0201D5B0	804A4810	D5B045E0	D28C9104	20024710	C0869120	007BC0
0021E0	20024780	C0484810	200845E0	D28C47F0	C034D203	D5302006	91802002	4780C05A	007BE0
002200	9200D5B0	5A30D5B0	5030D5AC	D203D53C	D5AC4330	2000D630	4230D5E0	9203D5E3	007C00
002220	9104D270	4780C082	45E0D4EC	47F0D2F6	5A30D5E4	5030D5AC	47F0C062	C5D5E3C5	007C20
002240	D9809602	0600D5C0	9200D659	D701D5F0	D5F0D207	9018D6E0	D201D5AC	80034810	007C40
002260	D5AC45E0	D28C9120	20024710	C0D89104	20024710	C0F04830	20041233	4770C068	007C60
002280	06002000	20004780	C2061700	91802002	4780C054	43002003	47F0C058	43002000	007C80
0022A0	45E0D27A	47F0C0C4	47F0C1CC	47F0C0C4	411000D4	45E0D4AC	D201D5F4	80034830	007CA0
0022C0	20041233	4780C088	17009180	20024780	C0924300	200347F0	C0964300	200045E0	007CC0
0022E0	D27A47F0	C0B847F0	C1EC47F0	C0B84810	2004D201	D5F42004	45E0D28C	47F0C076	007CE0
002300	4810D5F4	45E0D2BC	47F0C038	9102D600	4780C10C	4110D0C9	45E0D4AC	47F0C10C	007D00
002320	9608D602	411000C4	45E0D4AC	9601D630	9640D659	47F0C10C	411000E4	45E0D4AC	007D20
002340	9640D659	9601D600	9608D602	91102002	4710C218	17774370	800245E0	D4FC0200	007D40
002360	7000D659	D2007001	2000D200	70022033	9108D602	4710C20E	9200D5AC	D202D5AD	007D60
002380	20075830	D5ACD203	D5B08005	5830D530	5030D5AC	D2027003	D5AD94F7	06029140	007D80
0023A0	D2704780	C1C4D205	D5AC80D9	4100D5AC	41109024	45E0D272	D2049030	C224D707	007DA0

Figure C-4. Unformatted Dump (Part 13 of 31)

0023C0	D5ACD5AC	0200J5AF	800245E0	04D4J20J	903AD656	9101D600	4710C180	0203D5AC	007DC0
0023E0	20069180	20024780	C1A69200	D5AC45E0	D4D4D207	9044D650	D203D5AC	800545E0	007DE0
002400	04D4D207	9068D650	45E0D292	94FC3600	47F0D2F6	9102D600	4780C1FA	411000E5	007E00
002420	45E0D4AC	9120J26F	4780C10C	9601J659	47F0C10C	0500D5F3	200047D0	C08E47F0	007E20
002440	C0A64110	00C545E0	D4AC47F0	C10C9280	D65947F0	C0C4D702	70037003	47F0C152	007E40
002460	411000C7	45E0D4AC	47F0C10C	C5E7E3D9	D538D207	9018D660	D701D5F0	D5F0D201	007E60
002480	D5AC8003	481D05AC	45E0D28C	9180D202	4710C02A	4100000C	47F0C026	17774370	007E80
0024A0	800245E0	D4FC9200	D659D200	7000J659	D2007001	2003D200	7002D5AC	D203D5AC	007EA0
0024C0	20069200	D5AC5830	D5ACD203	D5AC8005	5830D5AC	5030D5AC	D2027003	D5AD9120	007EC0
0024E0	D2704780	C08D2D202	902EC086	9604J6J2	45E0D4E4	47F0D2F6	C3D6D460	9120D600	007EE0
002500	4710C07A	9104D602	4780C01A	D500D203	D5F347F0	C0289180	20024780	C02C0D500	007F00
002520	2000D5F3	4770C08E	D203D5AC	20069180	20024780	C03E9200	D5AC5830	D5AC45E0	007F20
002540	04D4D207	9044D650	D203D5AC	80095A30	D5AC45E0	D4D4D207	905CD650	D6038009	007F40
002560	80094780	C07A0630	5030D5AC	45E0D4D4	D2079050	D650D707	D5ACD5AC	D200D5AF	007F60
002580	800245E0	D4D4D201	903A0656	D205J5AC	80004100	D5AC4110	902445E0	D272D203	007F80
0025A0	D5AC8005	45E0D4D4	D2079068	D650N5E0	D29294F8	D60247F0	D2764110	00D745E0	007FA0
0025C0	D4AC47F0	C02C0700	5C40D63C	5A50J264	187547F0	D2F64850	D5F245E0	D5449200	007FC0
0025E0	D5ACD202	D5AD7004	947FD5AD	D203D5E4	D5ACD203	D5DCD5AC	D200D5E0	D5F39203	007FE0
002600	D5E3D602	70017001	4780C0C8	D501J5F2	D0D64780	C10C4130	00454030	D5F645E0	008000
002620	D52C9243	60009290	60014330	D5F3D630	42360002	D7016003	6003D203	6005D5AC	008020
002640	92006009	D202E00A	7001D207	600D35C0	18110A07	5000D110	F1330110	D110D202	008040
002660	6015D110	5010D110	F133D110	D110J202	6018D110	D2036018	D5E8D207	601FD5C8	008060
002680	D21D6027	D24D41E0	60455060	D61C4830	06144130	30454030	D61447F0	D2F69102	008080
0026A0	D2704780	C0369258	D388D201	D38CC13E	9261D38E	D207D38F	D5C09208	D3C70700	0080A0

Figure C-4. Unformatted Dump (Part 14 of 31)



0026C0	47F0C0F4	3C000E33	00100A46	12004730	C0FE0A1C	D206D3B4	C13745E0	D29247F0	0080C0
0026E0	C0369102	D2704780	D2F6D207	9001D5C0	D20A900F	C12C45E0	D53C45E0	D29247F0	0080E0
002700	D2F6D5D6	C4C54060	40D9D6D6	E3E0E0E0	60E0D2F0	0065D205	D58E0232	45E0D4B4	008100
002720	4830D5F2	41303001	4030D5F2	4030D268	9104D270	4780C026	45E0D4EC	9640D602	008120
002740	47F0D2F6	1833D707	D5C0D5C0	D701D5AC	D5AC4130	30D14143	D58E9540	400D4780	008140
002760	C02A4140	40014930	CC644740	C00E4850	D5F25950	D0344740	C0425850	D0849A01	008160
002780	D5AC47F0	C02E0200	400C05AD	4254D001	D6014000	C0664130	30024430	C05E47F0	008180
0027A0	D2F6D200	D5C0D5B8	0005F0F0	45E0D534	9200D602	45E0D4B4	4830D5F2	41303001	0081A0
0027C0	4030D5F2	4030D268	9102D270	4780C028	45E0D4F4	9104D270	4780C034	45E0D4EC	0081C0
0027E0	9102D270	4780C08E	D2079001	D5C0D7D3	D5ACD5AC	D200D5AF	804D5850	D5AC4158	0081E0
002800	500095D8	804E4770	C068D207	900FC0D2	47F0C06E	D205900F	C0DA4130	9018D200	008200
002820	30005000	41505001	41303001	954050D0	4780C0A2	95685000	4770C072	D2089021	008220
002840	C0E04130	90334150	50D147F0	C072D207	D5C89033	4850D5F2	45E0D544	45E0D53C	008240
002860	45E0D292	45E0D4DC	45E0D4BC	9240D5C8	D206D5C9	D5C894FE	D60147F0	D2F6D9C5	008260
002880	C7C9D6D5	4060D5D6	C4C54060	C1D3C9C1	E240D5C1	D4C54060	9122D270	4780C0AE	008280
0028A0	4850D5F2	45E0D544	91807000	4780C0AE	4810D22C	45E0D23C	91802002	4710C030	0082A0
0028C0	4100000C	47F0C02C	D5002003	D5F34770	C09C0205	902EC114	9200D5AC	D202D5AD	0082C0
0028E0	20075830	D5AC45E0	D4D4D207	9044D650	D202D5AD	20045A30	D5AC45E0	D4D4D207	0082E0
002900	905C0650	5030D5AC	45E0D4D4	D2079050	D6504810	200045E0	D28C0205	D5AC2003	008300
002920	4100D5AC	41109024	45E0D272	45E0D292	47F0C0A4	48102000	45E0D28C	48102000	008320
002940	12114770	C01C95D1	D5F2D0D6	4780C0FA	D2017001	804A58E0	D0A858F0	E0605810	008340
002960	F0009110	E12C4780	C0D64111	00D15010	D110D202	70D3D111	5830D61C	5830D618	008360
002980	5830C110	5030D5AC	D20070D6	D5AF47F0	D2F6D701	7C017001	D2027003	D228D200	008380
0029A0	70D6D228	47F0D2F6	00000045	C3D6D4D4	D6D54850	D5F245E0	D5449200	D5ACD202	0083A0

Figure C-4. Unformatted Dump (Part 15 of 31)

0029C0	05AD7001	583005AC	45E00404	0207905C	06500202	05A07004	947FD5AD	584005AC	0083C0
0029E0	45E00404	02079044	06501233	478002FE	1A340630	503005AC	45E00404	02079050	0083E0
002A00	065047F0	02F69E04	06015050	05005850	060C4130	80051853	50500504	583001F0	008400
002A20	581000AC	45E0029A	58E000AC	9608E12C	58E0E05C	5030E000	45E0D4C4	050001F5	008420
002A40	80034740	084A45E0	04C41788	47FDC04E	488001F4	411888005	5080D60C	17334338	008440
002A60	00004130	30024030	06069164	02704780	C08E9258	D385D201	D38EC092	47F0C080	008460
002A80	3C000E2D	00100A4E	12004780	C08ADA1C	45E00292	47F0D2F6	01319120	02704780	008480
002AA0	02F60207	9018D660	920005AC	02079024	80000202	9030C06C	D20205AD	801545E0	0084A0
002AC0	04049260	90090201	90080652	9261900D	0201900E	06549261	90100201	90110656	0084C0
002AE0	020205AD	801845E0	04040201	90140652	92489016	02019017	06549260	901A45E0	0084E0
002B00	029247F0	02F606C2	01000707	05AC05AC	020005AF	05E35830	05AC4130	30084030	008500
002B20	05824130	30024030	05F645E0	052C0200	60000583	92136001	433005F3	06304236	008520
002B40	00029203	60030200	600405E3	0703E035	60050203	6009D5DC	950305E3	4720C060	008540
002B60	02026000	05E047FC	06E5850	05A85830	05AC0630	4430C094	5A60D580	41606002	008560
002B80	5060061C	4840D614	5A400580	41404002	40400614	0701D5E2	05E247F0	02F60200	008580
002BA0	60005000	0E008004	80044780	C10E0707	05AC05AC	41408000	02030638	80099530	0085A0
002BC0	80044720	C1069102	06024710	C0CA5850	05A8D72F	50005000	0701D65A	065AD200	0085C0
002BE0	06588004	020005AF	40005870	05AC45E0	04FC9140	70304710	C106D202	05AD7003	0085E0
002C00	9180D5AD	4780C068	92FF05AC	5830D5AC	5A300638	50300638	92000638	95038004	008600
002C20	4700C0A2	43307001	06304235	0000321F	50014A50	00C64440	00C69AFD	065A4720	008620
002C40	C04047F0	C0824330	70010630	42300580	020005E0	05809104	06014710	C0CA9602	008640
002C60	0602D200	05E39004	020305DC	06389140	02704780	02F60202	9031C12E	0707D5AC	008660
002C80	05AC0200	05AF8002	45E004D4	0201903A	065E0203	05AC8009	45E0D4D4	02079068	008680
002CA0	065045E0	029247F0	02F69140	02704780	02F60207	9018D660	0701D5F0	05F04110	0086A0

Figure C-4. Unformatted Dump (Part 16 of 31)

002CC0	00C445E0	D4AC47F0	C0D2E3D9	C60045E0	D5349120	D2714780	C0344110	D5AC5010	0086C0
002CE0	C05A4110	D65C5010	C05E50A0	C0624110	A1005010	C0669280	CG664110	C05A1700	0086E0
002D00	0A1D9102	D2704780	C04045E0	D4F4D200	A003D615	17664110	C0524100	00320A18	008700
002D20	D3D5D2C5	C4E3F4F3	000AC4D7	58C3D6D4	F140D106	18000000	D2079018	D660D203	008720
002D40	D5ACD5DC	45E0D4D4	D2D7900F	D65D45E0	D29247F0	D2F60700	00000000	00000000	008740
002D60	00000000	1305704A	00001263	213812D0	33000000	06300000	00000000	00000000	008760
002D80	00000000	00000000	00020000	00000000	00000000	00000000	00000000	00001C40	008780
002DA0	40400000	10000000	20000E13	00030300	00000000	00048000	1F0014A1	00000000	0087A0
002DC0	00000000	00000000	C5D5C4D3	C9C24040	02D29858	70315058	F030A805	EFD20030	0087C0
002DE0	84D29A58	70315058	F030BC05	EF587031	5058F030	C405EF49	90D28C47	70D15047	0087E0
002E00	F0D1D695	F0301447	10D22C41	0000D158	70315058	F030EC05	EF410000	01587031	008800
002E20	5058F030	9C05EF18	FFD60830	85308547	70D2860A	1A18131B	0006000A	3A120047	008820
002E40	80D24247	F0D16818	0018130A	1C00FF07	0000D100	530C0000	00000000	00000000	008840
002E60	00000000	00000000	00002E6C	0000D14E	80E4D5C4	C5C6D940	40800000	0105E4D5	008860
002E80	C4C5C6D9	40400800	00013ED7	D9C9D5E3	C5D94004	000006A9	C5D5C4D3	C9C24040	008880
002EA0	A1000012	CAE4D5C4	C5C6D9F0	F090D000	1305C5D5	C4D3C9C2	4040A100	00295240	0088A0
002EC0	40404040	40404040	40404040	40404040	40404040	40404040	40404040	40404040	0088C0
0088E0 TO 008960 SAME AS ABOVE									
002F60	40404040	40404040	40404040	00002D68	000014D6	CE372400	00000000	00000000	008960
002F80	0480D7D9	58C9D6C5	4040D326	74D1D938	D4D6C4E4	D3C540C7	C5D5C5D9	C1E3C5C4	008980
002FA0	40E5C9C1	40E3D9C1	D5E2C1E3	4040130C	0200D600	00000000	0004AA5D	96C95850	0089A0
002FC0	0000000F	0802D007	00000000	1176C359	40C00000	0F080200	08000000	001176C3	0089C0
002FE0	594C0000	000F0802	00090000	00001176	C3594C40	00000F08	02000500	00000011	0089E0
003000	76C3594D	8000D00F	0802D00A	0G00D0D0	1176C359	4C800000	0F080200	08000000	008A00

Figure C-4. Unformatted Dump (Part 17 of 31)

003020	003176C3	59404000	000F0E02	000C3000	00001176	C3594000	00000F08	02000000	008A20
003040	00000011	76C3594C	C0000000	00078830	00009800	00385800	00079000	00074800	008A40
003060	00000000	00000000	00000000	000030F7	00003074	00001208	40330201	A8000000	008A60
003080	01060000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	008A80
0030A0	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	008AA0
	008AC0 TO 008B00 SAME AS ABOVE								
003100	00000000	00000000	00000000	00023030	00000000	00000000	00000000	00000000	008B00
003120	00001C40	40400000	10000000	2000JE03	01020380	00107800	00000001	1F001400	008B20
003140	00000000	00000000	00000000	C5D5C433	C5C24040	40404040	40404040	40404040	008B40
003160	40404040	40404040	40404040	40404040	40404040	00002F70	04A8D200	04A90001	008B60
003180	00000430	00013000	00000001	00003030	47F0D2F8	41C00008	47F0D2F8	45E0D898	008B80
0031A0	41107000	9180D1E9	47E0C05A	41303704	17004300	07000507	70003000	4780C056	008BA0
0031C0	41303009	4600C01A	9200D4AA	921434A8	45E0D880	47F0D2F6	41300010	47F0C048	008BC0
0031E0	41300008	92107013	96186000	18C347F0	D2F822FF	30089210	04A84850	092A45E0	008BE0
003200	D8AD47F0	D2F647F0	C08A9180	20084710	C07A45EC	087045E0	D678960A	600041C0	008C00
003220	001047F0	D2F89208	70139618	600047F0	C0820700	07000700	008E011A	000E0709	008C20
003240	D4AC0446	9208D4A8	D20304AC	70059500	70024780	C08245E0	D8584110	700545E0	008C40
003260	D8C89108	60004710	C04A4100	00429241	700F45E0	D510D205	7005D4A0	92007010	008C60
003280	41CC0008	47F0D2F8	91106000	47E0C05A	9250700F	47F0C038	D201D4A6	60014810	008C80
0032A0	D4AE45E0	D28C5800	D4ACD202	D111203A	9200D110	58000110	5000D4AC	9218D4A8	008CA0
0032C0	92FF0445	41107009	4850092A	45E0D8A0	47F0D2F6	47F0C09E	47F0C0A2	9208700F	008CC0
0032E0	4010D110	D2017003	D11047F0	C0423737	70007000	4810D6CC	12114720	C03445E0	008CE0
003300	D8989280	D4A845E0	D4EE4010	D6CC45E0	D87845E0	D858960A	600041C0	001047F0	008D00
003320	D2F845E0	D28C4300	200945E0	04FE4230	2009D600	D0FE00FC	D202200A	7006D502	008D20

Figure C-4. Unformatted Dump (Part 18 of 31)

003340	200D700A	4780C020	0202200D	700A47F0	C02045E0	D8989280	D4A84110	700D45E0	008D40
003360	D85047F0	D2F647F0	C0F69520	20084780	C0AA9104	20084710	C0BE9180	20084710	008D60
003380	C0D69108	20084710	C0EA9111	200847E0	C0CE4830	200645E0	D4C61813	4840D268	008D80
0033A0	12114780	C09445E0	D2BC910A	20084770	C088D502	D4AD200D	4780C070	D202D4AD	008DA0
0033C0	200D45E0	D4CE4810	200A4300	200945E0	D4FE4200	D26947F0	C04E4100	003645E0	008DC0
0033E0	D91047FC	C070D200	D4A9D269	4040D268	181345E0	D2BCD201	D4A62006	45E0D4E6	008DE0
003400	96DA6000	45E0D878	41C00010	47F0D2F8	43002009	45E0D4FE	4200D4A9	47F0C0A4	008E00
003420	91D220D8	4710C0EA	D502D4AD	200D479D	C0BE0202	D4AD200D	47F0C0BE	4100D036	008E20
003440	45E0D91D	47F0C0BE	45E0D4EE	47F0C0AE	0002D501	D4C4D6C0	4780C010	D2D1D4C4	008E40
003460	D6C09180	D1E94780	C0241837	4170D7D0	45E0D918	18735830	D10C4130	30035930	008E60
003480	D1D44780	C16A9106	30004780	C028D201	D1103001	4810D110	45E0D2BC	D600D0FD	008E80
0034A0	D0FC9104	30004710	C07A1700	400D20JA	D200200C	D26994FD	30009180	200847E0	008EA0
0034C0	C028D202	200A200D	47F0C028	91803000	4710C0A2	96103000	91403000	4710C028	008EC0
0034E0	91D42008	4780C028	92202008	9610D218	47F0C028	91403000	4710C0D6	91D42008	008EE0
003500	4780C0D6	91102008	4710C0BE	9610D26E	D2002009	D2699120	300047E0	C028D200	008F00
003520	200AD269	47F0C028	43002009	45E0D4FE	42002009	4340200C	91082008	47E0C124	008F20
003540	4140D0FF	91102008	4780C124	185145E0	D4F64300	200945E0	D4FE4200	2009D600	008F40
003560	D0FD00FC	91802008	4710C028	4340200C	181545E0	D2BC9120	30004780	C0289500	008F60
003580	200A477D	C0289180	20084710	C0289104	20084710	C15A1804	45E0D27A	47F0C156	008F80
0035A0	47F0C156	47F0C028	9610D26E	D200200A	D269D600	D0FD00FC	47F0C028	9110D218	008FA0
0035C0	4780C182	94EF0218	5830D10C	41303003	5930D1D4	4780C182	91843000	47E0C17A	008FC0
0035E0	D2D1D110	30014810	D11045E0	D28C9520	20084770	C17A9204	2008D600	D0FD00FC	008FE0
003600	47F0C17A	947F01E9	92F0D757	94FD01E9	5810D6BC	580D10E0	45E0D29A	45E0D888	009000
003620	47F0D2F6	47F0D2F6	00400004	45E0D28C	D2D12006	D6EED600	D0FD00FC	18314810	009020

Figure C-4. Unformatted Dump (Part 19 of 31)

003640	D6CA4030	D6CA1211	4780C034	45E0D23C	40302006	D600D0FD	D0FC1813	47F0D2F6	009040
003660	4030D48E	47F0C02E	18F545E0	D7B447FD	D2F647F0	COAA9520	20084780	C0969110	009060
003680	D4A847E0	C09E9104	20084710	C0889110	20084710	C0789180	200847E0	C0889108	009080
0036A0	D4A84780	C04E4100	003645E0	D91041C0	000447F0	D2F8D2D0	2009D269	D202200A	0090A0
0036C0	D4AAD600	D0FD00FC	D502200D	D4AD4790	C070D202	200DD4AD	41C00008	47F0D2F8	0090C0
0036E0	91112008	47E0C046	4810200E	47F0C046	9108Q4A8	4780C09E	D20D4A9	200945E0	0090E0
003700	D4EE47F0	C0709104	20084780	C07847F0	C09045E0	D4EE47F0	C070D600	40D75BC3	009100
003720	D6D4F640	00059200	200A4780	C1089640	D4A89610	D26E4240	D4A99108	D4A84780	009120
003740	C14E4250	D4A9910E	20084780	C14E1831	4810D4A6	45E0D2BC	424D2009	D600D0FD	009140
003760	D0FC9180	20084780	C14094BF	D4A81813	45E0D2BC	91042008	4710C156	91102008	009160
003780	4710C15E	45E0D4E6	47F0C07A	183192A8	D4A045E0	D4DE1851	181345E0	D2BC0200	009180
0037A0	200CD269	4840200E	4050200E	D20D2009	D4A99108	D4A84710	C19094FD	200E0E00	0091A0
0037C0	D0FD00FC	181445E0	D2BC4050	200A0600	D0FD00FC	181547F0	C07A9104	20084780	0091C0
0037E0	C0CE0200	D4AA200A	4250D4A9	47F0C156	45E0D4EE	47F0C07A	9500D269	18314E10	0091E0
003800	D6CA4030	D6CA1211	4770C018	4030D43E	47F0C026	45E0D2BC	40302006	D600D0FD	009200
003820	D0FC1813	47F0D2F6	9835D4A4	45E0D23C	4010D4AE	4010D4AE	D203D4A9	20099213	009220
003840	D4A89108	20084710	C02694FD	D4A892A8	2000D600	D0FD00FC	45E0D4EE	9035D4A4	009240
003860	5830D10C	41303003	5930D1D4	4790D2F6	91043000	4780C03C	D5013001	200E4770	009260
003880	C03C4010	D110D201	3001D110	47F0C03C	F060F2F3	5D40C9E2	40C9D5E5	C1D3C9C4	009280
0038A0	40D6D940	D3C5E2E2	40E3C8C1	D540J9C5	C3D6D9C4	40D3C5D5	C7E3C84B	40C2D3D6	0092A0
0038C0	C3D240D3	C5D5C7E3	C840C9E2	40C1E2E2	E4D4C5C4	40C5D8E4	C1D340E3	D640D9C5	0092C0
0038E0	C3D6D9C4	40D3C5D5	C7E3C84B	D057F0F2	F5D9C5C3	D6D9C440	D3C5D5C7	E3C8404D	0092E0
003900	C3D6D3E2	40F2F460	F2F75D40	C9E240C9	D5E5C1D3	C9C440D6	58703150	58F030DC	009300
003920	D5EF9200	3014D609	308E3085	4770D242	D20DD299	30845E70	315058FD	308805EF	009320

Figure C-4. Unformatted Dump (Part 20 of 31)

003940	0200029A	30840200	30840299	95F03015	4780D1CC	95F0307B	4780D1A0	9101029E	009340
003960	4780D1B4	47F0D1CC	91F0D29B	4710D1CC	00E405C4	C5C6D940	40000098	92010298	009360
003980	47F0D1F4	0207D29C	307C92FF	029B070E	3078307B	58703150	58F030C8	05EF1E99	009380
0039A0	58703150	58F030CC	05EF910F	029B4780	01F492F0	029B0208	3078D298	95F03015	0093A0
0039C0	4780D242	9102D298	4780D20E	58703150	58F030D4	05EF9602	029E5670	31505EFO	0093C0
0039E0	30A805EF	02003084	D29A5870	315058F0	303C05EF	58703150	58F030C4	05EF4590	0093E0
003A00	028C4770	015047F0	01D60000	00000030	022C4100	00Q15270	315058F0	30EC05EF	009400
003A20	41000001	58703150	58F0309C	05EF18FF	06083085	30854770	02860A1A	18131E00	009420
003A40	06000A3A	12004780	024247F0	01E81B00	1E130A1C	00FF0700	00010053	0C000000	009440
003A60	00000000	00000000	00000000	00000F20	58F03188	07FF45E0	0098F2F9	1E110A07	009460
003A80	5E10311C	50010010	F3531008	101096F0	1000D203	1000100A	02011004	100E0201	009480
003AA0	1000100C	02011002	100A0201	10041008	02050000	1000F211	0000100E	F2350000	0094A0
003AC0	1000F211	0000100A	F2110000	100C07FE	4770D274	13FF0000	00C3F160	C3F940C3	0094C0
003AE0	C8C1C9D5	C9D5C740	D3D6C7C9	C3110000	0234E3C1	C2D3C540	C9D5D7E4	E3E1D6E4	0094E0
003900	E3D7E4E3	15000008	58C9D5D7	E4E3E1D6	E4E3D7E4	E340C9D5	E3C5D9C3	C5D7E30F	009500
003820	FF000000	D9C1C6E1	E3C1C740	C6C9D3C5	40D9E3D5	15000002	38C9D5D7	91FF3001	009520
003840	4780C070	91403001	4780C02C	91413001	4710C022	0201D385	C19247F0	C03A0201	009540
003860	D385C194	47F0C03A	91253001	4710C054	0201D385	C196925B	D3840700	47F0C04A	009560
003880	3C000E2C	00100A4E	12004780	C054JA1C	024F03C0	300245E0	029247F0	02F6D207	009580
0038A0	F000E000	41F0F00A	47F0C090	05013002	C1884770	C0AA1755	43503000	41E03C04	0095A0
0038C0	41F0D3D0	95FFE008	4770C062	41E0E009	4650C088	41E0D3D0	19EF4780	02F645E0	0095C0
0038E0	02929201	30001755	43503000	05013002	C18E4770	C0C40207	D3343004	47F0C0E0	0095E0
003C00	05013002	C1844780	C0BA9A01	D2309110	D2704780	D2F6D207	D3B5C18A	41E0C198	009600
003C20	0201D110	300241F0	008049F0	01104770	C1049261	D3BF0207	D3C0D208	920E03CE	009620

Figure C-4. Unformatted Dump (Part 21 of 31)

003C40	41F00081	49F03110	4770C11E	92E1333F	D207D3C0	02009208	D3C841F0	008249F0	00964D
003C60	D1104770	C1389261	D3BF0207	D3C0D210	920E03CE	4AE0D110	41F0C21A	19EF47B0	009660
003C80	C178925B	D38C3201	D38DE000	47F0C15A	3C00DE34	00100A9E	120047B0	C1E4DA1C	00966D
003CA0	95E0D3B5	4770C174	92D2D38A	92F0333B	45E0D292	41303002	4E50C0CE	47F0D2F6	0096AD
003CC0	00E80066	00E2E060	60606060	60E03068	00E900EA	0009000A	00080020	00210022	0096C0
003CE0	00230024	00250026	00270028	0029302A	002B002C	002D002E	002F0030	00310032	0096E0
003D00	00330034	00350036	00370038	0039303A	003B003C	003D003E	003F0040	00410042	009700
003D20	00430044	00450046	00470048	0049304A	004C004D	004E004F	00500067	00E60051	009720
003D40	00520053	00540055	00560057	00583059	005A005B	0061E340	0000D1F0	CE16FF02	009740
003D60	13000000	0440F454	91041046	4710F454	92421038	96201032	07F850FF	02550000	009760
003D80	00045491	40100207	8E922310	389E1010	3207FE91	7F103247	80F49291	80104847	009780
003DA0	80F48C91	40104847	80F48E98	EC000C58	0010580A	3D98EC00	0C0A3D41	E0F4920A	0097A0
003DC0	3D914010	484780F4	A498EC00	0C583010	5807FF98	EC000C07	FF0E0002	0303000C	0097C0
003DE0	00020000	0000021F	00142C00	00000000	00000000	000000C5	05C403C9	C2404037	0097E0
003E00	00000000	00000000	00000000	07D95BC9	06C14040	75040818	08474040	40404040	009800
003E20	40404040	40404040	40404040	40404040	40404040	40404040	13080100	00000000	009820
003E40	00000002	28D7D958	C9D6C140	403A9601	104A9120	10024780	00003F60	000014A2	009840
003E60	9E1B0002	13000000	0440F454	91041046	4710F454	92421038	96201032	07F85000	009860
003E80	02550000	00045491	40100207	8E922310	389E1010	3207FE91	7F103247	80F49291	009880
003EA0	80104847	80F48C91	40104847	80F48E98	EC000C58	0010580A	3D98EC00	0C0A3D41	0098A0
003EC0	E0F4920A	3D914010	484780F4	A498EC00	0C580010	5807FF98	EC000C07	FF0E2002	0098C0
003F00	F0F00840	04F0F091	40104847	80F01850	00105858	00105490	EC000C50	E0D01041	009980
003FA0	A0F18041	80F1E292	00103291	40104E47	10F03E92	13103896	02103207	F8910210	0099A0
003FC0	4E071895	20103147	80F05692	14103896	02103207	F8185058	40105048	90102C91	0099C0

Figure C-4. Unformatted Dump (Part 22 of 31)



003FE0	18104647	20F06A05	EA483010	4C91345A	02D15200	40000070	10484710	F0029101	0099E0
004000	10484710	F0929110	10464710	FOAE1864	18694330	600147F0	F0829548	10444770	009A00
004020	FOA25830	105847F0	F0B24330	10445833	000047F0	F0B24330	50011839	4780F0BA	009A20
004040	1B334930	104C4700	F0CE4802	28D7D95B	C906C140	403A9601	104A9120	10024780	009A40
004060	00003D58	000019FB	35900002	88000000	03574547	70F36450	50105847	F0F3721B	009A60
004080	44434010	45414400	00505040	0091FF10	434770F3	8C910610	3C4710F3	8C960110	009A80
0040A0	32921810	3807F80A	00911810	464770F3	9805EA91	80103C47	80F3A494	F3104A91	009AA0
0040C0	80104A47	80F3AE07	FB910410	4A078B47	F0FOAED2	00400050	00918010	024710F3	009AC0
0040E0	C80A0191	02104A47	80F3E894	FD104A96	04104A91	01104A47	80F3E867	00025F00	009AE0
004100	000003E0	94FE104A	9608104A	917F1002	078E9108	10024780	F43A91E0	104A4780	009B00
004120	F43A94F7	10029118	10464780	F4129604	104A47F0	F4169602	104A9101	10494780	009B20
004140	F43A9108	10164780	F43A9118	10464780	F4369608	104A47F0	F43A9601	104A9120	009B40
004160	10024780	00003E5C	C5080000	00000008	00060000	00000000	00000000	00000000	009B60
004180	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	009B80
009BA0 TO 009E60 SAME AS ABOVE									
004460	00000000	00000000	00000000	0A0A3233	34353637	38390A0A	0A0A0A0A	40414243	009E60
004480	44454647	48490A0A	0A0A0A0A	FCFC3303	8D31D191	A1C1F1E1	89B9D999	A9C9F9E9	009E80
0044A0	88B2D292	A2C2F2E2	8ABADA9A	AACAFAEA	8EB4D494	A4C4F4E4	8C8C0C9C	ACCCFCEC	009EA0
0044C0	8F30D090	A0C0F0E0	88B8D898	A8C8F8E8	0085D595	A5C5F5E5	8D395919	29497969	009EC0
0044E0	01B3D393	A3C3F3E3	8B3A5A1A	2A4A7A6A	0234D696	A6C6F6E6	8E3C031C	2C4C7C6C	009EE0
004500	07B7D797	A7C7F7E7	8F385818	28487868	3D355515	25457565	0D855D1D	2D4D7D6D	009F00
004520	38335313	23437363	08835818	28487868	3E365616	26467666	0E865E1E	2E4E7E6E	009F20
004540	3F375717	27477767	0F875F1F	2F4F7F6F	05315111	21417161	0981DD9D	A0C0F0ED	009F40
004560	06325212	22427262	0A82D898	ABC8F8EB	5C865414	24447464	DC84DE9E	AECEFEFE	009F60

Figure C-4. Unformatted Dump (Part 23 of 31)

004580	04305010	20407060	0880DF9F	AFCFFFE	0F0FF0F0	00010203	04050607	08090A0B	009F80
0045A0	0C0D0E0F	10111213	14151617	18191A1B	1C1D1E1F	20212223	24252627	28292A2B	009FA0
0045C0	2C2D2E2F	30313233	34353637	38393A3B	3C3D3E3F	F0414243	44454647	48494A4B	009FC0
0045E0	4C4D4E4F	C0515253	54555657	58595A5B	5C5D5E5F	D0616263	64656667	68696A6B	009FE0
004600	6C6D6E6F	70717273	74757677	78797A7B	7C7D7E7F	80818283	84858687	88898A8B	00A000
004620	8C8D8E8F	90919293	94959697	98999A9B	9C9D9E9F	A0A1A2A3	A4A5A6A7	A8A9AAAB	00A020
004640	ACADAEAF	B0B1B2B3	B4B5B6B7	B8B9BA3B	BCBDBEBF	C0C1C2C3	C4C5C6C7	C8C9CACB	00A040
004660	CCCDCECF	D0D1D2D3	D4D5D6D7	D8D9DA3B	DCDDDEDF	E0E1E2E3	E4E5E6E7	E8E9EAEB	00A060
004680	ECEDEEEF	F0F1F2F3	F4F5F6F7	F8F9FAFB	FCFDFEFF	40D6D940	41000001	95501000	00A080
0046A0	4780F036	06000600	95601000	4780F036	41000001	D300F03A	100095C0	F03A4780	00A0A0
0046C0	F0360600	9500F03A	4770F036	06001200	07FE0040	C306D3E4	D100F014	1000DC00	00A0C0
0046E0	F014F015	01001000	F01407FE	000C3C3C	0C0C0C0D	0C0C0C0C	000C000C	0C0D0A0B	00A0E0
004700	0C0D0E0F	910C1000	071E9506	1000D100	1000F001	077E9601	100007FE	C4480071	00A100
004720	900FF1F0	91A0E001	47E0F01A	18884380	E0015480	F2384540	FOAC9140	E0014770	00A120
004740	F03A0700	47F0F02E	00000001	03200A46	12004740	F1B647F0	FOA491C0	E00147E0	00A140
004760	F0560700	47F0F04E	00000001	03403A46	12004740	F1B61812	4300E000	9110E001	00A160
004780	47E0F06A	18884540	FOAC0700	47F0F07A	00000009	00000001	0F430A46	12004740	00A180
0047A0	F1B60600	12004780	FOA49110	E00147E0	F09A4540	F1DE47F0	FOA44200	F09F0200	00A1A0
0047C0	20009000	980FF1F0	47FE0002	18303630	4430F0F0	D2119000	F0F64130	90101B38	00A1C0
0047E0	924B3000	DE119000	90124110	90004100	00121830	95401000	4770F0EE	06304430	00A1E0
004800	F1084181	30009240	800047F0	F0D407F4	F8709012	10004020	20202020	20202020	00A200
004820	20202020	20212060	D2001000	10011889	41003012	41A0901A	95608000	4770F128	00A220
004840	9680F230	41880001	95F08000	4740F146	95F98000	4720F146	D200A000	800041AA	00A240
004860	000147F0	F14A47F0	F1984188	00019540	80004770	F1285820	F1F84180	901A18A8	00A260

Figure C-4. Unformatted Dump (Part 24 of 31)

004880	58E0F228	18774370	E0000670	41677001	196A4740	F19806A0	89700004	1A7A4470	00A280
0048A0	F1929180	F2300784	4360E000	06601A62	07F4F200	2000901A	47F0F1A2	3C0048D2	00A2A0
0048C0	00400A96	12004740	F1AA980F	F1F047F0	F0005804	A340980F	F1F092F0	30E59206	00A2C0
0048E0	31C447FE	00029013	4770819C	41110004	95018082	47808182	91C01014	47708182	00A2E0
004900	920180B2	58E00018	92F01001	47F0839A	96F03015	0208307B	808E47F0	80789501	00A300
004920	80804780	81A89201	80815080	001458F0	00000000	00007FEA	00000000	58A10008	00A320
004940	FFFF1FB2	1FB81FC0	207718E0	1FD221F0	216F220C	24912422	0000000F	D9C10506	00A340
004960	900FF188	18500650	5C40F320	5A5030F4	58605000	48705008	1A765070	F1C89500	00A360
004980	E0004770	F0384E10	9030F321	60079036	96F06009	47F0F06A	18774370	E0000670	00A380
0049A0	910BE002	47E0F050	4470F1CC	47F0F06A	41C77000	44C0F1E4	414C6007	D2014000	00A3A0
0049C0	F20041CC	000244C0	F108D200	6010F330	45C0F090	D2066000	F34E4540	F16C1222	00A3C0
0049E0	4780F088	45C0F0D8	980FF188	47FE0004	4180300C	41903098	41A0F202	4180601E	00A3E0
004A00	95F08000	4770F0B2	D2018000	A00041BB	00035980	F1C84740	F0C8D20D	6010F330	00A400
004A20	4540F16C	4180307C	41880001	41AA0032	19894740	F0A007FC	189991E0	E00347E0	00A420
004A40	F0F24870	20024390	20015880	200447F0	F0FE4170	00014390	E0010690	18829110	00A440
004A60	E00347E0	F10A5888	00009108	E00347E0	F13E4880	500848B0	F33E1998	4740F136	00A460
004A80	D20D6000	F3244480	F1D24540	F16C1B9B	1A8B47F0	F11A4490	F1D247F0	F15841A9	00A480
004AA0	900044A0	F1EA414A	600ED201	4000F200	41AA0002	44A0F1DE	41889001	D20D6000	00A4A0
004AC0	F3404540	F16C4670	F10A07FC	920250DE	9201501D	92015012	180558D0	30A005E0	00A4C0
004AE0	98EF E040	07F44780	83529200	803447F0	83E0181F	D5000000	101C4770	838892F0	00A4E0
004B00	30849201	80B447F0	82429200	803448C0	80AE12CC	47708242	47F0818E	91081014	00A500
004B20	478083CC	48C080AE	46C083B6	D2006037	1000D200	600E8000	DE006006	1000DE00	00A520
004B40	E0008000	D2006007	F1F0D200	600EF1F0	20202020	20202020	20202020	20202020	00A540
004B60	2160F0C1	F0C2F0C3	F0C4F0C5	F0C6F0C7	F0E5F1D7	D309F0F0	F0F1F0F2	F0F3F0F4	00A560

Figure C-4. Unformatted Dump (Part 25 of 31)

004880	F0F5F0F6	F0F7F0F8	F0F9F1F0	F1F1F1F2	F1F3F1F4	F1F5F1F6	F1F7F1F8	F1F9F2F0	00A580
0048A0	F2F1F2F2	F2F3F2F4	F2F5F2F6	F2F7F2F8	F2F9F3F0	F3F1F3F2	F3F3F3F4	F3F5F3F6	00A5A0
0048C0	F3F7F3F8	F3F9F4F0	F4F1F4F2	F4F3F4F4	F4F5F4F6	F4F7F4F8	F4F9F5F0	F5F1F5F2	00A5C0
0048E0	F5F3F5F4	F5F5F5F6	F5F7F5F8	F5F9F6F0	F6F1F6F2	F6F3F6F4	F6F5F6F6	F6F7F6F8	00A5E0
004900	F6F9F7F0	F7F1F7F2	F7F3F7F4	F7F5F7F6	F7F7F7F8	F7F9F8F0	F8F1F8F2	F8F3F8F4	00A600
004920	F8F5F8F6	F8F7F8F8	F8F9F9F0	F9F1F9F2	F9F3F9F4	F9F5F9F6	F9F7F9F8	F9F9D3F0	00A620
004940	D3F1D3F2	D3F3D3F4	D3F5D3F6	D3F7D3F8	D3F9D4D9	C8F0C8F1	C8F2C8F3	C8F4C8F5	00A640
004960	C8F6C8F7	C8F8C8F9	E4F1E4F2	E4F3E4F4	E4F5E4F6	E4F7E4F8	E4F990DE	914898AD	00A660
004980	00000020	D9C5E2E4	D3E340C6	C9C5D3C4	C9D5C4C9	C3C1E3D6	D9E240D6	D560000D	00A680
0049A0	D9C5E2E4	D3E340C6	C9C5D3C4	E040C4C5	C2E4C7E0	40EE0000	5010F02C	411AD020	00A6A0
0049C0	41110001	191047B0	F01CD500	A0201000	4780F008	06109258	10005810	F02C0410	00A6C0
0049E0	07F10000	00000000	90CEF058	4810F064	D201F01A	100217EE	43E01001	88E00004	00A6E0
004A00	41000000	1A0E18CD	4E00F02E	41E0E001	18CE47F0	F03241C0	0001924E	C000F200	00A700
004A20	F066D000	F800F066	F06647B0	F04A3260	C000D300	D000F067	98CEF058	47F01010	00A720
004A40	00000000	00000000	00000000	001030F1	940F9000	F8779010	F071F9F7	9000F071	00A740
004A60	4780F01A	F67F9010	9000F8FF	90189000	4720F042	4740F030	F87F9010	900007FE	00A760
004A80	92F03085	92C531C4	509031C8	58E031A0	07FE9200	9018F0F7	90189010	F9779018	00A780
004AA0	9010078E	FA779018	9010F070	9018F070	F8769010	9018F8FF	90189000	47F0F042	00A7A0
004AC0	2C999999	99999999	9C3E3F3A	3B1C1425	9200F088	D6001200	4780F03A	95401000	00A7C0
004AE0	4770F022	9101F088	071E9602	F08847F0	F03295F0	1000074E	95F91000	072E9601	00A7E0
004E00	F0884111	00014600	F00C9540	10004770	F0509101	F088071E	9602F088	47F0F072	00A800
004E20	D300F086	1000D100	F0871000	9509F087	072E95C0	F086074E	95E0F086	078E9601	00A820
004E40	F0889502	F088478E	00049702	F0889502	F08847FE	00040000	00101E26	CA02FC76	00A840
004E60	9045F054	1B445850	313C4340	0001D201	F03AD002	D201F040	D0044440	F0364440	00A860

Figure C-4. Unformatted Dump (Part 26 of 31)

004E80	F03C4440	F0424440	F0484440	F04E9845	F05447FD	000E0200	F058F058	D200F158	00A880
004EA0	F158DC00	F0585000	DC00F158	50003530	F058F158	778E8F08	08092124	15EE9C9D	00A8A0
004EC0	72730B08	09222415	EF767772	730C383A	211F240D	E2969772	7311080A	21240FD5	00A8C0
004EE0	96977273	0DE59899	90910C08	0A222024	0DE47677	8A8B0F08	0A22240D	E476778A	00A8E0
004F00	8B12C47E	770E2E1C	004FDC02	2C0B3709	21240DE0	76777273	0B070922	240DE17E	00A900
004F20	77727316	07212415	EC3A4376	770DE03A	81727313	F976773A	4318070A	22260515	00A920
004F40	EC3A4376	770DE13A	B1727313	F976773A	43110809	212415EB	3A437273	0DE07677	00A940
004F60	3A821108	09222415	EB3A4372	730DE176	773AB217	080A2124	15EC3A43	767715EE	00A960
004F80	3A917273	13F97677	3A431D08	0A222415	EC3A4376	7715EE3A	81727311	D53A473A	00A980
004FA0	9513F976	773A4306	2E1C004F	DC022C0A	07092411	D59E9F8C	8D11070A	26031305	00A9A0
004FC0	3A438C8D	0FD59A9B	3A430A08	09241035	9E9F9293	08240FD5	9A9B9293	10280431	00A9C0
004FE0	08240B4C	9A9B3AD2	26271C2D	050A2B21	7F090724	11D57677	8C8D0908	2410D576	00A9E0
005000	77929305	0828217F	09092411	D59E9F72	73100A26	0115D53A	4372730F	D59A9B3A	00AA00
005020	43051C00	4FD03A07	092411D5	76777273	0428217F	050A2B21	7F102810	3100240B	00AA20
005040	4C76770A	D226271C	2D0F2601	164E3A49	6E6F0C4F	3A497273	08180525	0201C03A	00AA40
005060	433AB614	18250116	D93E3F3A	4312C43A	A801C03A	433AA80A	0325010F	D53AB73A	00AA60
005080	47090304	260112D2	3A470B03	05260110	DC3AAC3A	A0D9269A	16F97677	3AAC0527	00AA80
0050A0	2A215F10	10260916	4E3A496E	6F084F3A	49727314	3E250116	E63A3B6E	6F0FD53A	00AAA0
0050C0	A43A4A12	D23A4A0B	0E112601	10083AA2	3AA3100E	2604160B	3A433AA2	084F3A49	00AAC0
0050E0	7273140F	250116E7	3A3B7273	0FD53AA4	3A4A12D2	3A4A0B0F	11260110	083AA23A	00AAE0
005100	A31D0F26	0116EA3A	436E6F08	C93A493A	A20B0F17	259116F5	76773A49	160F2593	00AB00
005120	01D13A49	3A48104D	76773A49	0FD59A9B	3A480A17	258F16F7	76773A49	15269101	00AB20
005140	E83A493A	B310F676	773A490F	D59A9B3A	B30A1E26	8D16F876	773AB41A	04260319	00AB40
005160	C43A3B3A	3B16E73A	3B3AB40F	D53AA73A	4A12D23A	9A0F2601	16E73A3B	3AB4150B	00AB60

Figure C-4. Unformatted Dump (Part 27 of 31)

005180	3A433A3B	051C004D	F00A0526	0110383A	483A3B09	268816F5	76773AA5	0707091C	00AB80
0051A0	0051A704	2B217F07	3E371C00	51EA0C37	2E020B4C	CDC20B65	7273051C	0051C809	00ABA0
0051C0	3E112604	09D27475	0C362602	0B4CCDC3	08D27677	042B217F	093E1126	01096570	00ABCO
0051E0	71093D10	2E020362	88B9173E	113D102E	01096570	71036288	8909D274	75036288	00ABE0
005200	89112601	00D5D2CD	65CD0B65	654B0B02	D24B0B36	373D1126	4B03D28E	89093D11	00AC00
005220	264A1F4C	88B9DC47	00000000	00000000	00000C47	00000000	00000000	00000726	00AC20
005240	010AD272	730E2D26	01102902	53062403	CAD2C0DE	240BD452	5324311C	2D0BD454	00AC40
005260	5506061C	0052C405	072B217F	11123426	0916E63E	3F6E6F07	E73E3F72	730A1226	00AC60
005280	0807F16E	6F727322	13250715	EB3A4372	7311053A	AE3A4713	DC3AA93A	A916EA3E	00AC80
0052A0	3F6E6F07	DC3E3F3A	A91B2506	15EA3A43	6E6F1105	3AAE3A47	13DC3AA9	3AA907EB	00ACAD
0052C0	3AA97273	050E2B21	7F0F0B4A	28D03138	26D10AD2	2E271C6B	0A0B2607	05DF6E6F	00ACCO
0052E0	72730D2E	013EC63A	3B0D313A	4B3A3B0A	DC26020D	E03A3B72	73092601	0DDF3A3B	00ACE0
005300	6E6F0E4A	28D03138	26D10AD2	2E271C6B	0ADC2602	05DF6E6F	3A3B0926	0105E03A	00AD00
005320	3B727302	24122800	30C0240B	4C6E6F0A	D226271C	2D627208	0026010E	CD848508	00AD20
005340	0126010E	CD868705	26D12468	063A1C00	53E0063B	1C005360	08022601	0ECD8889	00AD40
005360	093E1126	01096570	710B4500	3D0D2604	0362B889	06451C00	53840A01	3D0E2602	00AD60
005380	03E28889	093D0F26	01036288	890C06D7	0B260305	E072736E	6F0C4008	3F260207	00AD80
0053A0	F172736E	6F042B21	5F06451C	0053DC0A	3A3D0C26	090366B8	890A003D	0D260103	00ADA0
0053C0	C588B90A	023D0C26	01036288	893A313D	0E260103	6688B905	1C005404	0A3B3D0C	00ADC0
0053E0	260403D4	88B90A02	3D0C2601	03628889	0A003D0D	260103D4	88B90A01	3D0E2601	00ADE0
005400	03C68889	0A422601	018CBA8B	696A0837	26010B02	D2C30726	010B6565	C2093D11	00AE00
005420	260103D2	88B90A02	3E0C2601	0E028889	06451C00	5484063A	1C0054CD	0B02003D	00AE20
005440	12260103	D288890A	003E0D26	0B0E0284	850B0201	3D122601	03D28889	18013E0E	00AE40
005460	3D0F2601	18650362	88B90ED2	86870B0D	CDC23765	0B013726	010B0C0D	C337D205	00AE60

Figure C-4. Unformatted Dump (Part 28 of 31)

005480	1C0054C0	06381C00	54C00801	02301226	01030288	890A013E	0E26040E	D286870B	00AE80
0054A0	00023012	260103D2	88B91800	3E0D3DJF	26011865	0362B8B9	0ED28485	0BCDC0C1	00AEA0
0054C0	37650800	37260108	CD0C0337	D2094112	26012065	8E8F0F37	463D1026	0109D274	00AECO
0054E0	75036688	89083726	0120D2A0	A13A423D	10260203	D288B90C	3E0F4226	0316BC8A	00AEE0
005500	88696A05	1C005512	0A240465	32338587	89C0043E	10240624	20653233	08240A65	00AF00
005520	32332231	082D2601	1D290253	0A240AE2	7273172E	1C641A28	00312C23	084C6E6F	00AF20
005540	08CD0C01	08C57677	0A022627	1C2DC3C2	0E000678	797A03D1	88B903D1	88B90800	00AF40
005560	26010E1A	84850801	26010E1A	86870702	240E1A88	89022C08	3626010B	6565C204	00AF60
005580	43260208	35260108	4C4CC104	19260304	1A260208	37260108	D2D2C30A	383D1126	00AF80
0055A0	011FC0B8	8906391C	0040E902	240C231B	573EC401	E73E3F72	73DA2800	3198240E	00AFA0
0055C0	7226270F	280030C0	26010AD2	26271C2D	CD720702	2425D288	89022408	0026010E	00AF00
0055E0	CD848506	481C0056	0E0A0626	04DDDF3A	A36E6F0E	49260315	EE3AA3EE	6F1BD23A	00AFE0
005600	A4051C00	56171126	0116E63A	A36E6FA2	CD3AA320	CD3AA30D	2800314C	26010AD2	00B000
005620	26271C2D	0A002602	726225D2	84850526	0172C006	021C0056	4106011C	00564102	00B020
005640	24072601	51D2CD3F	08022601	25D28889	08012601	25D28687	02240C28	20300024	00B040
005660	0AD22627	1C2D0925	0116EB3A	3B727306	441C0056	990B2601	0EC43E49	0EA23EA3	00B060
005680	0A042602	0CC93A3B	3E4C0F26	0108C93A	383E4C16	C93A3B3A	3B122840	30FC240A	00B080
0056A0	0226271C	2D16F576	773E4C03	202C0929	2411CD76	772A2806	2412C476	77042026	00B0A0
0056C0	02072601	12C47677	09292416	FC767776	77D22C0A	073E1526	0136CD76	770E073D	00B0C0
0056E0	15260223	4C72736E	4CB8B907	26013673	76771200	3D102601	0E028485	0362B8B9	00B0E0
005700	3ECD8485	12013D10	26010E02	868703C6	88B90ECD	86871202	3D102601	0ED28889	00B100
005720	034CB889	0ECD8889	043E1024	0708241B	73767708	2421CD76	77727307	08241273	00B120
005740	76771224	0E343A3B	19CD3A3B	727335CD	76773A3E	1C280031	28240B0D	CD720B4C	00B140
005760	6E6F0BC5	76770AD2	26271C2D	C1C3E3E4	08082000	00000050	00001024	08000000	00B160

Figure C-4. Unformatted Dump (Part 29 of 31)

```

005780 00780000 FF000000 00000000 00003030 00000000 00000000 00000000 00000000 008180
0057A0 00000000 00000000 00000000 00003030 00000000 00000000 00000000 00000000 0081A0
      00B1C0 TO 00B220 SAME AS ABOVE
005820 00000000 00003000 00000000 00003030 00000000 00000000 00000000 00000620 008220
005840 00000000 00003000 00000000 00003030 00000000 00000000 00000000 00000000 008240
      00B2E0 TO 00B280 SAME AS ABOVE
005880 00000000 00003000 00000000 00000030 000006F0 00000000 00000000 00000000 008280
0058A0 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0082A0
0058C0 00000000 00003000 00000000 00003030 00000000 00000000 00000000 11001100 0082C0
0058E0 11000000 00000000 00110000 00000000 2000FFFF 00000000 00000000 00000000 0082E0
005900 C630C3C1 09C4C9D5 4040C9D7 C5C1C630 00005040 000F4040 000040E1 40000000 008300
005920 00000000 00000000 00004040 40404030 C630D7D9 C9D5E3C5 0940D640 4040C600 008320
005940 00007840 30114040 00004000 40003030 00000000 00000000 00004040 40404000 008340
005960 C96DF0F1 F0F13017 00002C40 C9C42600 00000000 001C004F 40804031 E4004F08 008360
005980 00000000 00000000 00000000 00000000 0000000E 41001640 40F14040 4040D7D9 008380
0059A0 C9D5E3C5 09404030 17D44500 00200031 E4004F08 00000000 00000000 0000005A 0083A0
0059C0 00000000 00003000 00000000 00003030 00000000 00000000 00000000 00000000 0083C0
0059E0 00000000 00000000 00000000 00000651 00164040 40F0F740 400709C9 05E3C5D9 0083E0
005A00 40403011 FDFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE 008400
005A20 FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE FEFEFEFE 008420
      00B440 TO 00B500 SAME AS ABOVE
005B00 FEFEFEFE FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 008500
005B20 FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 1176C359 4CC00106 18000000 00000000 008520
005B40 1176C359 4D000106 18000000 00000000 1176C359 4D400106 18000000 00000000 008540

```

Figure C-4. Unformatted Dump (Part 30 of 31)



0058E0	1176C359	4C800106	18000000	00000000	1176C359	4C400106	18000000	00000106	008560
005880	18000000	00000000	01061800	00000000	00000106	18000000	00000000	01061800	008580
0058A0	00000000	00000106	18000000	00000000	01061800	00000000	00000106	18000000	0085A0
0058C0	00000000	01031000	00000000	00000106	18000000	00000000	01031800	00000000	0085C0
0058E0	08A00100	10000000	00000480	00000800	00000000	15280000	08000000	00001528	0085E0

Figure C-4. Unformatted Dump (Part 31 of 31)



## Appendix D. RPG II Compilation Time Messages

### D.1. GENERAL

If any errors are detected in your program while it is being compiled, the RPG II compiler sets the UPSI byte in the communication region of the job preamble according to OS/3 system standards to indicate the type of errors that occurred.

<u>UPSI Byte</u>	<u>Setting</u>	<u>Meaning</u>
Bit 0	1	Catastrophic errors were detected in the source program. An object module was not generated.
	0	No catastrophic errors were detected.
Bit 1	1	Serious errors were detected. An object module was generated but the results will be unpredictable.
	0	No serious errors were detected.

### D.2. MESSAGES

When an error occurs during the compilation process of your program, the appropriate message from the following list is printed on the compilation output listing. An asterisk (\*) indicates that the message is printed out due to a catastrophic error. A dagger (†) indicates that the message is printed out due to a serious error. A blank indicates that the message printed out is informational or is due to a nonserious error. These indicators are not printed out with the message.

#### Message-Meaning/Corrective Action

* NOTE000	END OF FILE HAS BEEN ENCOUNTERED IN INPUT STREAM PRIOR TO AN OUTPUT SPECIFICATION. EXECUTION IS DELETED.
† NOTE001	COMPILE TIME TABLE IS NOT IN SEQUENCE
NOTE002	COMPILE TIME TABLE HAS TOO MANY ENTRIES

**Message-Meaning/Corrective Action**

- NOTE003      INVERTED PRINT ENTRY (COLUMN 21) IS INVALID. ENTRY OF I IS ASSUMED.
- \* NOTE004      RPG CONTROL CARD IS MISSING. COMPILATION IS BYPASSED.
- NOTE005      DEBUG OPERATIONS ARE NOT GENERATED AS DEBUG OPTION IS NOT SPECIFIED IN HEADER CARD (COL. 15)
- † NOTE006      FILE TYPE (COLUMN 15) IS INVALID. SPECIFICATION IS NOT PROCESSED.
- † NOTE007      INVALID ENTRY IN COLUMNS 28, 31 OR 32. SPECIFICATION IS NOT PROCESSED.
- † NOTE008      LENGTH OF KEY OR RECORD ADDRESS FIELD (COLUMNS 29-30) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- NOTE009      MORE THAN ONE RECORD ADDRESS FILE IS PRESENT. SUCCEEDING ONES ARE NOT PROCESSED.
- NOTE010      EXTENSION CODE (COLUMN 39) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- NOTE011      INPUT FILE DESIGNATION (COLUMN 16) IS INVALID OR MISSING. ENTRY OF R IS ASSUMED FOR TAG FILE. OTHERWISE, S IS ASSUMED.
- NOTE012      OVERFLOW INDICATOR (COLUMN 33) IS NOT 0. ENTRY OF 0 IS ASSUMED.
- † NOTE013      OVERFLOW INDICATOR (COLUMNS 33-34) IS INVALID. ENTRY OF BLANKS IS ASSUMED.
- † NOTE014      MORE THAN ONE PRIMARY FILE IS SPECIFIED. FILE IS ASSUMED TO BE A SECONDARY FILE.
- † NOTE015      MODE OF PROCESSING (COLUMN 28) IS INVALID. ENTRY OF R IS ASSUMED.
- † NOTE016      FIXED FORMAT IS SPECIFIED, BUT BLOCK LENGTH IS NOT A MULTIPLE OF RECORD LENGTH. BLOCK LENGTH IS INCREASED TO NEXT HIGHER MULTIPLE.
- NOTE017      TYPE FILE ORGANIZATION (COLUMN 32) IS NOT BLANK. ENTRY OF BLANK IS ASSUMED.



**Message-Meaning/Corrective Action**

- NOTE033 NAME OF LABEL EXIT (COLUMNS 54-59) MUST BE BLANK FOR DEVICES OTHER THAN TAPE OR DISC. ENTRY OF BLANKS IS ASSUMED.
- † NOTE034 OVERFLOW INDICATOR (COLUMNS 33-34) IS APPLICABLE TO OUTPUT FILES ONLY. ENTRY OF BLANKS IS ASSUMED.
- † NOTE035 FILE ADDITION ENTRY FOR INDEX SEQUENTIAL (COLUMN 66) IS INVALID. ENTRY OF A IS ASSUMED.
- † NOTE036 KEY LENGTH OR RECORD ADDRESS (COLUMNS 29-30) IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
- NOTE037 PERCENT CYLINDER OVERFLOW (COL. 67) IS MISSING OR INVALID. NO OVERFLOW ALLOCATED FOR ISAM FILE LOAD.
- † NOTE038 INVALID REWIND OPTION. ENTRY OF U IS ASSUMED FOR A TAPE FILE. ENTRY OF BLANK IS ASSUMED FOR NON-TAPE FILE.
- † NOTE039 FORM TYPE (COLUMN 6) IS INVALID OR OUT OF SEQUENCE. SPECIFICATION IS NOT PROCESSED.
- \* NOTE040 FILE DESCRIPTION SPECIFICATIONS ARE MISSING. EXECUTION IS DELETED.
- ↓  
NOTE041 WARNING: PRIMARY FILE NOT SPECIFIED. IF SECONDARY FILES ARE SPECIFIED, THE FIRST ONE IS ASSUMED PRIMARY. OTHERWISE, LR MUST BE SET ON TO TERMINATE THE PROGRAM.
- ↑  
NOTE042 WARNING - FILE EXTENSION OR LINE COUNTER SPECIFICATION IS MISSING.
- † NOTE043 FILENAME (COLUMNS 7-14) IS MULTI-DEFINED. SPECIFICATION IS NOT PROCESSED.
- NOTE044 WARNING - LENGTH OF RAF FIELD (COLUMNS 29-30) MUST BE 10. ENTRY OF 10 IS ASSUMED.
- † NOTE045 DIRECT FILE CANNOT BE SPECIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE046 KEY FIELD LOCATION IS INVALID. SPECIFICATION IS NOT PROCESSED.
- NOTE047 COLUMNS 66-67 MUST BE BLANK. ENTRY OF BLANKS IS ASSUMED.
- NOTE048 RECORD LENGTH (COLUMNS 24-27) IS INVALID. ENTRY OF 132 IS ASSUMED.

**Message-Meaning/Corrective Action**

- † NOTE049 MORE THAN ONE RECORD ADDRESS FILE IS SPECIFIED ON FILE EXTENSION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE050 'FROM FILENAME' (COLUMNS 11-18) IS NOT SPECIFIED ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE051 EXTENSION CODE (COLUMN 39) OF FILE DESCRIPTION SPECIFICATION IS NOT E. SPECIFICATION IS NOT PROCESSED.
- † NOTE052 LENGTH OF TABLE ENTRY (COLUMNS 40-42 OR 52-54) EXCEEDS 256 CHARACTERS FOR AN ALPHANUMERIC FIELD. ENTRY OF 256 IS ASSUMED.
- † NOTE053 CHAINING FIELD (COLUMN 9-10) IS MISSING, INVALID, OR NOT RIGHT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE054 NUMBER OF FIELD NAMES EXCEEDS ALLOCATED MEMORY STORAGE. ADDITIONAL SPECIFICATIONS CONTAINING TABLE OR ARRAY NAMES WILL NOT BE PROCESSED.
- † NOTE055 'TO FILENAME' (COLUMNS 19-26) IS NOT SPECIFIED AS ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE056 'TO FILENAME' (COLUMNS 19-26) IS NOT SPECIFIED AS A CHAINED FILE ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE057 LENGTH OF TABLE ENTRY (COLUMNS 40-42 OR 52-54) EXCEEDS 15 DIGITS FOR A NUMERIC FIELD. ENTRY OF 15 IS ASSUMED.
- † NOTE058 'TO FILENAME' (COLUMNS 19-26) IS NOT SPECIFIED AS A PRIMARY OR SECONDARY FILE ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE059 TABLE SEQUENCE (COLUMNS 45 OR 57) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- † NOTE060 TABLE NAME (COLUMNS 27-32 OR 46-51) IS MULTI-DEFINED. SPECIFICATION IS NOT PROCESSED.
- NOTE061 'TO FILENAME' (COLUMNS 19-26) IS NOT SPECIFIED AS ON FILE DESCRIPTION SPECIFICATION. ENTRY OF BLANKS IS ASSUMED.

**Message-Meaning/Corrective Action**

- † NOTE062 'TO FILENAME' (COLUMNS 19-26) IS NOT SPECIFIED AS AN OUTPUT FILE ON FILE DESCRIPTION SPECIFICATION. ENTRY OF BLANKS IS ASSUMED.
- † NOTE063 TABLE NAME (COLUMNS 27-32 OR 46-51) IS MISSING, INVALID, OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- NOTE064 FIRST THREE CHARACTERS OF TABLE NAME (COLUMNS 27-29 OR 46-48) ARE NOT 'TAB'. ENTRY OF 'TAB' IS ASSUMED.
- † NOTE065 NUMBER OF TABLE ENTRIES PER RECORD (COLUMNS 33-35) IS MISSING OR INVALID. ENTRY OF 1 IS ASSUMED.
- † NOTE066 NUMBER OF TABLE ENTRIES PER TABLE (COLUMNS 36-39) IS MISSING OR INVALID. ENTRY OF 1 IS ASSUMED.
- † NOTE067 LENGTH OF TABLE ENTRY (COLUMNS 40-42 OR 52-54) IS MISSING OR INVALID. ENTRY OF 1 IS ASSUMED.
- NOTE068 FORMAT OF TABLE ENTRY (COLUMN 43 OR 55) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- NOTE069 'DECIMAL POSITIONS' (COLUMN 44 OR 56) IS INVALID. ENTRY OF ZERO IS ASSUMED.
- † NOTE070 RECORD SEQUENCE OF THE CHAINING FILE (COLUMNS 7-8) IS INVALID. BOTH POSITIONS MUST BE EITHER NUMERIC OR ALPHABETIC. SPECIFICATION IS NOT PROCESSED.
- † NOTE071 FILENAME (COLUMNS 7-16) IS NOT SPECIFIED AS ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE072 OVERFLOW OR HOME PAPER CHANNEL IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
- † NOTE073 FILENAME IS NOT SPECIFIED AS AN OUTPUT FILE OR AN OUTPUT FILE REQUIRING A LINE COUNTER SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE074 LINE NUMBER OR CHANNEL NUMBER IF INVALID OR MISSING. SPECIFICATION IS NOT PROCESSED.
- † NOTE075 CHANNEL NUMBER IS MULTI-DEFINED. SPECIFICATION IS NOT PROCESSED.
- † NOTE076 LINE COUNTER FILE NAME (COLUMNS 7-14) IS MISSING, INVALID, OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.



**Message-Meaning/Corrective Action**

- † NOTE077 'FROM FILE NAME' (COLUMNS 11-18) IS MISSING. SPECIFICATION IS NOT PROCESSED.
- † NOTE078 'FROM FILE NAME' (COLUMNS 11-18) IS INVALID OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE079 'TO FILE NAME' (COLUMNS 19-26) IS INVALID OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- \* NOTE080 THERE ARE NO VALID INPUT SPECIFICATIONS IN THIS PROGRAM. EXECUTION IS DELETED.
- NOTE081 FIELD INDICATOR (COLUMNS 65-66, 67-68, 69-70) IS INVALID OR NOT A VALID ENTRY. ENTRY OF BLANKS IS ASSUMED.
- NOTE082 INDICATOR LO IS SPECIFIED AS A FIELD INDICATOR, BUT IS NOT ALLOWED. INDICATOR IS IGNORED.
- † NOTE083 FIELD-RECORD RELATION (COLUMNS 63-64) IS INVALID. ENTRY OF 00 IS ASSUMED.
- † NOTE084 IMPROPER VALUE DETECTED IN (COLS 21-24, 28-31, 35-38); OR (COLS 44-47); OR (COLS 48-51). A VALUE 1 IS ASSUMED.
- NOTE085 NO COMPILE TIME TABLES/ARRAYS IN INPUT STREAM.
- NOTE086 WARNING - INDICATOR 00 SHOULD BE USED ONLY IN OUTPUT SPECIFICATIONS.
- NOTE087 COLUMNS 7-24 OF FIELD DESCRIPTION SPECIFICATION SHOULD CONTAIN BLANKS. ENTRY OF BLANKS IS ASSUMED.
- † NOTE088 FORM TYPE (COLUMN 6) IS NOT I, C, O, OR T, AND COLUMN 7 IS NOT AN ASTERISK. SPECIFICATION IS NOT PROCESSED.
- † NOTE089 FILENAME (COLUMNS 7-14) IS INVALID OR UNDEFINED. ALL SPECIFICATIONS FOR THIS FILE ARE NOT PROCESSED.
- † NOTE090 FILENAME (COLUMNS 7-14) IS NOT A VALID INPUT/OUTPUT FILE. ALL SPECIFICATIONS FOR THIS FILE ARE NOT PROCESSED.
- † NOTE091 'AND'/'OR' RECORD IS OUT OF SEQUENCE, I.E., FIRST INPUT/OUTPUT SPECIFICATION OR FOLLOWS FIELD DESCRIPTOR SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- NOTE092 WARNING: NO RECORD IDENTIFICATION CODES (COLUMNS 21-41) SPECIFIED ON RECORD PRECEDING 'AND' RECORD.

**Message-Meaning/Corrective Action**

- † NOTE093 'AND' RECORD CONTAINS NO RECORD IDENTIFICATION CODES. SPECIFICATION IS NOT PROCESSED.
- † NOTE094 RECORD TYPE IS OUT OF SEQUENCE I.E., FIRST INPUT/OUTPUT RECORD SPECIFICATION MISSING OR HAS AN INVALID FILENAME OR THERE ARE TWO RECORD SPECIFICATIONS WITHOUT INTERVENING FIELDS. ALL SPECIFICATIONS FOR THIS RECORD ARE NOT PROCESSED.
- NOTE095 RECORD INFORMATION AND FIELD NAME ARE PRESENT IN THE SAME SPECIFICATION. FIELD INFORMATION IS DELETED.
- NOTE096 RECORD SEQUENCE (COLS 15-16) IS INVALID OR BLANK. ENTRY OF AA IS ASSUMED IF BLANK.
- NOTE097 ALPHABETIC SEQUENCE FOUND AFTER NUMERIC SEQUENCE. NUMERIC SEQUENCE HIGHER THAN PREVIOUS NUMERIC SEQUENCE IS ASSUMED.
- \* NOTE098 NUMERIC SEQUENCE NOT ASCENDING IN FILE. ASCENDING SEQUENCE IS ASSUMED.
- NOTE099 NUMBER (COLUMN 17) IS NOT N OR 1 FOR A NUMERIC SEQUENCE. ENTRY OF N IS ASSUMED.
- NOTE100 OPTION (COLUMN 18) IS NOT 0 OR BLANK. ENTRY OF 0 IS ASSUMED.
- NOTE101 RESULTING INDICATOR (COLUMNS 19-20) IS BLANK OR INVALID. INDICATOR OF 99 IS ASSUMED.
- NOTE102 INVALID OR UNSUPPORTED STACKER SELECT ENTRY (COLUMN 42). ENTRY OF BLANK IS ASSUMED.
- NOTE103 'NOT' (COLUMNS 25, 32, OR 39) IS NOT N OR BLANK. ENTRY OF N IS ASSUMED.
- NOTE104 'C/Z/D' (COLUMNS 26, 33, OR 40) IS NOT C, Z, OR D. ENTRY OF C IS ASSUMED.
- † NOTE105 RECORD IDENTIFICATION IS OUT OF SEQUENCE - I.E., FIRST INPUT SPECIFICATION OR FOLLOWING AN INVALID 'OR', 'AND', OR FILE NAME. SPECIFICATION IS NOT PROCESSED.
- † NOTE106 'START' (COLUMNS 44-47) OR 'END' (COLUMNS 48-51) IS BLANK. ENTRY OF 1 IS ASSUMED.
- † NOTE107 'START' (COLUMNS 44-47) IS GREATER THAN 'END' (COLUMNS 48-51). 'END' IS ASSUMED TO HAVE A VALUE EQUAL TO 'START'.

**Message-Meaning/Corrective Action**

- † NOTE108      DECIMAL POSITION (COLUMN 52) IS NOT NUMERIC. ENTRY OF ZERO IS ASSUMED.
- † NOTE109      UNPACKED NUMERIC FIELD IS MORE THAN 15 BYTES LONG. LENGTH OF 15 IS ASSUMED.
- † NOTE110      ASTERISK INDICATOR MAY NOT CROSS RECORD TYPES (H, D, T)
- NOTE111      'PACKED' (COLUMN 43) IS NEITHER P NOR BLANK. ENTRY OF P IS ASSUMED.
- † NOTE112      ALPHANUMERIC FIELD LENGTH IS MORE THAN 256 BYTES LONG. LENGTH OF 256 IS ASSUMED.
- † NOTE113      FIELD INDICATOR (COLUMNS 65-66, 67-68) ENTRIES ARE NOT VALID ENTRIES FOR AN ALPHANUMERIC FIELD. ENTRY OF BLANKS IS ASSUMED.
- NOTE114      WARNING COMPILE TIME TAB/ARR DOES NOT HAVE ENOUGH ENTRIES TO FILL IT. REMAINDER OF TAB/ARR FILLED WITH BLANKS OR ZEROES.
- NOTE115      THE NUMBER OF SYMBOLS USED IN THIS PROGRAM CAUSED THE COMPILER TO RUN LESS EFFICIENTLY THAN IF AN INCREASED MEMORY SIZE WERE ALLOCATED.
- NOTE116      THERE ARE TOO MANY COMPILE TAB/ARR IN INPUT STREAM
- NOTE117      FOR RECORD TYPE SPECIFICATIONS, COLUMNS 43-74 MUST BE BLANK. ENTRY OF BLANKS IS ASSUMED.
- NOTE118      FOR 'AND' TYPE RECORDS, COLUMNS 17-20 AND COLUMN 42 MUST BE BLANK. ENTRY OF BLANKS IS ASSUMED.
- † NOTE119      FILENAME HAS BEEN PREVIOUSLY REFERENCED ON INPUT SPECIFICATIONS. ALL SPECIFICATIONS FOR THIS RECORD ARE NOT PROCESSED.
- NOTE120      CONTROL LEVEL IS SPECIFIED BUT COLUMN 59 IS NOT L. ENTRY OF L IS ASSUMED.
- NOTE121      CONTROL LEVEL IS SPECIFIED BUT COLUMN 60 IS NOT L-9. ENTRY OF 1 IS ASSUMED.
- NOTE122      MATCHING OR CHAINING FIELD IS SPECIFIED BUT COLUMN 62 IS NOT 1-9. ENTRY OF 1 IS ASSUMED.
- NOTE123      MATCHING OR CHAINING FIELD IS SPECIFIED BUT COLUMN 61 IS NOT M OR C. ENTRY OF M IS ASSUMED.

**Message-Meaning/Corrective Action**

- NOTE124 'PACKED' (COLUMN 43) IS NOT BLANK, BUT AN ALPHANUMERIC FIELD IS SPECIFIED. NUMERIC FIELD IS ASSUMED.
- NOTE125 SEQUENCE CHARACTERS (COLUMNS 15-16) ARE NOT BOTH ALPHABETIC OR BOTH NUMERIC. ENTRY OF AA IS ASSUMED.
- † NOTE126 'POSITION' (COLUMNS 21-24, 28-31, OR 35-38) IS INVALID OR NOT RIGHT-JUSTIFIED. ENTRY OF 0001 IS ASSUMED.
- NOTE127 FILE NAME (COLUMNS 7-14) IS INVALID OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE128 'START' (COLUMNS 44-47) OR 'END' (COLUMNS 48-51) IS INVALID OR NOT RIGHT-JUSTIFIED. ENTRY OF 0001 IS ASSUMED.
- † NOTE129 PACKED NUMERIC FIELD IS MORE THAN 8 BYTES LONG. LENGTH OF 8 IS ASSUMED.
- † NOTE130 FIELD NAME (COLUMNS 53-58, INPUT OR COLUMNS 32-37, OUTPUT) IS INVALID OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- NOTE131 FILENAME EXCEEDS 7 CHARACTERS. 7 CHARACTERS ARE USED.
- NOTE132 NO INPUT AND/OR OUTPUT SPECIFICATIONS FOUND FOR THIS FILE.
- † NOTE133 UNDEFINED TABLE SPECIFIED IN LOKUP OPERATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE134 DECIMAL POSITION IS INVALID. ENTRY OF ZERO IS ASSUMED FOR NUMERIC FIELD. ENTRY OF BLANK IS ASSUMED FOR ALPHANUMERIC FIELD.
- † NOTE135 FIELD LENGTH IS IMPROPERLY SPECIFIED OR IS NOT SPECIFIED. ENTRY OF ZERO IS ASSUMED FOR INVALID CHARACTER. WHEN REQUIRED LENGTH IS NOT SPECIFIED, ENTRY OF 4 IS ASSUMED.
- † NOTE136 OPERATION CODE (COLUMNS 28-32) IS INVALID OR MISSING. SPECIFICATION IS NOT PROCESSED.
- † NOTE137 ENTRY IN FACTOR 1 (COLS 18-27) OR FACTOR 2 (COLS 33-42) IS INVALID. SPECIFICATION IS NOT PROCESSED.
- † NOTE138 WARNING - ENTRY IN FACTOR 1 (COLS 18-27) OR FACTOR 2 (COLS 33-42) IS INVALID.

**Message-Meaning/Corrective Action**

- † NOTE139 FACTOR1, FACTOR2, OR RESULT FIELD IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
- † NOTE140 FORM TYPE (COLUMN 6) IS INVALID. SPECIFICATION IS NOT PROCESSED.
- NOTE141 'NOT' (COLUMNS 9, 12, OR 15) IS NOT N OR BLANK. ENTRY OF N IS ASSUMED.
- † NOTE142 CONTROL LEVEL IS IMPROPERLY SPECIFIED. ENTRY OF LO IS ASSUMED.
- † NOTE143 RESULTING INDICATOR IS INVALID, INDICATOR IS NOT PROCESSED.
- NOTE144 'HALF ADJUST' ENTRY (COLUMN 53) IS INVALID. ENTRY OF H IS ASSUMED.
- † NOTE145 FIELD NAME IS IMPROPERLY USED. SPECIFICATION IS NOT PROCESSED.
- NOTE146 INDICATOR IN COLS. 9-17 IS INVALID. BLANK ASSUMED.
- † NOTE147 REQUIRED RESULTING INDICATOR (COLUMNS 54-55, 56-57, OR 58-59) IS NOT SPECIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE148 'MVR' DOES NOT FOLLOW 'DIV', OR FOLLOWS A 'DIV' WITH HALF ADJUST SPECIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE149 UNPAIRED BEGSR-ENDSR MAY CAUSE PROGRAM ERROR OR OPERATION CODE IS NOT SUPPORTED BY THIS VERSION OF RPG.
- † NOTE150 DETAIL CALCULATION SPECIFICATION FOLLOWS A TOTAL CALCULATION SPECIFICATION. DETAIL SPECIFICATION IS NOT PROCESSED.
- † NOTE151 RESULT FIELD LENGTH (COLUMNS 49-51) IS GREATER THAN ALLOWED. A LENGTH OF 256 IS ASSUMED FOR AN ALPHANUMERIC FIELD. A LENGTH OF 15 IS ASSUMED FOR A NUMERIC FIELD.
- † NOTE152 OPERATION CODE OF TESTZ IS INVALID FOR THE 9300 MODE OF COMPUTATION. SPECIFICATION IS NOT PROCESSED.
- NOTE153 RECORD ADDRESS TYPE IS INCORRECT FOR A TAG FILE. "I" SPECIFICATION IS ASSUMED.

**Message-Meaning/Corrective Action**

- † NOTE154 FIELD NAME (COLUMNS 32-37) SPECIFIED ON OUTPUT SPECIFICATION IS NOT VALID, I.E., FIELD WAS DEFINED AS LABEL, KEYCV, ETC. SPECIFICATION IS NOT PROCESSED.
- † NOTE155 FILENAME (COLUMNS 7-14) IS MISSING, OR RECORD TYPE (COLUMN 15) IS IN WRONG ORDER, SPECIFICATION IS NOT PROCESSED.
- † NOTE156 CORRESPONDING FILENAME (COLUMNS 7-14) CANNOT BE DETERMINED. SPECIFICATION IS NOT PROCESSED.
- NOTE157 'STACKER SELECT' (COLUMN 16) IS INVALID, ENTRY OF BLANK IS ASSUMED.
- NOTE158 'SPACE BEFORE' (COLUMN 17) IS INVALID, ENTRY OF 1 IS ASSUMED.
- NOTE159 'SPACE AFTER' (COLUMN 18) IS INVALID. ENTRY OF 1 IS ASSUMED.
- NOTE160 'SKIP BEFORE' (COLUMNS 19-20) IS INVALID. ENTRY OF 01 IS ASSUMED.
- NOTE161 'SKIP AFTER' (COLUMNS 21-22) IS INVALID. ENTRY OF 01 IS ASSUMED.
- † NOTE162 RECORD TYPE (COLUMN 15) IS NOT AN H, D, T OR E RECORD. ALL SPECIFICATIONS FOR THIS ARE NOT PROCESSED.
- NOTE163 COLUMNS 17-22 MUST BE BLANK FOR 'AND' TYPE SPECIFICATIONS. ENTRY OF BLANK IS ASSUMED.
- NOTE164 COLUMNS 7-13 MUST BE BLANK FOR 'AND' OR 'OR' TYPE SPECIFICATIONS. ENTRY OF BLANK IS ASSUMED.
- † NOTE165 CORRESPONDING RECORD SPECIFICATION IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
- † NOTE166 EDIT CODE (COLUMN 38) ENTRY IS INVALID. ENTRY OF BLANK IS ASSUMED.
- NOTE167 'PACKED FIELD' (COLUMN 44) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- † NOTE168 FIELD NAME (COLUMNS 32-37) IS NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE169 'END POSITION' (COLUMNS 40-43) IS INVALID OR MISSING. SPECIFICATION IS NOT PROCESSED.

**Message-Meaning/Corrective Action**

- † NOTE170 LEADING OR CLOSING APOSTROPHE (') IN EDIT WORD IS NOT CORRECT. ENTRY OF BLANKS IN COLUMNS 45-70 IS ASSUMED.
- NOTE171 'BLANK AFTER' (COLUMN 39) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- † NOTE172 PUNCH AND PRINT FUNCTIONS ARE SPECIFIED FOR THE SAME FILE. ENTRY OF BLANKS IS ASSUMED FOR COLUMNS 17-22.
- NOTE173 EDIT CODES (COLUMN 38) MAY NOT BE SPECIFIED WITH LITERALS OR EDIT WORDS. ENTRY OF BLANK IS ASSUMED.
- † NOTE174 FIELD NAME (COLUMNS 32-37) IS UNDEFINED. SPECIFICATION IS NOT PROCESSED.
- NOTE175 WARNING - 'BLANK AFTER' (COLUMN 39) IS SPECIFIED FOR CONSTANT. ALL IDENTICAL CONSTANTS WILL BE BLANKED.
- NOTE176 WARNING: LITERAL/EDIT WORD IS NOT LEFT-JUSTIFIED.
- † NOTE177 EDIT WORD (COLUMNS 45-70) IS NOT LEFT-JUSTIFIED. ENTRY OF BLANKS IN COLUMNS 45-70 IS ASSUMED.
- NOTE178 'PACKED FIELD' (COLUMN 44) MAY NOT BE SPECIFIED WITH CONSTANT OR EDIT WORD. ENTRY OF BLANK IN COLUMN 44 IS ASSUMED.
- † NOTE179 FILENAME (COLUMNS 7-14) IS NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
- † NOTE180 EDIT WORD (COLUMNS 45-70) CONTAINS NO DIGIT POSITIONS OR MORE THAN FIFTEEN. ENTRY OF BLANKS IN COLUMNS 45-70 IS ASSUMED.
- † NOTE181 LEADING/CLOSING APOSTROPHES FOR LITERAL/EDIT WORD ARE MISSING. SPECIFICATION IS NOT PROCESSED.
- † NOTE182 'AND' OR 'OR' FOLLOWING A FIELD NAME SPECIFICATION OR AS FIRST OUTPUT SPECIFICATION IS INVALID. SPECIFICATION IS NOT PROCESSED.
- NOTE183 WARNING NO FIELDS DESCRIBED FOR THIS OR PREVIOUS RECORD.
- NOTE184 WARNING OUTPUT FIELDS OVERLAP.
- NOTE185 OUTPUT INDICATOR (COLUMNS 24-25, 27-28, OR 30-31) IS INVALID OR UNDEFINED. ENTRY OF LO IS ASSUMED.

**Message-Meaning/Corrective Action**

- NOTE186      OUTPUT INDICATORS SHOULD START IN COLUMNS 23-25;  
THEN 26-28, AND FINALLY 29-31. ENTRY IS SHIFTED LEFT.
- NOTE187      'NOT' (COLUMNS 23, 26 OR 29) IS NOT A BLANK OR N. ENTRY  
OF N IS ASSUMED.
- NOTE188      WARNING-OVERFLOW INDICATOR IS SPECIFIED IN 'AND' TYPE  
SPECIFICATION. RECORD WILL NOT BE PUT OUT AS OVERFLOW  
LINE.
- NOTE189      DECIMAL POSITIONS MUST BE ZERO FOR PAGE(N) FIELD.  
ENTRY OF ZERO IS ASSUMED.
- † NOTE190      SPECIFICATION TYPE CANNOT BE DETERMINED. RECORD AND  
FIELD DEFINITION ARE SPECIFIED IN SAME LINE OR BOTH ARE  
BLANK. SPECIFICATION IS NOT PROCESSED.
- † NOTE191      FORM TYPE (COLUMN 6) IS INVALID (NOT 0) SPECIFICATION IS  
NOT PROCESSED.
- † NOTE192      NO OUTPUT INDICATOR (COLUMNS 24-25, 27-28, OR 30-31) IS  
SPECIFIED FOR 'AND' OR 'OR' TYPE SPECIFICATION.  
SPECIFICATION IS NOT PROCESSED.
- † NOTE193      HEXADECIMAL CONSTANT (COLUMNS 45-70) CONTAINS  
INVALID CHARACTER. SPECIFICATION IS NOT PROCESSED.
- † NOTE194      LEADING OR CLOSING APOSTROPHE (') IN HEXADECIMAL  
CONSTANT IS NOT CORRECT. SPECIFICATION IS NOT  
PROCESSED.
- NOTE195      WARNING: PAGE 1 NOT SUPPORTED AS AN AUTOMATIC PAGE  
COUNTER IF IN 9200/9300 MODE OF PROCESSING.
- NOTE196      WARNING-PAGE2,...,PAGE7 ARE NOT SUPPORTED AS  
AUTOMATIC PAGE COUNTERS IN 9200/9300 AND MOD 20  
MODE OF PROCESSING.
- NOTE197      DECIMAL POSITIONS MUST BE BLANK FOR DATE FIELD. ENTRY  
OF BLANK IS ASSUMED.
- † NOTE198      FIELD NAME (COLUMNS 32-37) IS INVALID. SPECIFICATION IS  
NOT PROCESSED.
- † NOTE199      FILE NAME (COLUMNS 7-14) IS INVALID. SPECIFICATION IS  
NOT PROCESSED.
- NOTE200      WARNING: TABLE FILE MUST BE SUBMITTED AT EXECUTION.



**Message-Meaning/Corrective Action**

- NOTE201      RESULTING INDICATOR IS INVALID OR UNDEFINED. ENTRY OF  
LO IS ASSUMED.
- NOTE202      WARNING - RESULTING INDICATOR IS UNREFERENCED.
- NOTE203      FIELD NAME IS UNDEFINED. FIELD IS PROCESSED WITH  
ASSUMED LENGTH OF 004.
- NOTE204      WARNING - FIELD NAME IS MULTI-DEFINED.
- NOTE205      WARNING - FIELD NAME IS UNREFERENCED.
- \* NOTE206      THE COMBINED LENGTHS OF LITERALS AND FIELD NAMES  
EXCEED ALLOCATED MAIN STORAGE.
- NOTE207      FACTOR 1 MUST HAVE SAME UNPACKED LENGTH AS KEY  
FIELD OF FILE SPECIFIED IN FACTOR 2. SPECIFICATION NOT  
PROCESSED.
- NOTE208      WARNING: THE DATA FORMAT FOR MATCHING FIELD IS NOT  
CONSISTENT.
- \* NOTE209      THE SUM OF THE LENGTHS OF THE MATCHING FIELDS FOR THE  
PRIMARY FILE DOES NOT EQUAL THAT OF EACH SECONDARY  
FILE. EXECUTION IS DELETED.
- \* NOTE210      THE SUM OF THE LENGTHS OF THE MATCHING FIELDS IS NOT  
CONSTANT IN EACH RECORD WHICH SPECIFIED MATCHING  
FIELDS FOR A FILE. EXECUTION IS DELETED.
- NOTE211      WARNING - THE SUM OF THE LENGTHS OF THE CONTROL  
FIELDS IS NOT CONSTANT IN EACH RECORD WHICH SPECIFIED  
CONTROL FIELDS.
- \* NOTE212      AN EXCESSIVE AMOUNT OF OBJECT CODE HAS BEEN  
GENERATED FOR A SINGLE RECORD GROUP. EXECUTION IS  
DELETED.
- † NOTE213      THIS SPECIFICATION CONTAINS AN ASTERISK CONDITION  
INDICATION AND FOLLOWS A SPECIFICATION THAT HAS BEEN  
DELETED. SPECIFICATION IS NOT PROCESSED.
- † NOTE214      UNDEFINED OVERFLOW INDICATOR. ENTRY IGNORED.
- † NOTE215      WARNING: FIELD LENGTHS ARE INVALID FOR THIS  
OPERATION.

**Message-Meaning/Corrective Action**

- NOTE216 PLUS AND/OR MINUS RESULTING INDICATORS (COLUMNS 54-55 OR 56-57) ARE NOT ALLOWED FOR TESTING ALPHANUMERIC FIELDS. INDICATORS ARE IGNORED.
- † NOTE217 FIELD TYPE IS INVALID FOR THIS OPERATION. SPECIFICATION IS NOT PROCESSED.
- NOTE218 ENTRY IN COLUMNS 16-18 VALID FOR INDEXED-SEQUENTIAL ADD ONLY. ENTRY IS IGNORED.
- NOTE219 REQUIRED 'ADD' ENTRY (COLUMNS 16-18) IS MISSING. 'ADD' IS ASSUMED.
- † NOTE220 FILE SPECIFIED ON OUTPUT FORMAT SPECIFICATION IS UNDEFINED OR NOT AN OUTPUT FILE (U, C, O OR INDEX SEQUENTIAL WITH ADDED RECORDS). SPECIFICATION IS NOT PROCESSED.
- NOTE221 WARNING - FILENAME (COLUMNS 7-14) IS NOT REFERENCED ON OUTPUT SPECIFICATIONS.
- \* NOTE222 NO VALID OUTPUT SPECIFICATIONS ARE PRESENT. EXECUTION IS DELETED.
- NOTE223 ALL OUTPUT LINES OF A PRINTER FILE MUST INDICATE EITHER SPACING AND/OR SKIPPING. SINGLE SPACING IS ASSUMED FOR ALL OUTPUT LINES OF NAMED FILE WHICH HAVE NO PRINT FUNCTION.
- NOTE224 STACKER SELECT MAY NOT BE SPECIFIED WITH PRINT FILE. STACKER SELECT IS IGNORED AND SINGLE SPACING IS ASSUMED FOR ALL LINES OR NAMED FILE.
- NOTE225 PRINT OR PUNCH FUNCTION MAY NOT BE SPECIFIED FOR AN OUTPUT RECORD OF TAPE OR DISC FILE. STACKER SELECT, SPACING, OR SKIPPING IS IGNORED ON ALL RECORDS OF NAMED FILE.
- NOTE226 PRINT FUNCTION MAY NOT BE SPECIFIED FOR OUTPUT RECORD OF PUNCH FILE. SPACE AND SKIP ENTRIES ARE IGNORED FOR ALL RECORDS OF NAMED FILE.
- \* NOTE227 NUMBER OF LINES OF OUTPUT EXCEEDS THE CAPACITY OF RPG. MAXIMUM NUMBER IS 1023. EXECUTION IS DELETED.
- NOTE228 IMPROPER USE OF PACKING OR ZERO SUPPRESSION ON ALPHANUMERIC OR PACKED FIELD. ENTRY OF BLANK IS ASSUMED FOR INVALID CODE.

**Message-Meaning/Corrective Action**

- NOTE229      END POSITION SPECIFIED FOR THE FIELD IS GREATER THAN THE RECORD LENGTH. ALL OR PART OF THE FIELD IS LOST, STARTING WITH THE RIGHTMOST POSITION.
- † NOTE230      END POSITION IS LESS THAN THE FIELD LENGTH. FIELD IS NOT PROCESSED.
- NOTE231      WARNING: FIELD TO BE EDITED IS TOO SMALL OR TOO LARGE FOR EDIT WORD.
- NOTE232      FIELD TO BE EDITED IS NOT NUMERIC. NO EDITING IS PERFORMED.
- NOTE233      CONTROL LEVEL, MATCHING FIELD, OR CHAINING FIELD ENTRY SPECIFIED FOR BINARY FIELD.
- NOTE234      INCORRECT LENGTH SPECIFIED FOR BINARY FIELD. DEFAULT IS A LENGTH OF 4.
- † NOTE235      RESULT FIELD FOR TIME OPERATION MUST BE 6 OR 12 DIGIT NUMERIC WITH NO DECIMAL POSITIONS. SPECIFICATION IS NOT PROCESSED.
- NOTE236      \*PRINT IS NOT SUPPORTED. SPECIFICATION IS NOT PROCESSED.
- NOTE237      \*IN COL 40 OF OUTPUT SPECIFICATION IS NOT SUPPORTED. ENTRY IS IGNORED.
- \* NOTE238      MAXIMUM NUMBER OF EXTERNAL SYMBOLS (EXIT SUBROUTINES, ULABL ENTRIES, AND LABEL EXITS) HAS BEEN EXCEEDED. EXECUTION IS DELETED.
- † NOTE239      INVALID USE OF ASTERISK INDICATOR. ASTERISK IS IGNORED.
- † NOTE240      ASTERISK INDICATOR IS INVALID IN 'OR' OR 'AND' TYPE SPECIFICATIONS. SPECIFICATION IS NOT PROCESSED.
- NOTE241      DEVICE (COLUMNS 40-46) IS INVALID FOR DISPLAY FILES. 'CONSOLE' IS ASSUMED.
- NOTE242      ONLY ONE DISPLAY FILE MAY BE SPECIFIED PER PROGRAM. SPECIFICATION IS NOT PROCESSED.
- NOTE243      FILE ORGANIZATION (COLUMN 32) IS INVALID. 'I' IS ASSUMED FOR INDEXED SEQUENTIAL FILES AND BLANK IS ASSUMED OTHERWISE.

**Message-Meaning/Corrective Action**

- NOTE244 RECORD ADDRESS TYPE (COLUMN 31) IS INVALID. 'A' IS ASSUMED FOR INDEXED SEQUENTIAL FILES, 'R' FOR DIRECT FILES AND BLANK OTHERWISE.
- NOTE245 KEY LENGTH (COLUMNS 29-30) MUST BE 3 OR GREATER. ENTRY OF 3 IS ASSUMED.
- † NOTE246 FILE CONDITIONING INDICATOR (COLUMNS 71-72) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- † NOTE247 FILE CONDITIONING INDICATOR (COLUMNS 71-72) IS NOT ALLOWED. BLANK IS ASSUMED.
- NOTE248 FOR SPECIAL FILES COLUMNS 28-32, 35-38, 53, 60-70 MUST BE BLANK. BLANKS ARE ASSUMED.
- NOTE249 WARNING: BLOCK LENGTH (COLUMNS 20-23) FOR INDEXED SEQUENTIAL FILE IS MISSING. A BLOCK LENGTH EQUAL TO RECORD LENGTH IS ASSUMED.
- NOTE250 BLOCK LENGTH (COLUMNS 20-23) IS SMALLER THAN MINIMUM FOR DEVICE, OR LESS THAN 80 IF USER LABELS ARE SPECIFIED. IF USER LABELS ARE SPECIFIED, 80 IS ASSUMED; OTHERWISE, MINIMUM FOR DEVICE IS ASSUMED.
- † NOTE251 RECORD LENGTH (COLUMNS 24-27) IS INVALID OR MISSING. SPECIFICATION IS NOT PROCESSED.
- † NOTE252 FROM FILENAME (COLUMNS 11-18) IS NOT DEFINED AS A TABLE FILE ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- † NOTE253 FROM FILENAME (COLUMNS 11-18) IS MULTIPLY REFERENCED BUT A CARD DEVICE IS NOT SPECIFIED ON THE FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
- NOTE254 TO FILENAME (COLUMNS 19-26) IS MULTIPLY REFERENCED BUT A PRINTER DEVICE IS NOT SPECIFIED ON THE FILE DESCRIPTION SPECIFICATION. ENTRY OF BLANKS IS ASSUMED.
- NOTE255 WARNING - FILE SPECIFIED AS RECORD ADDRESS OR TAG DATA FILE ON FILE DESCRIPTION SPECIFICATION IS MISSING ON THE EXTENSION SPECIFICATION.
- \* NOTE256 INSUFFICIENT MAIN MEMORY FOR COMPILATION. COMPILATION ABORTED.

**Message-Meaning/Corrective Action**

- † NOTE257      MULTIPLE DEFINITION OF AN EXECUTION TIME ARRAY.
- NOTE258      DATA FORMAT (COL 43) PACKED OR BINARY INVALID FOR DEVICE.
- NOTE259      FOR LOOK-A-HEAD RECORD, RECORD SEQUENCE (COLUMNS 15-16) MUST BE ALPHA. ENTRY OF AA IS ASSUMED.
- NOTE260      FOR LOOK-A-HEAD RECORD, NUMBER, OPTION AND RECORD IDENTIFICATION (COLUMNS 17, 18, COLUMNS 21-41) MUST BE BLANK. ENTRY OF BLANKS IS ASSUMED.
- † NOTE261      LOOK-A-HEAD RECORD USED WITH FILE NOT DEFINED AS PRIMARY OR SECONDARY. ALL SPECIFICATIONS FOR THIS RECORD ARE NOT PROCESSED.
- NOTE262      WARNING: LOOK-A-HEAD SHOULD BE THE LAST RECORD SPECIFICATION FOR THIS FILE.
- NOTE263      MATCHING RECORD INDICATOR (COLUMNS 61-62) IS SPECIFIED FOR FILE NOT DEFINED AS PRIMARY/SECONDARY FILE. ENTRY OF BLANKS IS ASSUMED.
- NOTE264      CHAINING INDICATOR (COLUMNS 61-62) IS SPECIFIED FOR INVALID CHAINING TYPE FILE. ENTRY OF BLANKS IS ASSUMED.
- † NOTE265      SUBSCRIPTED FIELD IS NOT DEFINED AS AN ARRAY. SPECIFICATION IS NOT PROCESSED.
- † NOTE266      FOR EXECUTION TIME ARRAY INPUT, NUMBER OF ELEMENTS SPECIFIED FOR INPUT EXCEEDS SIZE OF ARRAY. SIZE OF ARRAY IS ASSUMED.
- † NOTE267      INVALID VARIABLE SUBSCRIPT, I.E. VARIABLE IS NOT A NUMERIC FIELD.
- † NOTE268      INVALID LITERAL SUBSCRIPT, I.E. SUBSCRIPT IS NEGATIVE/ZERO (VALUE OF 1 IS ASSUMED) OR SUBSCRIPT IS OUT OF RANGE (HIGHEST VALUE IS ASSUMED).
- † NOTE269      FIELD SPECIFICATION IS OUT OF SEQUENCE, I.E. DOES NOT FOLLOW A RECORD INPUT SPECIFICATION. ALL SPECIFICATIONS UNTIL NEXT FILE RECORD ARE NOT PROCESSED.
- NOTE270      WARNING: NO RECORD IDENTIFICATION CODES SPECIFIED ON THE RECORD PRECEDING A 'AND' RECORD.

**Message-Meaning/Corrective Action**

- NOTE271 FOR 'AND' RECORD, STACKER SELECT MUST BE BLANK. ENTRY OF BLANK IS ASSUMED.
- NOTE272 WARNING - LITERAL/EDIT WORD (COLUMNS 45-70) IS NOT LEFT-JUSTIFIED.
- † NOTE273 FOR ARRAY INPUT, LENGTH OF FIELD LOCATION IS NOT A MULTIPLE OF THE ELEMENT LENGTH. NEXT LOWEST MULTIPLE ASSUMED.
- NOTE274 CONTROL FIELD INVALID WITH CHAINED OR DEMAND FILES. SPECIFICATION IS NOT PROCESSED.
- NOTE275 ADDED RECORD SPECIFIED ON OUTPUT FOR NON-ISAM FILE OR COLUMN 66 OF CORRESPONDING ISAM FILE DESCRIPTION IS NOT 'A'.
- NOTE276 PROGRAM ID. (COLS 75-80) BLANK OR INVALID. 'RPGOBJ' ASSUMED.
- NOTE277 FIELD EDITED WITH Y EDIT CODE REQUIRES 3-6 CHARACTER POSITIONS.
- † NOTE278 TRANSLATE RECORDS NOT BEGINNING WITH \*\* BLANK RECORD. FILE TRANSLATION PROCESSING TERMINATED.
- † NOTE279 FILE TRANSLATION SPECIFIED BUT NO TRANSLATE RECORDS PRESENT OR NOT IN PROPER SEQUENCE. FILE TRANSLATION PROCESSING TERMINATED.
- † NOTE280 INVALID TRANSLATION SPECIFICATION. WHEN USING \*FILES OR \*EQUATE ONLY ONE TRANSLATE TABLE CAN BE SPECIFIED. FILE TRANSLATION PROCESSING TERMINATED.
- † NOTE281 INVALID HEX CHARACTER SPECIFIED FOR TRANSLATION FILE TRANSLATION PROCESSING TERMINATED.
- † NOTE282 INVALID FACTOR SPECIFICATION. FACTOR 2 OR RESULT MUST BE ARRAY BUT MAY NOT REF. SAME ARRAY.
- † NOTE283 ALPHANUMERIC FACTOR 1 IS INVALID FOR CHAINING TO A DIRECT FILE, OR IN COMBINATION WITH A NUMERIC KEY FIELD ON THE FACTOR 2 FILE FOR CHAIN AND SETLL OPERATIONS. SPECIFICATION IS NOT PROCESSED.
- NOTE284 OVERFLOW INDICATOR INVALID FOR EXCEPTION OUTPUT LINE.
- NOTE285 POSITIONS 60-65 CONTAIN AN INVALID ENTRY. ASSUME BLANK.
- NOTE286 INVALID FILE TRANSLATION SPECIFICATION. CHECK VALIDITY OF TRANSLATE RECORDS.

**Message-Meaning/Corrective Action**

- † NOTE287      INVALID ALTSEQ RECORD. STANDARD COLLATING SEQUENCE  
SELECTED.
- † NOTE288      INVALID HEX CHARACTER IN ALTSEQ RECORD. STANDARD  
COLLATING SEQUENCE SELECTED.
- † NOTE289      ALTSEQ FILE NOT PRESENT. STANDARD COLLATING  
SEQUENCE SELECTED.
- NOTE290      DECIMAL POSITIONS SPECIFIED FOR BLANK FIELD LENGTH.  
DECIMAL POSITIONS IGNORED.
- † NOTE291      INVALID ALTSEQ CODE. STANDARD COLLATING SEQUENCE  
SELECTED.
- † NOTE292      SPECIFIED FILE NAME FOR FILE TRANSLATION NOT FOUND.  
FILE TRANSLATION PROCESSING TERMINATED.
- † NOTE293      FILE TRANSLATION SPECIFIED FOR DISPLAY FILE. FILE  
TRANSLATION PROCESSING TERMINATED.
- NOTE294      INVALID FILL CHARACTER SPECIFIED FOR EDIT CODE.
- NOTE295      DECIMAL POSITION ENTRY (COL 52) DOES NOT AGREE WITH  
NUMBER OF DECIMAL POSITIONS PREVIOUSLY SPECIFIED.  
PREVIOUS SPECIFICATION IS USED.
- † NOTE296      FIELD WAS PREVIOUSLY DEFINED WITH DIFFERENT LENGTH.  
SPECIFICATION IS NOT PROCESSED.
- NOTE297      WARNING - EXCPT OPERATION OR EXCEPTION OUTPUT  
SPECIFICATIONS MISSING.
- NOTE298      MATCHING FIELD INDICATOR M1-M9 MUST BE REFERENCED  
AT LEAST TWICE.
- NOTE299      RECORD LENGTH (COL 24-27) IS LESS THAN MINIMUM FOR  
DEVICE SPECIFIED, MINIMUM FOR DEVICE IS ASSUMED.
- NOTE300      FILE FORMAT (COLUMN 19) IS NOT F. ENTRY OF F IS ASSUMED  
FOR REMOTE FILES OR FILES USING S/3 MODE.
- † NOTE301      STATION TYPE (COLUMN 16) DOES NOT CONTAIN T OR R.  
SPECIFICATION NOT PROCESSED.
- NOTE302      COLUMN(S) 17-18, 21-47, 52, 61-64, NOT BLANK. ENTRY  
IGNORED.
- † NOTE303      REMOTE TERMINAL (COLUMNS 48-51) IS INVALID.  
SPECIFICATION NOT PROCESSED.

**Message-Meaning/Corrective Action**

- † NOTE304 PERMANENT ERROR INDICATOR (COLUMNS 53-54) IS INVALID. SPECIFICATION NOT PROCESSED.
- † NOTE305 WAIT TIME (COLUMNS 55-57) IS INVALID. 180 SECONDS IS ASSUMED.
- NOTE306 RECORD AVAILABLE INDICATOR (COLUMNS 58-59) IS INVALID. SPECIFICATION NOT PROCESSED.
- † NOTE307 LAST FILE PROCESSED (COLUMN 60) IS NOT L OR BLANK. SPECIFICATION NOT PROCESSED.
- NOTE308 WAIT TIME (COLUMNS 55-57) IS INVALID. ENTRY OF BLANK IS ASSUMED.
- NOTE309 RECORD AVAILABLE INDICATOR (COLUMNS 58-59) IS PRESENT ON TRANSMIT FILE, OR FOR PROGRAM WITH SINGLE REMOTE FILE. ENTRY OF BLANK IS ASSUMED.
- NOTE310 LAST FILE PROCESSED (COLUMN 60) IS NOT BLANK ON TRANSMIT FILE OR PRIMARY INPUT FILE. ENTRY OF BLANK IS ASSUMED.
- † NOTE311 CORRESPONDING FILE DESCRIPTION SPECIFICATION FILE IS NOT A REMOTE FILE. SPECIFICATION NOT PROCESSED.
- † NOTE312 A CONVERSATIONAL FILE WAS DEFINED WHEN NOT ALLOWED. SPECIFICATION NOT PROCESSED.
- † NOTE313 NO CORRESPONDING OUTPUT OR COMBINED FILE SPECIFICATION FOR TRANSMIT FILE. SPECIFICATION NOT PROCESSED.
- † NOTE314 NO CORRESPONDING INPUT OR COMBINED FILE SPECIFICATION FOR RECEIVE FILE. SPECIFICATION NOT PROCESSED.
- NOTE315 BLOCKED RECORD SPECIFIED FOR CONVERSATIONAL FILE. NO BLOCKING IS ASSUMED.
- † NOTE316 TELECOMMUNICATIONS SPECIFICATION MISSING FOR FILE DEFINED AS A REMOTE FILE. SPECIFICATION NOT PROCESSED.
- † NOTE317 LOOK AHEAD FIELDS SPECIFIED FOR REMOTE FILE. SPECIFICATION NOT PROCESSED.
- † NOTE318 MATCHING FIELDS DEFINED ON A TRANSMIT FILE WITH CONVERSATIONAL REPLY. SPECIFICATION NOT PROCESSED.



**Message-Meaning/Corrective Action**

- † NOTE319      MATCHING FIELDS DEFINED FOR A FILE DESIGNED AS LAST FILE. SPECIFICATION NOT PROCESSED.
- NOTE320      FOR A TRANSMIT THEN RECEIVE REMOTE PROGRAM, IF END-OF-FILE IS SPECIFIED FOR ANY INPUT FILE, E IS ASSUMED IN COLUMN 17 OF THE REMOTE INPUT FILE.
- † NOTE321      A TRANSMIT WITH CONVERSATIONAL REPLY FILE IS USED WITH FORCE OR READ OF CODE OR AS A PRIMARY FILE. SPECIFICATION NOT PROCESSED.
- † NOTE322      FILE NAME USED TWICE ON TELECOMMUNICATIONS SPECS. SPECIFICATION NOT PROCESSED.
- † NOTE323      DUAL I/O AREAS NOT PERMITTED FOR CONVERSATIONAL FILE. SPECIFICATION NOT PROCESSED.
- † NOTE324      INVALID DEVICE (COLUMNS 65-70) SPECIFIED. SPECIFICATION NOT PROCESSED.
- † NOTE325      FILENAME (COLUMNS 7-14) INVALID OR UNDEFINED. SPECIFICATION NOT PROCESSED.
- † NOTE326      PACKED FIELD OR BINARY FIELD SPECIFIED IN A FILE WITHOUT TRANSPARENCY. SPECIFICATION NOT PROCESSED.
- † NOTE327      TERMINAL NAME (COLUMNS 71-74) INVALID. SPECIFICATION NOT PROCESSED.
- NOTE328      TRANSPARENCY (COLUMN 19) IS NOT Y, N, OR BLANK. BLANK IS ASSUMED.
- NOTE329      CONFIGURATION ENTRY (COLUMN 15) OF P, S OR M NOT SUPPORTED. ENTRY IGNORED.
- NOTE330      COLUMNS 16-52, 55-70 MUST BE BLANK FOR MULTIPLE TERMINAL FILES AFTER FIRST SPECIFICATION. ENTRIES IGNORED.
- † NOTE331      TERMINAL NAME (COLUMNS 71-74) USED TWICE. SPECIFICATION NOT PROCESSED.
- † NOTE332      REMOTE AUXILIARY DEVICE (COLUMNS 65-70) NOT ALLOWED FOR MULTIPLE TERMINAL FILES. SPECIFICATION NOT PROCESSED.
- NOTE333      CCA NAME (COLUMNS 70-73) ON CONTROL CARD INVALID OR MISSING FOR REMOTE FILES. 'RPGI' ASSUMED.

**Message-Meaning/Corrective Action**

- NOTE334 SWITCHED ENTRY (COLUMN 20) IS NOT E, S, M, A, B, OR BLANK. BLANK IS ASSUMED.
- † NOTE335 REMOTE TERMINAL (COLUMNS 48-51) INVALID OR TERMINAL COMBINATION INVALID FOR MULTIPLE TERMINAL FILE. SPECIFICATION NOT PROCESSED.
- NOTE336 NUMBER OF EXTENTS (COL. 68-69) INVALID FOR FILE TYPE. ENTRY IGNORED.
- NOTE337 END POSITION SPECIFIED FOR \*PLACE GREATER THAN 256 OR LESS THAN TWICE PREVIOUS HIGH POSITION.
- NOTE338 FACTOR 1 ENTRY (COL. 18-27) INVALID FOR OPERATION. SPECIFICATION IS NOT PROCESSED.
- NOTE339 FACTOR 2 ENTRY (COL. 33-43) INVALID FOR OPERATION. SPECIFICATION IS NOT PROCESSED.
- NOTE340 INVALID SKIP ENTRIES IN COL. 19-22 OR GREATER THAN THE FORM LENGTH SPECIFIED, ASSUME BLANK.
- NOTE341 ALL THREE RESULTING INDICATORS ARE THE SAME.
- † NOTE342 MORE THAN 7 AN/OR LINES SPECIFIED.
- NOTE343 UNORDERED ENTRY (U IN COL. 66) IS INVALID. ENTRY OF 'A' IS ASSUMED.
- NOTE344 PROCESSING MODE (COL. 28) INVALID FOR CHAIN OUTPUT FILE. 'R' IS ASSUMED.
- NOTE345 FILE ORGANIZATION (COL. 32) NOT REQUIRED FOR CHAIN OUTPUT FILE.
- † NOTE346 TRAILER RECORD OVERLAPS HEADER RECORD. TR IGNORED.
- † NOTE347 NO TRAILER FIELDS FOR SPREAD CARD. TR IGNORED.
- † NOTE348 FILE TYPE INVALID FOR SPREAD CARD DESIGNATION. TR IGNORED.
- † NOTE349 LOOKAHEAD NOT ALLOWED WITH SPREAD CARDS. TR IGNORED.
- NOTE350 CONTINUATION OPTION (COL. 54-59) IS REPEATED FOR A FILE. SECOND ENTRY IS IGNORED.
- NOTE351 BUFOFF SPECIFIED FOR AN OUTPUT FILE. ENTRY IGNORED.

**Message-Meaning/Corrective Action**

- NOTE352 CONTINUATION OPTION (COL. 54-59) INVALID FOR DEVICE. CONTINUATION LINE IS IGNORED. ←
- † NOTE353 CONTINUATION OPTION (COL. 54-59) IS INVALID OR MISSING. CONTINUATION LINE IS IGNORED.
- † NOTE354 INVALID CONTINUATION LINE ENTRY (COL. 60-65). CONTINUATION LINE IS IGNORED.
- NOTE355 BUFOFF SPECIFIED FOR NON-ASCII TAPE FILE. ASCII FILE ASSUMED.
- † NOTE356 PREVIOUS SPECIFICATION INVALID, CONTINUATION RECORD IGNORED.
- † NOTE357 INVALID CONTINUATION LINE ENTRY (COL. 60-65) FOR ACCESS CONTINUATION OPTION. EXC IS ISSUED. ←
- † NOTE358 INVALID FORM LENGTH ENTRY (COL. 15-17). SPECIFICATION IS NOT PROCESSED.
- † NOTE359 INVALID OVERFLOW LINE ENTRY (COL. 20-22). SPECIFICATION IS NOT PROCESSED.
- NOTE360 OVERFLOW LINE EXCEEDS FORM LENGTH. FORM LENGTH ASSUMED FOR OVERFLOW LINE.
- NOTE361 INVALID ENTRY IN COL. 18-19. 'FL' ASSUMED.
- NOTE362 INVALID ENTRY IN COL. 23-27. 'OL' ASSUMED.
- † NOTE363 FILE DESCRIPTION AND CONTINUATION SPECIFIED ON SAME LINE, SPECIFICATION IS NOT PROCESSED.
- † NOTE364 INVALID FEATURE SPECIFIED FOR PROCESSING MODE.
- NOTE365 FETCH OVERFLOW ENTRY (COL. 16) INVALID FOR DEVICE.
- NOTE366 RESULT FIELD (COL. 43-48) IS INVALID FOR OPERATION. SPECIFICATION IS NOT PROCESSED.
- NOTE367 FETCH OVERFLOW AND OVERFLOW INDICATOR SPECIFIED FOR SAME OUTPUT LINE. INDICATOR IGNORED.
- NOTE368 NUMBER OF ENTRIES PER RECORD TIMES LENGTH OF ENTRY IS TOO LARGE. LENGTH OF ENTRY ASSUMED 1.
- NOTE369 OPERATOR CONTROL (COLUMN 9) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.

**Message-Meaning/Corrective Action**

- NOTE370 FIRST PAGE FORMS ALIGNMENT (COLUMN 41) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE371 COMBINED FILE (COLUMN 15) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE372 DISPLAY FILE (COLUMN 15) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE373 TABLE OR ARRAY FILE (COLUMN 16) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE374 BLOCK LENGTH (COLUMNS 20-23) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE375 ADDR/OUT/RECORD ADDRESS FILES (COLUMN 16) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- ↓  
NOTE376 DOUBLE BUFFERING (COLUMN 32) INVALID FOR IMS 90 ACTION PROGRAMS OR IRAM FILES. ONE I/O AREA ASSUMED.
- ↑  
NOTE377 SEQUENTIAL INPUT FILES INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE378 DEVICE (COLUMNS 40-46) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE379 LABELS/LABEL EXIT (COLUMNS 53-59) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRIES IGNORED.
- NOTE380 INDEX IN MAIN MEMORY (COLUMNS 60-65) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE381 CYLINDER OVERFLOW SPACE PERCENTAGE (COLUMN 67) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE382 NUMBER OF EXTENTS (COLUMNS 71-72) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE383 U1-U8 INDICATORS (COLUMNS 71-72) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE384 ERROR ANALYSIS DUMP (COLUMN 8) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE385 FILE LOAD INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.

**Message-Meaning/Corrective Action**

- NOTE386 FILE INDEX CONTINUATION OPTION (COLUMNS 54-59) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE387 TAPE REWIND (COLUMN 70) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE388 C1-C9 CHAINING (COLUMNS 9-10) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE389 SPREAD CARDS (COLUMNS 19-20) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE390 STACKER SELECT (COLUMN 42) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE391 STACKER SELECT (COLUMN 16) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY IGNORED.
- NOTE392 INVALID OPERATION (COLUMNS 28-32) FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE393 TELECOMMUNICATIONS SPECIFICATION (COLUMN 6) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE394 SIGN HANDLING (COLUMN 40) INVALID FOR IMS 90 ACTION PROGRAMS. ENTRY OF 'B' ASSUMED.
- NOTE395 UNORDERED LOAD (COLUMN 66) INVALID FOR IMS 90 ACTION PROGRAMS. SPECIFICATION NOT PROCESSED.
- NOTE396 REMOTE TERMINAL (COLUMNS 48-51) REQUIRES A COMBINED FILE. SPECIFICATION NOT PROCESSED.
- NOTE397 WARNING: RESULT FIELD MAY NOT BE LARGE ENOUGH FOR RESULT. RESULTING INDICATORS ARE NOT SET IF ARITHMETIC OVERFLOW OCCURS.
- NOTE398 COLUMNS 7-8 ARE NOT BLANK IN 9200/9300 OR IBM 360/20 MODE. BLANKS ASSUMED. ←
- NOTE399 RECORD SEQUENCE FOR CHAINING FILE ON FILE EXTENSION SPECIFICATION DOES NOT MATCH SEQUENCE ON INPUT SPECIFICATION. ENTRY ON INPUT SPECIFICATION IS ASSUMED.
- † NOTE400 REMOTE TERMINAL (COLUMNS 48-51) CANNOT BE A COMBINED FILE. SPECIFICATION NOT PROCESSED.
- † NOTE401 CDI NOT CONFIGURED. WORK STATION FILE NOT ALLOWED. SPECIFICATION NOT PROCESSED.

**Message-Meaning/Corrective Action**

- † NOTE402 ONLY ONE WORK STATION FILE ALLOWED. SPECIFICATION NOT PROCESSED.
- NOTE403 WORKSTATION FILE MUST BE A COMBINED FILE. COMBINED ASSUMED.
- † NOTE404 WORKSTATION FILE MUST BE PRIMARY OR DEMAND. SPECIFICATION NOT PROCESSED.
- † NOTE405 LOOK-AHEAD FIELDS NOT ALLOWED FOR WORKSTATION FILE, SPECIFICATION NOT PROCESSED.
- NOTE406 MATCHING OR CHAINING FIELDS NOT ALLOWED FOR WORKSTATION FILE. INDICATOR IGNORED.
- † NOTE407 FORMAT NAME SPECIFICATION FOR A WORKSTATION FILE IS INCORRECT. SPECIFICATION NOT PROCESSED.
- † NOTE408 SECONDARY FILES NOT ALLOWED WHEN WORK STATION FILE IS A PRIMARY FILE. SPECIFICATION NOT PROCESSED.
- † NOTE412 SPLIT MATCHING FIELDS NOT ALLOWED. EXECUTION IS DELETED.
- NOTE413 1P INDICATOR IS INVALID FOR TOTAL AND EXCEPTION OUTPUT RECORDS. RECORD WILL NOT BE WRITTEN WHEN PROGRAM IS EXECUTED.
- NOTE414 AN/OR LINE EXPECTED OR SPECIFIED IMPROPERLY. SPECIFICATION IS NOT PROCESSED.
- NOTE415 LR INDICATOR IS INVALID FOR HEADER AND DETAIL OUTPUT RECORDS. RECORD WILL NOT BE WRITTEN WHEN PROGRAM IS EXECUTED.
- NOTE416 WARNING: FIELD IS GREATER THAN DEFAULT VALUE FOR DEVICE. FIELD IS TRUNCATED.
- \* NOTE417 FACTOR 2 IN THE GOTO STATEMENT IS UNDEFINED.
- NOTE418 WARNING-CURRENCY SYMBOL (COLUMN 18) INVALID. CANNOT BE 0 \* , . - C R OR AMPERSAND. DOLLAR SIGN ASSUMED.
- † NOTE419 VALUE DETECTED IN COLS 21-24, 28-31, OR 35-38 EXCEEDS RECORD LENGTH. A VALUE OF 1 IS ASSUMED.
- † NOTE420 MORE THAN ONE CONSOLE FILE IN A PROGRAM IS INVALID. SPECIFICATION IS NOT PROCESSED.

**Message-Meaning/Corrective Action**

- † NOTE421 RECORD IDENTIFYING INDICATOR INVALID FOR CONSOLE FILE. ASSUME ID.
- † NOTE422 RECORD IDENTIFICATION CODES IN COLUMNS 35-41 ARE INVALID FOR THIS DEVICE. BLANKS IS ASSUMED.
- NOTE423 FILE TYPE (COL 15) IS INVALID FOR CONSOLE FILE. ENTRY OF I IS ASSUMED.
- † NOTE424 INVALID OVERLAPPING FIELDS. SPECIFICATION IS NOT PROCESSED.
- † NOTE425 WORKSTATION AND CONSOLE FILE NOT ALLOWED IN SAME PROGRAM. SPECIFICATION IS NOT PROCESSED.
- † NOTE426 MAXIMUM NUMBER OF RECORDS (10) FOR CONSOLE FILE EXCEEDED. SPECIFICATION IS NOT PROCESSED.
- † NOTE427 AND LINE IS INVALID FOR DEVICE. COLS 14-16.
- † NOTE428 INVALID OR BLANK ENTRIES IN COLUMNS 21-34 FOR THIS DEVICE.
- † NOTE429 FILE DESCRIPTION (COL 16) INVALID FOR DEVICE.
- † NOTE430 FIELD LENGTH DEFINED FOR RECORD IDENTIFICATION DOES NOT MATCH LENGTH OF IDENTIFICATION.
- † NOTE431 INVALID POSITION FOR FIRST FIELD TO BE PROMPTED.
- † NOTE432 TO FILENAME MUST BE LIMITS FILE IF FROM FILE IS CONSOLE.
- † NOTE433 INCONSISTENT USAGE OF RECORD IDENTIFYING CODES ON OR LINE. SPECIFICATION IS NOT PROCESSED.
- NOTE434 WARNING: TOTAL LENGTH OF ALL CONTROL FIELDS IS GREATER THAN 144 CHARACTERS.
- NOTE435 FIELD LENGTH GREATER THAN 66 IS INVALID FOR DEVICE. SPECIFICATION IS NOT PROCESSED.
- NOTE436 COL. 68 OF CONTINUATION STATEMENT IS NOT A BLANK OR D. ENTRY OF BLANK ASSUMED.
- NOTE437 COL. 69 OF CONTINUATION STATEMENT IS NOT A BLANK OR C. ENTRY OF BLANK ASSUMED.

- ↓
- NOTE438 MORE THAN 5 KEY STRUCTURES DESCRIBED FOR A FILE. STATEMENT NOT PROCESSED.
- NOTE439 FILE KEY STRUCTURE SPECIFIED IN COL 29-30 AND COL 35-38. CONTINUATION STATEMENT IS NOT PROCESSED.
- † NOTE440 SCREEN FORMAT NAME MISSING FOR WORKSTATION OUTPUT RECORD. ALL SPECIFICATIONS IN THIS RECORD TYPE ARE NOT PROCESSED.
- † NOTE441 INDICATOR IN COLS. 7-17 INVALID. SPECIFICATION NOT PROCESSED.
- NOTE442 WARNING: NO DATA STRUCTURE NAME FOUND ON INPUT SPECIFICATIONS MATCHING NAME ON WORKSTATION SAVDS CONTINUATION LINE.
- NOTE443 WARNING: IND AND/OR SAVDS IS SPECIFIED FOR WORKSTATION FILE BUT NUM IS NOT SPECIFIED OR IS NOT GREATER THAN 1. NO DATA SWAPPING WILL BE PERFORMED FOR WORKSTATIONS.
- NOTE444 WARNING: INDICATOR STATUS MAY NOT BE RETURNED TO USER, RESULTS MAY BE UNPREDICATABLE.
- NOTE445 WARNING: LENGTH OF FACTOR 2 VARIABLE ON ENDSR OPERATION FOR INFSR SUBROUTINE SUPPORT IS NOT 6. RESULTS MAY BE UNPREDICTABLE.
- NOTE446 WARNING: COLUMNS 60-65 BLANK FOR NUM CONTINUATION STATEMENT. 1 ASSUMED.
- † NOTE451 KEY LENGTH (COLUMNS 65-67) IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
- NOTE452 WARNING: INFSR SUBROUTINE IS MISSING OR INVALID.
- † NOTE453 RECORD LENGTH (COLUMNS 24-27) IS GREATER THAN 160 FOR S/34 CONSOLE RECORD ADDRESS FILE. SPECIFICATION IS NOT PROCESSED.
- NOTE454 FORMAT IS TOO LARGE TO FIT ON ONE SCREEN. FIELDS MUST BE REDUCED OR REDEFINED.
- ↑



NOTE459 NO ERROR MESSAGE ASSIGNED FOR THIS NOTE

NOTE460 NO ERROR MESSAGE ASSIGNED FOR THIS NOTE

NOTE461 NO ERROR MESSAGE ASSIGNED FOR THIS NOTE

NOTE462 NO ERROR MESSAGE ASSIGNED FOR THIS NOTE

NOTE463 NO ERROR MESSAGE ASSIGNED FOR THIS NOTE

NOTE482 WARNING: KEY CHANGE HAS TO BE SPECIFIED IN IMS  
CONFIGURATION. ENTRY IGNORED. ↓

NOTE483 REFER NOT ALLOWED FOR BATCH PROGRAMS.  
SPECIFICATION IS NOT PROCESSED.

NOTE484 REFER NOT ALLOWED FOR OUTPUT FILES. SPECIFICATION  
NOT PROCESSED.

NOTE485 NO PRIMARY KEY SPECIFIED FOR MULTIKEY FILE. FIRST KEY  
SPECIFIED ASSUMED AS PRIMARY.

NOTE486 WARNING: PRIMARY KEY ALREADY DEFINED. ENTRY  
IGNORED.

NOTE487 WARNING: KEY OF REFERENCE ALREADY DEFINED. ENTRY  
IGNORED.

NOTE488 WARNING: DUPLICATES NOT ALLOWED FOR PRMIARY KEY.  
ENTRY IGNORED.

NOTE489 SETK ONLY ALLOWED FOR CHAINED OR DEMAND FILES.  
SPECIFICATION IGNORED. ↑



## Appendix E. Main Storage Conservation Techniques

### E.1. GENERAL

The following subsections describe several techniques that you can use to reduce the amount of main storage required for executing your RPG II program.

### E.2. DIVIDING A PROGRAM INTO SEPARATE TASKS

If you have a program that performs more than one task, you can save main storage by dividing the program into separate tasks; that is, write a separate program for each task and execute each program separately. For example, assume that you have a program that updates a file and prints a listing of the updated file. You could save main storage by writing a separate program to update the file and a separate program to print a listing of the file.

### E.3. USING IRAM TO PROCESS ALL DISK FILES

Main storage can be saved by specifying that all disk files are to be processed by IRAM. In the IBM System/3 mode, IRAM is always used. For other modes, a considerable saving will result, especially if your program uses more than one access method; for example SAM and ISAM.

Note that before you can create an IRAM file that is to be accessed by another language processor, check to see if IRAM is supported by that language processor.

If you want to specify that IRAM is to be used, include a `// PARAM MOD = IRAM` statement in your compilation job control stream or, if you use the RPG job control procedure call statement, include the keyword parameter `MOD = IRAM`. Note that if you use IRAM, you should design your files so that the record lengths are even multiples of 256.

### E.4. CONTROL CARD SPECIFICATIONS FORM

The following subsections describe the entries and techniques you can use on this form that will reduce main storage requirements for program execution.

#### E.4.1. Error Analysis Dump

Do not specify an error analysis dump; that is, leave this field (column 8) blank.

### **E.4.2. Sign Handling**

Do not specify either input or output sign checking; that is, enter a B in this field (column 40).

## **E.5. FILE DESCRIPTION SPECIFICATIONS FORM**

The following subsection describes the entries and techniques you can use on this form that will reduce main storage requirements for program execution.

### **E.5.1. Device**

If you have card input, enter CTLRDR rather than READER in this field (columns 40 through 46).

## **E.6. INPUT FORMAT SPECIFICATIONS FORM**

The following subsections describe the entries and techniques you can use on this form that will reduce main storage requirements for program execution.

### **E.6.1. Record Identification Codes**

When you specify record identification codes (columns 21 through 41), use as few as possible to identify a record type and those that you specify should be specified with a C rather than Z or D in the C/Z/D field (column 26, 33, or 40).

### **E.6.2. Identical Input Fields in Two or More Record Types**

If the majority of the input fields are identical in two or more record types, use OR lines (columns 14 and 15) and field record relation entries (columns 63 and 64) for these fields rather than defining them for each record type. The fields used with a field record relation entry must be grouped together; that is, if three fields were related to record type 02, these fields must be described on contiguous lines and 02 must be specified in field record relation field (columns 63 and 64) on each line.

### **E.6.3. Fields Not Used in Your Program**

If there are fields in an input record that are not used in your program, do not describe these fields when you describe the fields for a given record type.

### **E.6.4. Using Same Field Name for Different Fields in Different Record Types**

Use the same field name if the field size and format (alphanumeric or numeric with the same number of decimal places) are the same for two or more input fields in different record types and the data from these fields does not have to be preserved from one record type to another.

### **E.6.5. Specifying a Numeric Field as a Nonnumeric Field**

Do not specify a numeric field as a numeric field unless it is to be used in a numeric operation in your program.

## **E.7. CALCULATION SPECIFICATIONS FORM**

The following subsections describe the entries and techniques you can use on this form that will reduce the main storage requirements for program execution.

### **E.7.1. Using the Same Result Field as a Common Work Field**

Use the same name for the result field in those cases where you have several calculations that require a result field with the same characteristics and do not require the result to be retained.

### **E.7.2. Using the GOTO Operation in Place of Conditioning Indicators**

Use the GOTO operation to bypass a group of inapplicable operations rather than conditioning each inapplicable operation by indicators, especially when different conditioning indicators would be required for two or more of these operations.

### **E.7.3. Grouping Operations by Indicator**

If conditioning indicators are required, group the operations that are conditioned by the same indicator together.

### **E.7.4. Using Subroutines Rather Than Repeating Identical Operations**

Use a subroutine rather than repeating identical operations at different points in your program.

### **E.7.5. Using Odd Length Numeric Fields**

Use an odd length numeric field rather than an even length field whenever possible.

### **E.7.6. Moving a Common Array Element to a Work Area**

If the same array element is used in more than one operation, move the element to a nonarray work area, and then use the work area name in the subsequent operations.

### **E.7.7. Restricting the Use of Half Adjust**

Do not use half adjust (H in column 53) unless it is absolutely necessary to do so.

**E.7.8. Using Factor 1 or Factor 2 as the Result Field**

Use factor 1 or factor 2 as the result field whenever possible.

**E.7.9. Using Actual Bit Patterns**

When you use the BITON, BITOF, or TESTB operation, specify the actual bit pattern in factor 2.

## Appendix F. Program Conversion Requirements

### F.1. GENERAL

As you know, the OS/3 RPG II compiler allows you to compile source programs that were written for other compilers. Before you compile one of these programs, however, there are certain actions that must be taken because of incompatibilities that exist between a program for another compiler and OS/3 RPG II.

### F.2. IBM SYSTEM/3 AND SYSTEM/34 RPG II SOURCE PROGRAMS ←

The following subsections describe the incompatibilities that exist in IBM System/3 and System/34 source programs and the recommended remedial actions for them. ←

#### F.2.1. Compiling an IBM System/3 or System/34 RPG II Source Program ←

If you want to compile an IBM System/3 or System/34 source program, you must replace the control card specifications with one that contains a 4 in column 7 or you must override the existing control card specification. ←

In the latter case, you can do this by including a // PARAM MOD=3 statement in your job control stream or, if you use the RPG job control procedure call statement, by including the keyword parameter MOD=3.

##### F.2.1.1. Control Card Specifications

The following subsections describe the incompatibilities that exist in the control card specifications and the recommended remedial actions.

###### F.2.1.1.1. Core Size to Compile (Columns 7 through 9)

You may specify the amount of main storage required to compile in the // JOB statement in your job control stream. If you do not, the minimum amount of main storage will be used.

**F.2.1.1.2. Object Output (Column 10)**

This option is not supported by OS/3 RPG II. Use a // PARAM OUT statement in your job control stream or the OUT keyword parameter in the RPG job control procedure call statement to specify the output option.

**F.2.1.1.3. Listing Option (Column 11)**

This option is not supported by OS/3 RPG II. Use a // PARAM LST statement in your job control stream or the LST keyword parameter in the RPG job control procedure call statement to specify the listing options.

**F.2.1.1.4. Core Size to Execute (Columns 12 through 14)**

Automatic segmentation of object programs is not supported by OS/3 RPG II. Specify sufficient main storage in the // JOB statement in your job control stream.

**F.2.1.1.5. Inquiry (Column 37)**

Specify preemptive priority for the job in the // JOB statement in your job control stream.

**F.2.1.1.6. Punch MFCU Zeros (Column 44)**

This option is not supported by OS/3 RPG II.

**F.2.1.2. File Description Specifications**

The following subsections describe the incompatibilities that exist in the file description specifications and the recommended remedial actions.

**F.2.1.2.1. Device (Columns 40 through 46)**

The maximum record length for a console file is 60 bytes.

Console update files are not supported by OS/3 RPG II.

Device independent files are not supported by the OS/3 RPG II. A device name must be specified in this field.

**F.2.1.2.2. Continuation Lines (Columns 53 through 65)**

If the entry INDEX is specified in the option field (columns 54 through 59) when loading an indexed file in the IBM System/3 mode, it must be used in every program that uses the file and the same amount of main storage must be specified in each case.



### **F.2.1.2.3. Number of Extents (Columns 68 and 69)**

In OS/3 RPG II if a multivolume file is created with only one volume at a time online, it must be processed with only one volume online at a time. If all volumes are online when it is created, all volumes must be online when it is processed.

### **F.2.1.3. File Extension Specifications**

There are no known incompatibilities.

### **F.2.1.4. Line Counter Specifications**

There are no known incompatibilities.

### **F.2.1.5. Input Format Specifications**

When using data structures, if you define an internal area more than once and use both alphanumeric (unpacked) and numeric (packed) formats, check your definitions to make certain both formats do not coexist.

### **F.2.1.6. Calculation Specifications**

The following subsections describe the incompatibilities that exist in the calculation specifications and the recommended remedial actions.

#### **F.2.1.6.1. Divide Operation (DIV)**

Divide operation (DIV) operands are limited to the specific length and decimal position attributes that are specified in 7.3.2.1.6.

#### **F.2.1.6.2. Time Operation (TIME)**

If the time operation (TIME) is used, the format of the date is the OS/3 format yymmdd rather than ddmmyy or mmddyy.

### **F.2.1.7. Output Format Specifications**

In OS/3 RPG II, heading and detail output is not performed if conditioned by the LR indicator.

The following subsections describe the incompatibilities that exist in the output format specifications and the recommended remedial actions.

**F.2.1.7.1. \*PRINT**

\*PRINT for printing and punching cards is not supported in OS/3 RPG II.

**F.2.1.7.2. \* in Column 40**

\* in column 40 for printing on punched cards is not supported in OS/3 RPG II.

**F.2.1.7.3. LR Output**

Heading and detail output conditioned by the LR indicator will not be output. Once the LR indicator is set on, only total output and last record output are performed. See Figure A-1 for the detail logic cycle.

**F.2.1.8. Telecommunications Specifications**

The following subsections describe the incompatibilities in the telecommunications specifications and the recommended remedial actions.

**F.2.1.8.1. Configuration (Column 15)**

The P, M, and S configuration entries are not supported by OS/3 RPG II. This must be specified in the communications network that is defined at system generation time. This entry should be replaced with a blank, otherwise a warning note will appear. For further information, see 17.4.

**F.2.1.8.2. Type of Control (Column 17)**

Type of station is not supported. This entry must be a blank, otherwise a warning note will appear.

**F.2.1.8.3. Type of Code (Column 18)**

Type of code is not supported by OS/3 RPG II. This must be specified in the communications network that is defined at system generation time. This entry should be replaced with a blank, otherwise a warning note will appear. For further information, see 17.4.

**F.2.1.8.4. Dial Number (Columns 21 through 31)**

Dial number is not supported by OS/3 RPG II. This must be specified in the communications network that is defined at system generation time. This entry should be replaced with a blank, otherwise a warning note will appear. For further information, see 17.4.

**F.2.1.8.5. Identification - This Station (Columns 32 through 39)**

Identification - this station is not supported. This entry should be replaced with a blank, otherwise a warning note will appear.

#### **F.2.1.8.6. Identification - Remote Station (Columns 40 through 47)**

Identification - remote station is not supported. This entry should be replaced with a blank, otherwise a warning note will appear.

#### **F.2.1.8.7. ITB (Column 52)**

ITB is not supported by OS/3 RPG II. This must be specified in the communications network that is defined at system generation time. This entry should be replaced with a blank, otherwise a warning note will appear. For further information, see 17.4.

#### **F.2.1.8.8. Polling Characters (Columns 61 and 62)**

Polling characters are not supported. This entry should be replaced with a blank, otherwise a warning note will appear.

#### **F.2.1.8.9. Addressing Characters (Columns 63 and 64)**

Addressing characters are not supported. This entry should be replaced with a blank, otherwise a warning note will appear.

#### **F.2.1.8.10. Interspersed Mode**

Interspersed mode is not supported. This entry should be replaced with a blank; otherwise a warning note will appear.

### **F.2.2. Executing an IBM System/3 RPG II Program**

The following two guidelines apply to IBM System/3 RPG II programs compiled and run in an operation control language (OCL) environment:

1. RPG II programs that use punched cards must have the file name PUNCH on the file description specifications form for the card output file. Programs that have a file name other than PUNCH for these files must be modified before they are compiled. The RPG II compiler does not automatically substitute the file name PUNCH for the user-defined card output file name used in these programs.
2. RPG II programs that use matching field processing between two or more card files will produce incorrect results. To get the expected results, these card files must be placed on disk or tape so that only one card file is used.

### **F.3. SPERRY UNIVAC 9200/9300 SYSTEM RPG SOURCE PROGRAMS**

The following subsections describe the incompatibilities that exist in SPERRY UNIVAC 9200/9300 RPG source programs and the recommended remedial actions.

### **F.3.1. Compiling a SPERRY UNIVAC 9200/9300 RPG Source Program**

If you want to compile a SPERRY UNIVAC 9200/9300 RPG program, you must replace the control card specifications with one that contains a 3 in column 7.

### **F.3.2. Control Card Specifications**

The following subsections describe the incompatibilities that exist in the control card specifications and the recommended remedial actions.

#### **F.3.2.1. Listing Option (Column 11)**

This option is not supported by OS/3 RPG II. Use a // PARAM LST statement in your job control stream or the LST keyword parameter in the RPG job control procedure call statement to specify the listing option.

#### **F.3.2.2. Rewind (Column 12)**

This option is not supported by OS/3 RPG II. Use column 70 on the file description specifications to specify the rewind option.

#### **F.3.2.3. Program Identification (Columns 30 through 33 or 46 through 49)**

This option is not supported by OS/3 RPG II. Use columns 75 through 80 on the control card specifications to specify the program identification.

### **F.3.3. File Description Specifications**

The following subsections describe the incompatibilities that exist in the file description specifications and the recommended remedial action.

#### **F.3.3.1. File Name (Columns 7 through 13)**

Any character in a file name is not supported by OS/3 RPG II. All characters used in file names in your program must conform to the OS/3 RPG II requirements. For further information, see 5.2.1.

#### **F.3.3.2. Logical Unit Number (Columns 47 through 52)**

This option is not supported by OS/3 RPG II. Use job control statements for device assignments.

#### **F.3.3.3. Alternate Logical Unit Numbers (Columns 54 through 56)**

This option is not supported by OS/3 RPG II. Use job control statements for device assignments.

### **F.3.4. File Extension Specifications**

The following subsection describes the incompatibilities that exist in the file extension specifications and the recommended remedial action.

#### **F.3.4.1. Compilation Time Table Indicator (Column 6)**

This option (T in column 6) is not supported by OS/3 RPG II. Each compilation time table must be preceded by a record that contains \*\*blank in positions 1 through 3. If // PARAM COL=7 statement is specified, the \*\*blank are in positions 7 through 9.

### **F.3.5. Input Format Specifications**

The following subsections describe the incompatibilities that exist in the input format specifications and the recommended remedial action.

#### **F.3.5.1. Matching Fields**

If your program uses matching fields without field record relation or with field record relation, the fields must be specified in accordance with OS/3 RPG II requirements. For further information see 6.3.6.1 and 12.4.3.

#### **F.3.5.2. Invalid Numeric Data**

OS/3 RPG II will not process numeric data that contains improper digits or signs. If your program contains numeric data with improper digits or signs, correct the data before you attempt to process it.

### **F.3.6. Calculation Specifications**

The following subsection describes the incompatibilities on the calculation specifications and the recommended remedial action.

#### **F.3.6.1. UPSI Special Name**

This option is not supported in OS/3 RPG II. Use U1 through U8 indicators.

### **F.3.7. Output Format Specifications**

No incompatibilities.



## Appendix G. Auto Report Error Messages

### G.1. GENERAL

If any errors are detected in your program while it is being analyzed by auto report, the UPSI byte is set in the communication region of the job preamble according to OS/3 system standards to indicate the type of errors that occurred.

<u>UPSI Byte</u>	<u>Setting</u>	<u>Meaning</u>
Bit 0	1	Catastrophic errors were detected in the source program. Generated source was incomplete.
	0	No catastrophic errors were detected.
Bit 1	1	Serious errors were detected. A source module was generated but the results will be unpredictable.
	0	No serious errors were detected.

### G.2. COMPILE-TIME MESSAGES

When an error occurs during the processing of the source program, the appropriate message from the following list is printed on the compilation listing. An asterisk (\*) indicates that the message is printed out due to a catastrophic error. A dagger (†) indicates that the message is printed out due to a serious error. A blank indicates that the message printed out is informational. These indicators are not printed out with the message.

\* **AR001 SOURCE PROGRAM IS MISSING. PROGRAM IS TERMINATED.**

Terminal error. The first record in the source program is /\* or \*\*.

Auto report terminates the job.

Correct the source program and resubmit the job.

↓

† **AR002 RPG II CONTROL SPECIFICATION IS MISSING. A CONTROL SPECIFICATION WITH BLANK ENTRIES IS CREATED.**

Warning error for control (H) specification. The source program or the copied module doesn't contain a control specification.

Auto report creates a control specification containing blank entries.

Correct the source program or module.

† **AR003 SOURCE PROGRAM CONTAINS MORE THAN ONE RPG II CONTROL SPECIFICATION. ALL BUT THE FIRST ARE DROPPED.**

Warning error for control (H) specification. The source program contains multiple control specifications or the copied module contains a control specification.

Auto report drops all but the first control specification.

Correct the source program and resubmit the job.

† **AR004 DUPLICATE FILENAMES ON THE FILE DESCRIPTION SPECIFICATION. DUPLICATE IS DROPPED.**

Warning error for file description (F) specification. The file name on the specification isn't unique.

Auto report ignores the duplicate file description specification.

Assign a unique name to the file and resubmit the job.

† **AR005 REQUESTED SOURCE MODULE CANNOT BE FOUND. SPECIFICATION IS DROPPED.**

Warning error for /COPY specification. The module isn't located because:

1. You used the wrong name for the module.
2. You specified the wrong disk for the library file.
3. There aren't any records in the library file.

Auto report ignores the specification.

Correct the error and resubmit the job.

↑



† **AR006 DUPLICATE FILENAMES ON F-SPECS READ FROM THE FILE SOURCE.** ↓

Warning error for file description (F) specification. You can use only one file description specification for a file name in a file module. The error occurs because:

1. Two file description specifications with the same file name are in one file module.
2. File description specifications with the same file name are in more than one file module.

Auto report ignores the duplicate file description specification.

Assign a unique file name and resubmit the job.

† **AR007 TABLE AREA PROVIDED FOR INPUT OVERRIDES EXCEEDED. OVERRIDE FUNCTION IS DISCONTINUED FOR THIS /COPY.**

Warning error for /COPY statement. The number of /COPY modifier statements for the input field specification exceeds the available space in the table. The fields that can be overwritten are added to the table. Because of this, you can get invalid specifications in the generated program.

Auto report discontinues the override function for the /COPY statement. ↑

Place override statements first, then specify the input fields added to the copied specifications. This permits the override statements to use all the table space. Auto report can handle at least 20 overrides.

† **AR008 INVALID RPG II SPECIFICATION TYPE. SPECIFICATION IS DROPPED.** ←

Warning error for /COPY statement. The specification is invalid. The error occurs because:

1. Column 6 doesn't contain an H, F, E, L, T, I, C, or O.
2. Column 7 doesn't contain an asterisk.
3. The /COPY statement is on a control (H) specification. (The /COPY statement is processed correctly.)

Auto report ignores the specification. ←

Correct the error and resubmit the job.

→ \* **AR009 INVALID OR UNDEFINED FILE FOR \*AUTO LINES.  
DROP ALL SPECIFICATIONS TO NEXT RECORD TYPE.**

Terminal error for file description (F) or output (O) specification. The file is invalid or undefined. The error occurs because:

1. The auto report file isn't a printer or line counter file.
2. The auto report file doesn't contain a file description specification.
3. The names on the file description specification and the auto report file don't match.

The job terminates.

Correct the error and resubmit the job.

↓ † **AR010 TABLE AREA FOR FIELD NAMES USED ON \*AUTO LINES EXCEEDED.  
SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. Each field you used in the H-\*AUTO, D-\*AUTO, or T-\*AUTO specifications is placed into a table.

Auto report ignores the specification.

Take out the unused and unneeded field names and resubmit the job.

**AR011 TABLE AREA PROVIDED FOR FIELD NAMES EXCEEDED. NON-  
UNIQUE FIELD NAMES MAY BE GENERATED.**

Warning error for output (O) specification. The field name table contains the generated field names that end in a number from 1 through 9 or an R.

Auto report may generate the nonunique field names.

To avoid duplicate field names, don't use names that end in a number from 1 through 9 or an R.

**AR012 GENERATED TOTAL FIELD PREVIOUSLY DEFINED WITH DIFFERENT  
ATTRIBUTES. PREVIOUS DEFINITION IS USED.**

Warning error for output (O) specification. You defined the generated total field with a different field length or a different number of decimal positions.

Auto report uses the first or previous definition and prints both the total field and generated field names with the error note number.

↑ Correct the error and resubmit the job.

† **AR013 \*AUTO PREVIOUSLY USED FOR A DIFFERENT FILE. DROP ALL SPECIFICATIONS TO NEXT RECORD TYPE.**

Warning error for output (O) specification. You can specify \*AUTO for only one file.

Auto report drops all specifications to the next record type.

Correct the error by specifying \*AUTO for only one file and resubmit the job.

**AR014 POSITIONS 7-22 ARE NOT BLANK ON OUTPUT FIELD SPECIFICATION. BLANKS ARE ASSUMED.**

Warning error for output (O) specification. You must leave columns 7 through 22 of the output field specification blank.

Auto report assumes blanks are in columns 7 through 22.

Leave columns 7 through 22 blank and resubmit the job.

**AR015 INVALID INDICATOR. BLANKS ARE ASSUMED.**

Warning error for output (O) specification. Columns 24 through 25, 27 and 28, or 30 and 31 don't contain 01 through 99, L0 through L9, MR, 1P, H1 through H9, OA through OG, OV, or blanks.

Auto report prints the invalid indicator with the note number and assumes blanks in the columns.

Enter a valid indicator and resubmit the job.

† **AR016 INVALID FIELD NAME. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. The field name is invalid. The error occurs because:

1. The field name wasn't found.
2. You didn't define the field name.
3. The array index contains a blank after the comma or it has a comma as the first character.
4. Field name usage is invalid with a constant.

Auto reports ignores the specification and the column headings for the field.

Enter a field name in columns 32 through 37 and resubmit the job.

↓

**AR017 INVALID ENTRY IN POSITION 38 AND/OR 44. BLANKS ARE ASSUMED.**

Warning error for output (O) specification. You must leave column 38 and column 44 blank for alphanumeric fields.

Auto report assumes blanks are in column 38 and column 44.

Leave columns 38 and 44 blank and resubmit the job.

**AR018 INVALID ENTRY IN POSITION 39. BLANK IS ASSUMED.**

Warning error for output (O) specification. Column 39 must contain blank or B for any field on an H-\*AUTO specification.

Auto report assumes a blank is in column 39.

Enter B or blank in column 39 and resubmit the job.

**AR019 INVALID ENTRY IN POSITIONS 40-43. BLANKS ARE ASSUMED.**

Warning error for output (O) specification. You can't specify an end position on the field description for an H-\*AUTO specification.

Auto report assumes blanks are in columns 40 through 43.

Leave columns 40 through 43 blank and resubmit the job.

**AR020 INVALID ENTRY IN POSITIONS 45-70. BLANKS ARE ASSUMED.**

Warning error for output (O) specification. You can't specify a heading or edit word with an alphanumeric field.

Auto report assumes blanks are in columns 45 through 70.

Enter valid entries in columns 45 through 70 and resubmit the job.

**AR021 FIELD NAME WILL BE CONDITIONED BY THE INDICATOR N1P.**

Warning error for output (O) specification. You specified a field name on an H-\*AUTO specification but didn't enter an indicator to condition the line. If the field is printed on the first page, it may contain meaningless data because the first record has not been read.

Auto report generates an N1P indicator for this specification.

Enter the proper indicator and resubmit the job.

↑

**AR022 INVALID EDIT CODE, POSITION 38. BLANK IS ASSUMED.**

Warning error for output (O) specification. Column 38 doesn't contain A, B, C, D, J, K, L, M, 1, 2, 3, 4, X, Y, or Z.

Auto report assumes a blank in column 38.

Enter a valid edit code in column 38 and resubmit the job.

**AR023 INVALID ENTRY IN POSITION 44. BLANK IS ASSUMED.**

Warning error for output (O) specification. You must leave column 44 blank for a numeric field.

Auto report assumes a blank in column 44.

Leave column 44 blank and resubmit the job.

**AR024 POSITIONS 23-31 ARE NOT BLANK FOR TOTALING FIELD.**

Warning error for output (O) specification. The indicators you specified on a totaling field with a T-\*AUTO specification aren't used when specifications are generated.

Auto report assumes blanks are in columns 23 through 31.

Leave columns 23 through 31 blank and resubmit the job.

**AR025 CONSTANT/EDIT WORD IN POSITIONS 45-70 IS INVALID.  
BLANKS ARE ASSUMED IN POSITIONS 45-70.** ←

Warning error for output (O) specification. The entry in columns 45 through 70 is invalid. The error occurs because:

1. The entry in columns 45 through 70 doesn't begin with an apostrophe.
2. The entry in columns 45 through 70 doesn't end with an apostrophe.
3. There is an embedded single apostrophe (not paired with a corresponding apostrophe) in columns 46 through 69.
4. Columns 45 through column 70 aren't blank after the last apostrophe.

Auto report drops this specification if you didn't specify a field name and assumes blanks in columns 45 through 70 if you did specify a field name.

Correct the entry in columns 45 through 70 and resubmit the job.

↓

**† AR026 UNABLE TO DETERMINE IF FIELD OR RECORD SPECIFICATION.  
SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. Column 15 is blank, which indicates that the specification is a field description, but columns 32 through 37 and 45 through 70 are also blank.

Auto report drops the specification and the column headings for the field.

Make the proper entries and resubmit the job.

**AR027 POSITIONS 38-44 ARE NOT BLANK WHEN A LITERAL IS SPECIFIED.  
BLANKS ARE ASSUMED.**

Warning error for output (O) specification. You must leave columns 38 through 44 blank when you specify a title on the H-\*AUTO specification.

Auto report assumes blanks are in columns 38 through 44.

Leave columns 38 through 44 blank when specifying a title and resubmit the job.

**AR028 POSITIONS 7-13 ARE NOT BLANK ON AND/OR SPECIFICATION.  
BLANKS ARE ASSUMED.**

Warning error for output (O) specification. You must leave columns 7 through 13 blank when you specify AND in columns 14 through 16 or specify OR in columns 14 and 15.

Auto report assumes blanks are in columns 7 through 13.

Leave columns 7 through 13 blank when using AND or OR entries and resubmit the job.

**AR029 SPACE AND/OR SKIP ENTRIES IN POSITIONS 17-22 ARE INVALID.  
BLANKS ARE ASSUMED.**

Warning error for output (O) specification. The space entry in columns 17 and 18 doesn't contain 0 through 9 or blank, and the skip entry in columns 19 and 20 and columns 21 and 22 doesn't contain 01 through 15.

Auto report assumes blanks are in columns 17 through 22.

Make the proper space and skip entries and resubmit the job.

**AR030 POSITIONS 37-70 NOT BLANK ON RECORD SPECIFICATION.  
BLANKS ARE ASSUMED.**

Warning error for output (O) specification. You must leave columns 37 through 70 blank for output file identifications.

Auto report assumes blanks are in columns 37 through 70.

Leave columns 37 through 70 blank and resubmit the job.

↑

**AR031 INVALID ENTRY IN POSITION 38. BLANK IS ASSUMED.** ↓

Warning error for output (O) specification. You can't specify an edit code along with a heading on D-\*AUTO or T-\*AUTO specifications.

Auto report assumes a blank is in column 38.

Leave column 38 blank and resubmit the job.

**AR032 END POSITION IN POSITIONS 40-43 IS INVALID. BLANKS ARE ASSUMED.**

Warning error for output (O) specification. Columns 40 through 43 contain invalid numbers or the end position exceeds the record length.

Auto report assumes blanks are in columns 40 through 43.

Enter a valid end position in columns 40 through 43 and resubmit the job.

**AR033 GENERATED FIELD LENGTH EXCEEDS 15. 15 IS ASSUMED.**

Warning error for output (O) specification. When you specify a D-\*AUTO or T-\*AUTO specification with an A in column 39, the generated total field has a length that is two positions longer than the field you specify. An error occurs when the generated total field exceeds 15.

Auto report assumes the generated total field is 15 positions long.

Decrease the length of the field and resubmit the job.

**AR034 DEFINITION OF FIELD IS INVALID. DEFINITION NOT USED.**

Warning error for field description (F), file extension (E), input (I), or calculation (C) specification. The definition of the field is invalid. The error occurs because:

1. The length is 0.
2. The length is greater than 15 for a numeric field.
3. The length is not numeric.
4. The length is less than the decimal position.
5. The decimal position is not numeric.
6. Column 43 doesn't contain P, B, or blank.

Auto report ignores the definition of the field.

Enter a valid field definition and resubmit the job.

 ↑

↓

**AR035 ARRAY NAME IS SPECIFIED ON \*AUTO LINE. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. The name is invalid. The error occurs because:

1. You specified the name of an array in columns 32 through 37 of the H-\*AUTO, D-\*AUTO, or T-\*AUTO specification.
2. The name generated by a D-\*AUTO or T-\*AUTO specification is an array name. The system drops the specification along with its column headings.

Auto report prints both the total field name and the generated field array name or it drops the specification.

Enter the name of a numeric field that is to be accumulated in columns 32 through 37 and resubmit the job.

**AR036 RECORD LENGTH FOR FILE WITH \*AUTO LINES IS INVALID. ASSUME RECORD LENGTH OF 120.**

Warning error for file description (F) specification. The record length is invalid. The error occurs because:

1. The record length is 0.
2. The record length is not numeric.
3. The record length is blank.

Auto report assumes the record length is 120.

Enter a valid record length and resubmit the job.

**AR037 TOTALING (A IN POSITION 39) SPECIFIED FOR AN INVALID FIELD NAME. ASSUME POSITION 39 IS BLANK.**

Warning error for output (O) specification. Column 39 on a D-\*AUTO or T-\*AUTO contains an A, but the field name is:

1. Blank
2. A table name
3. An indexed array name
4. A page field

Auto report drops the specification and all its column headings.

Enter the name of a numeric field that is to be accumulated in columns 32 through 37 and resubmit the job.

↑



↓

**AR038 TOTALING (A IN POSITION 39) SPECIFIED FOR AN ALPHANUMERIC FIELD. ASSUME POSITION 39 IS BLANK.**

Warning error for output (O) specification. The field name you entered in on the D-\*AUTO or T-\*AUTO specification is alphanumeric but there is an A in column 39.

Auto report assumes a blank is in column 39.

Enter the name of a numeric field that is to be accumulated in columns 32 through 37 and resubmit the job.

**AR039 POSITIONS 7-38 NOT BLANK FOR A COLUMN HEADING. BLANKS ARE ASSUMED.**

Warning error for output (O) specification. When you specify a C in column 39, columns 7 through 38 must be blank.

Auto report assumes blanks are in columns 7 through 38 when there is a C in column 39.

Leave columns 7 through 38 blank and resubmit the job.

**AR040 INVALID ENTRY IN POSITION 39. BLANK IS ASSUMED.**

Warning error for output (O) specification. The entry in column 39 is invalid. The error occurs because:

1. Column 39 contains B but there isn't a field name.
2. Column 39 doesn't contain A, B, C, 1 through 9, R, or blank on the D-\*AUTO or T-\*AUTO field description.

Auto report assumes a blank is in column 39.

Enter a valid entry in column 39 and resubmit the job.

**† AR041 COLUMN HEADING, C IN POSITION 39, SPECIFIED BUT LITERAL NOT PRESENT. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. When you specify a C in column 39, columns 45 through 70 must contain a column heading.

Auto report ignores the specification.

Enter a column heading in columns 45 through 70 and resubmit the job.

↑

↓

**AR042 EDIT CODE AND EDIT WORD ARE BOTH SPECIFIED. EDIT WORD IS DROPPED.**

Warning error for output (O) specification. You can't use an edit code in column 38 and an edit word in columns 45 through 70.

Auto report assumes blanks are in columns 45 through 70.

Leave columns 45 through 70 blank when you specify an edit code and resubmit the job.

**AR043 EDITING SPECIFIED FOR AN ALPHANUMERIC FIELD. ASSUME BLANKS IN POSITIONS 38 AND 45-70.**

Warning error for output (O) specification. You must leave columns 38 and 45 through 70 blank for alphanumeric fields.

Auto report assumes blanks are in columns 38 and 45 through 70.

Correct the error and resubmit the job.

**AR044 INVALID ENTRY IN POSITION 16. BLANK IS ASSUMED.**

Warning error for output (O) specification. Column 16 doesn't contain F or blank.

Auto report assumes a blank is in column 16.

Enter a valid entry in column 16 and resubmit the job.

† **AR045 AND/OR SPECIFICATION OUT OF SEQUENCE. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. The AND or OR entries in columns 14 through 16 don't follow a D-\*AUTO or T-\*AUTO output file identification.

Auto report drops the specification.

Ensure that the output file identification entries in columns 15 through 31 precede any AND or OR lines and resubmit the job.

† **AR046 MULTIPLE D/T-\*AUTO LINES SPECIFIED IN THE PROGRAM. DROP ALL SPECIFICATIONS TO NEXT RECORD TYPE.**

Warning error for output (O) specification. You can't use D-\*AUTO and T-\*AUTO specifications in the same program.

Auto report drops all specifications to the next record type.

Use either D-\*AUTO or T-\*AUTO specifications and resubmit the job.

↑



† **AR047 COLUMN HEADING SPECIFICATION OUT OF ORDER.  
SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. A field description with a C in column 39 must follow a field description with a C, B, A, or blank in column 39.

Auto report ignores the specification.

Place the column heading specification (C in column 39) in the correct order and resubmit the job.

**AR048 END POSITION INVALID FOR THIS SPECIFICATION. ASSUME  
BLANKS IN POSITIONS 40-43.**

Warning error for output (O) specification. You can't specify an end position if there is a C, R, or 1 through 9 in column 39.

Auto report assumes blanks are in columns 40 through 43.

Leave columns 40 through 43 blank and resubmit the job.

**AR049 SPECIFIED END POSITION IS LESS THAN FIELD OR LITERAL  
LENGTH. ASSUME BLANKS IN POSITIONS 40-43.**

Warning error for output (O) specification. The end position you specify in columns 40 through 43 must be at least as large as the field, column heading, or heading.

Auto report assumes blanks are in columns 40 through 43.

Enter a valid numeric entry in columns 40 through 43 and resubmit the job.

† **AR050 MORE THAN THREE COLUMN HEADING LINES SPECIFIED.  
SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. You can only use two consecutive specifications with a C in column 39.

Auto report ignores the specification.

Remove the incorrect column heading specification (C in column 39) and resubmit the job.

**AR051 NO VALID TOTALING FIELD SPECIFIED.**

Warning error for output (O) specification. Column 39 on a D-\*AUTO or T-\*AUTO specification doesn't contain an A. Because of this, no automatic totaling is done and no total lines are generated.

Auto report drops the total line entries 1 through 9 or R in column 39.

Enter an A in column 39 when you want automatic totaling and then resubmit the job.



↓

† **AR052 1-9, R IS INVALID IN POSITION 39. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. Column 39 contains a number from 1 through 9 but you didn't specify the associated level indicator L1 through L9 in columns 59 and 60 of the input specification. Or, the lowest level indicator you used on a T-\*AUTO specification is of greater or equal level to the associated level indicator L1 through L9 in columns 59 and 60 of the input specifications.

Auto report ignores the specification.

If you specified a 2 in column 39, then you must specify L2 as a level indicator on the input specifications. If this is a T-\*AUTO specification, the lowest level indicator you can use is L1. Make the necessary entries and resubmit the job.

**AR053 INDICATORS NOT ALLOWED ON THIS SPECIFICATION TYPE. BLANKS ARE ASSUMED IN POSITIONS 23-31.**

Warning error for output (O) specification. You can't use indicators on:

1. Field descriptions that follow an H-\*AUTO specification.
2. Field descriptions with 1 through 9 or R in column 39 that follow a D-\*AUTO or T-\*AUTO specification.

Auto report assumes blanks are in columns 23 through 31.

Leave columns 23 through 31 blank and resubmit the job.

**AR054 SPECIFIED END POSITION CAUSES OVERLAYING OF FIELDS OR LITERALS. BLANKS ARE ASSUMED IN POSITIONS 40-43.**

Warning error for output (O) specification. The end position you specified is less than the length of the line up to this specification plus the length of the field, column heading, or heading. An end position is automatically generated if you leave columns 40 through 43 blank.

Auto report assumes blanks are in columns 40 through 43.

Make the corrections and resubmit the job.

\* **AR055 I/O ERROR OCCURRED. PROGRAM IS TERMINATED.**

Terminal error. An input/output error has occurred. Additional information is printed along with the error that describes the problem:

1. Permanent disk error.
  2. Not enough tracks are allocated for work file.
- ↑

3. Not enough tracks are allocated for source file.

Auto report terminates the job.

Increase the size of the work file and resubmit the job.

† **AR056 LIBRARY SOURCE MEMBER NAME IS INVALID. ENTRY IS DROPPED.**

Warning error for the auto report options (U) specification or the /COPY statement. A /COPY statement can't have a U or H in column 6 of the specifications form. The auto report options specifications form must have a U in column 6. The error occurs because:

1. F1 or LFD name is missing.
2. There isn't a comma between the LFD name and module name.
3. The LFD name or module exceeds 8 characters.
4. The LFD name or module is missing, doesn't exist in the library file, or is incorrectly specified.
5. The LFD name or module name contains an embedded blank.

Auto report drops the entry.

Identify and correct the error and resubmit the job.

† **AR057 TOTALING SPECIFIED MORE THAN ONCE FOR THIS FIELDNAME. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. You can specify totaling (A in column 39) for any particular field name only once in each program.

Auto report drops the specification.

Enter A in column 39 only once for each field name and resubmit the job.

† **AR058 MAXIMUM NUMBER OF H-\*AUTO LINES EXCEEDED. DROP ALL SPECIFICATIONS TO THE NEXT RECORD TYPE.**

Warning error for output (O) specification. You can't specify more than five H-\*AUTO specifications.

Auto report drops all H-\*AUTO specifications after the first five.

Ensure that there is no more than five H-\*AUTO specifications and resubmit the job.



↓

**AR059 INVALID ENTRY IN POSITION 7 OF U SPECIFICATION. BLANK IS ASSUMED.**

Warning error for auto report options (U) specification. You must use C or blank in column 7.

Auto report assumes a blank is in column 7.

Use C or blank in column 7 and resubmit the job.

**AR060 FILE NAME IN POSITIONS 8-24 IS NOT BLANK. BLANKS ARE ASSUMED.**

Warning error for auto report options (U) specification. You must leave columns 8 through 24 blank if column 7 is blank. If column 7 contains a C, then:

1. Columns 8 through 24 must contain F1 or LFD name followed by a comma and the module name.
2. The name must be one to eight characters long.
3. The first character of the name must be alphabetic.

Auto report assumes blanks are in columns 8 through 24.

Make the proper entries and resubmit the job.

**AR061 INVALID ENTRY IN DATE SUPPRESS, POSITION 27. BLANK IS ASSUMED.**

Warning error for auto report options (U) specification. You must use N or blank in column 27.

Auto report assumes a blank is in column 27.

Use N or blank in column 27 and resubmit the job.

**AR062 INVALID ENTRY IN ASTERISK SUPPRESS, POSITION 28. BLANK IS ASSUMED.**

Warning error for auto report options (U) specification. You must use N or blank in column 28.

Auto report assumes a blank is in column 28.

Use N or blank in column 28 and resubmit the job.

↑

**† AR063 AND/OR SPECIFICATION IS INVALID. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. The AND or OR entry is invalid. The error occurs because:

1. The output file identification doesn't contain indicators that condition the line.
2. The AND or OR entry doesn't have indicators.

You can use AND or OR entries with \*AUTO output indicators when you enter an output indicator on the first output file identification.

Auto report ignores the specification.

Make the proper entries and resubmit the job.

**AR064 D/T-\*AUTO LINE OVERFLOW WILL OCCUR WITH GENERATION OF  
ASTERISK INDICATION. ALL ASTERISKS ARE SUPPRESSED.**

Warning error for output (O) specification. One or more asterisks cause an overflow to the printer record length you defined.

Auto report suppresses all asterisks.

Enter an N in column 28 of the auto report options (U) specification to suppress asterisk indication, then resubmit the job.

**† AR066 MORE THAN 20 AND/OR LINES CONDITION AN \*AUTO LINE. THIS  
AND ALL FOLLOWING AND/OR SPECIFICATIONS ARE DROPPED.**

Warning error for output (O) specification. You can't use more than 20 AND or OR entries.

Auto report ignores the excessive AND or OR specifications.

Remove all excessive AND or OR lines.

**† AR068 NUMBER OF FILE DESCRIPTION SPECIFICATIONS EXCEEDS THE  
MAXIMUM ALLOWED. SPECIFICATION IS DROPPED.**

Warning error for file description (F) specification. This error occurs when:

1. You have more than 20 file description specifications.
2. You have exceeded the internal limit (110) for file description specifications.

Auto report ignores the excessive specifications.

Remove all excessive file description specifications and resubmit the job.



↓

**AR069 AUTOMATIC TOTALING OF THIS FIELD RESULTS IN GENERATED FIELD NAME CONFLICTS. ASSUME POSITION 39 BLANK.**

Warning error for output (O) specifications. This error occurs because:

1. A field name that was generated for totaling was previously defined as alphanumeric.
2. Another field name, which duplicates the first five characters of this field name, appears in the program and is used as a totaling field. (Both names are printed.)

**AR070 GENERATED LINE IS TOO LONG. EXCESS IS DROPPED.**

Warning error for output (O) specifications. This error occurs because:

1. The length of H-\*AUTO line exceeds the record length.
2. The length of D/T\*AUTO line exceeds twice the record length.

† **AR071 INVALID OUTPUT RECORD TYPE IN POSITION 15. SPECIFICATION IS DROPPED.**

Warning error for output (O) specification. You must enter H, D, T, or E in column 15.

Auto report ignores the specification.

Enter a valid entry in column 15 and resubmit the job.

**AR072 PAGE FIELD NOT AVAILABLE FOR USE IN PAGE HEADING. NO PAGE NUMBERING WILL OCCUR.**

Warning error for output (O) specification. For page numbering, auto report uses one of the unused page fields (PAGE, PAGE1 through PAGE7). No page numbering occurs if all these page fields have been used. The format of the page heading line may not be correct.

No page numbering occurs.

Ensure that one of the page fields is available and resubmit the job.

\* **AR073 ATTEMPT TO PUT SOURCE MODULE IN THE FILE FAILED. FILE NOT WRITTEN.**

Warning error. This error occurs because:

1. The library file is full.
2. There is an invalid operation (such as the library file not being allocated).

Auto report doesn't catalog the program.

↑  
Correct the error and resubmit the job.



**AR074 DUPLICATE FILE SOURCE MODULE NAME WAS FOUND IN THE FILE.  
PREVIOUS MODULE WAS REPLACED.**

Warning error. The module name you specified is the same as a name that is already in the library file.

This program replaces the old module.

Use a unique name or delete the old module before you catalog the new module.

**AR075 ENTRIES IN COLUMNS 25, 26, 29, AND/OR 31-74 ARE NOT  
BLANK. ASSUME BLANK.**

Warning error for auto report options (U) specification. You must leave columns 25, 26, 29, and 31 through 74 blank.

Auto report assumes blanks are in columns 25, 26, 29, and 31 through 74.

Leave columns 25, 26, 29, and 31 through 74 blank and resubmit the job.

**† AR076 TOTALING CONSTANT END POSITION > RECORD LENGTH. ALL T-  
TYPE CONSTANTS IGNORED.**

Warning error for output (O) specification. This error occurs because:

1. The length of the total line constants (1 through 9 or R) is longer than the length of the record.
2. The beginning position of the first A-type field is greater than the length of the record.

Auto report drops all total line constants (1 through 9 or R).

Correct the error and resubmit the job.

**† AR077 LEVEL INDICATOR USED ON T-\*AUTO LINE IS UNDEFINED.  
INDICATOR IS DROPPED.**

Warning error for output (O) specification. You must define the control level indicator you used on the T-\*AUTO specification in columns 59 and 60 of the input specifications.

Auto report assumes blanks in these fields.

Enter a control level indicator in columns 59 and 60 of the input specifications and resubmit the job.

**AR079 D-\*AUTO IS CONDITIONED BY MORE THAN 7 AND/OR LINES. ONLY THE FIRST 7 WILL APPLY.**

Warning error for calculation (C) specification. The indicators that condition the line generated by D-\*AUTO are used to condition the generated EXSR calculation used for total rolling.

→ Auto report processed only the first 7 lines of AND or OR conditioning indicators in the calculations.

Remove the excessive AND or OR entries and resubmit the job.

**AR080 INVALID ENTRY IN LISTING OPTION, POSITION 30. BLANK IS ASSUMED.**

Warning error for auto report options (U) specification. You must use B, P, or blank in column 30.

→ Auto report assumes a blank is in column 30.

Use B, P, or blank in column 30 and resubmit the job.

## Index

Term	Reference	Page	Term	Reference	Page
<b>A</b>					
Action programs			Ampersand, edit words	14.3.2.2	14-8
continuity data area	16.4	16-4	AN/OR lines, using	Fig. 7-3	7-6
IMS - RPG II interface areas	16.4	16-4	AND/OR relationships field		
indicators reset	16.1	16-1	calculation specifications		
input message area	16.4	16-4	form	7.2.3	7-5
output message area	16.4	16-4	input format specifications	Fig. 7-3	7-6
program information block	16.4	16-4	form	6.2.8	6-12
required entries, file description			output format specifications		
specifications form	Table 16-2	16-5	form	8.2.4	8-4
restrictions, RPG II features	16.3	16-2	Arithmetic operations		
RPG II specifications forms	Table 16-1	16-2	add (ADD)	7.3.2.1.1	7-8
summary of file organization, access	16.2	16-2	cross-foot (XFOOT)	7.3.2.1.9	7-12
methods and file types	Table 16-3	16-6	description	7.3.2.1	7-7
transaction buffer area	16.4	16-4	divide (DIV)	7.3.2.1.6	7-10
Add (ADD) operation	7.3.2.1.1	7-8	move remainder (MVR)	7.3.2.1.7	7-10
Adding and deleting existing indexed			multiply (MULT)	7.3.2.1.5	7-9
sequential file and deleting transaction			resulting indicator	12.2.4	12-7
buffers	8.2.5	8-4	square root (SQRT)	7.3.2.1.8	7-12
ADDRROUT file, processing	13.6	13-21	subtract (SUB)	7.3.2.1.3	7-9
	Fig. 13-9	13-21	zero and add (Z-ADD)	7.3.2.1.2	7-8
Alphanumeric data format	2.4.1	2-11	zero and subtract (Z-SUB)	7.3.2.1.4	7-9
	Fig. 2-5	2-11	Array file	2.1.2	2-2
Alternate collating sequence				5.2.3	5-6
causing characters to be			Array name	9.2.5	9-4
considered equal	13.11.3.1	13-76		9.2.12	9-7
changing position of	Fig. 13-30	13-76			
characters	13.11.3.3	13-78			
defining	Fig. 13-32	13-78			
field	13.11.2	13-72			
inserting a character between	4.2.7	4-5			
two existing characters	13.11.3.2	13-77			
using	Fig. 13-31	13-77			
	13.11.3	13-76			

Term	Reference	Page	Term	Reference	Page
<b>Arrays</b>			<b>Auto report</b>		
binary search field	4.2.8	4-5	auto report options specifications form	19.5	19-84
calculating totals with arrays	Fig. 13-28	13-70	/COPY statement	19.6	19-89
calculating totals without arrays	Fig. 13-27	13-67	D-*AUTO detail reports	19.3	19-21
definition	13.10.2	13-49	error messages	Appendix G	
definition examples	13.10.2.2	13-53	H-*AUTO page headings	19.2	19-3
field name field	6.3.4	6-14	job control	18.2.2	18-18
file description entries	13.10.2.1	13-50	non-English language factor	19.9	19-107
	Table 13-4	13-51	summary	19.7	19-100
file extension entries	13.10.2.1	13-50	T-*AUTO total reports	19.4	19-54
	Table 13-5	13-52			
formats	13.10.1	13-48	<b>Auto report options specifications form</b>		
	Fig. 13-20	13-48	asterisks suppress	19.5.1.4	19-87
loading, using calculations specifications form	13.10.3.3	13-58	cataloging source program in library file	19.5.1.1	19-84
	Fig. 13-24	13-59	date suppress	19.5.1.3	19-86
loading, using fixed indexes	13.10.3.1	13-55	description	19.5	19-84
	Fig. 13-22	13-56	examples of entries	19.5.2	19-89
loading, using input fields as indexes	13.10.3.2	13-57	form entries	19.5.1	19-84
	Fig. 13-23	13-57	library file name	19.5.1.2	19-86
LOKUP operation	13.10.3.4	13-60	list options	19.5.1.5	19-87
move	7.3.2.2.2	7-16	module name	19.5.1.2	19-86
number of entries	9.2.7	9-5			
processing files	13.10	13-47	<b>AUTRPG job control procedure call statement</b>	18.2.2	18-18
reducing number of statements	13.10.3.6	13-66		Fig. 18-13	18-29
using	13.10.3	13-55	<b>AUTRPG job control procedure call statement</b>	18.2.2	18-18
using to format output records	13.10.3.5	13-64			
	Fig. 13-26	13-65	<b>AUTRPG job control procedure call statement</b>	18.2.2	18-18
writing elements using EXCPT operation	13.10.3.7	13-69			
	Fig. 13-29	13-70			
<b>Asterisk</b>					
calculations	7.2.2	7-4			
	Fig. 7-2	7-4			
catastrophic error messages	Appendix D				
comments	3.2.5	3-5			
edit words	14.3.2.2	14-8			
<b>Auto job control procedure call statement</b>	18.2.2	18-18			
	Fig. 18-9	18-25			
	Fig. 18-11	18-27			
<b>*AUTO</b>					
D-*AUTO	19.3.1.7	19-24			
H-*AUTO	19.2.1.6	19-9			
T-*AUTO	19.4.1.7	19-58			

Term	Reference	Page	Term	Reference	Page
<b>B</b>					
Batch terminals			Block length field	2.3.2	2-9
description	17.1	17-1		5.2.7	5-7
modes	17.3	17-2	Table 5-2		5-12
receive-only, DCT 2000	17.3.1.1	17-5	Blocks, indexed sequential files	2.2.2	2-3
transmit a file, then	Fig. 17-2	17-5	Fig. 2-1		2-4
receive another file,			Branch (GOTO) operation	7.3.2.4.1	7-26
BCS remote terminal	17.3.1.3	17-9	Branching and exit operations		
transmit-only, DCT 1000	Fig. 17-4	17-9	begin internal subroutine		
using	17.3.1.2	17-7	(BEGSR)	7.3.2.4.3	7-27
BCS remote terminal	Fig. 17-3	17-7	branch (GOTO)	7.3.2.4.1	7-26
Begin internal subroutine	17.3.1	17-5	description	7.3.2.4	7-26
(BEGSR) operation			end internal subroutine		
BEGSR operation	17.3.1.3	17-9	(ENDSR)	7.3.2.4.4	7-27
description	Fig. 17-4	17-9	execute internal subroutine		
internal subroutines	17.3.1.2	17-7	(EXSR)	7.3.2.4.5	7-28
Between limits	17.3.1	17-5	exit to linked subroutine		
Binary data format			(EXIT)	7.3.2.4.6	7-29
2.4.2			external subroutine access		
Fig. 2-6			to RPG II program (RLABL)	7.3.2.4.7	7-29
Binary search	4.2.8	4-5	RPG II program to external		
Bit setting operations			subroutine access (ULABL)	7.3.2.4.8	7-30
description	7.3.2.6	7-31	tag (TAG)	7.3.2.4.2	7-26
set bit off (BITOF)	7.3.2.6.2	7-32			
set bit on (BITON)	7.3.2.6.1	7-31			
BITOF operation	7.3.2.6.2	7-32			
BITON operation	7.3.2.6.1	7-31			
*BLANK	7.3.1	7-6			
*BLANKS	7.3.1	7-6			
Blank after field	8.3.4	8-9			
Blank characters	6.2.6.3	6-10			
	Table 6-2	6-10			
Blanks, edit words	14.3.2.2	14-8			
Block header bytes	2.3.1	2-9			
	2.3.2	2-9			

Term	Reference	Page	Term	Reference	Page
<b>C</b>					
C/Z/D field	6.2.6.3 Table 6-2	6-10 6-10	resulting indicator entries	See resulting indicator entries.	
Calculation conditioning indicators specifying	12.4.4 Fig. 12-6	12-16 12-17	summary of operations workstation entries	Table 7-1 13.13.3	7-39 13-85
Calculation entries arithmetic operations	See arithmetic operations.		Call statements, job control procedure	See procedure call statements.	
bit setting operations branching and exit operations	7.3.2.6  See branching and exit operations.	7-31	Catastrophic error messages	Appendix D	
compare and test operations	See compare and test operations.		CCA	17.2 17.4	17-1 17-22
decimal positions factor 1 and factor 2 field length	7.3.3.3 7.3.1 7.3.3.2	7-41 7-6 7-40	CCA name field	4.2.14	4-7
file processing operations	See file processing operations.		CHAIN operation creating a direct file description resulting indicator See also chaining.	13.4.3 7.3.2.8.1 12.2.4	13-13 7-33 12-7
half adjust indicator setting operations	7.3.3.4 7.3.2.5	7-41 7-30	Chained file file designation	2.1.2 5.2.3	2-2 5-5
look-up operations move operations	7.3.2.7 See move operations.	7-32	retrieving record	7.3.2.8.1	7-33
operations result field summary of operations	7.3.2 7.3.3 Table 7-1	7-7 7-40 7-39	Chaining creating a direct file direct files indexed sequential files processing updating an indexed sequential file using CHAIN operation using C1 through C9	13.4.3 13.4.2 Fig. 13-5 13.4.1 13.4 B.7 Fig. 13-3 Fig. 13-4	13-13 13-13 13-14 13-8 13-7 B-33 13-9 13-10
Calculation lines	7.2.2 Fig. 7-2	7-4 7-4	Chaining fields	6.3.6 6.3.6.2 9.2.2	6-17 6-18 9-1
Calculation operations detail total	1.4.1 1.4.2	1-7 1-7	Chaining file, record sequence	9.2.1	9-1
Calculation specifications form calculation entries	See calculation entries.		Chaining subroutine	A.3.2 Fig. A-2	A-8 A-7
conditions entries	See conditions entries.		Channel number field	10.2.3	10-3
description	7.1 Fig. 7-1	7-1 7-2	Channel number translation, skip	Table 8-1	8-6
examples	7.5 Fig. 7-4	7-44 7-45	Channel numbers	Fig. 14-8	14-14
loading an array	13.10.3.3 Fig. 13-24	13-58 13-59			

Term	Reference	Page	Term	Reference	Page
Character fields			Communications control area (CCA)		
C/Z/D field (columns 26, 33, 40)	6.2.6.3	6-10	network definition	17.4	17-22
field (columns 27, 34, 41)	Table 6-2	6-10	using telecommunications	17.2	17-1
	6.2.6.4	6-11	Communications files		
Characters, alternate collating sequence			description	17.2	17-1
causing characters to be considered equal	13.11.3.1	13-76	processing modes	Fig. 17-1	17-3
changing position	Fig. 13-30	13-76		17.3	17-2
inserting character between two existing characters in sequence	13.11.3.3	13-78	COMP operation	7.3.2.3.1	7-23
	Fig. 13-32	13-78		12.2.4	12-7
Characters, edit words	14.3.2.2	14-10	Compare and test operations		
Characters, translating	See file translation.		compare (COMP)	7.3.2.3.1	7-23
Codes, edit	See edit codes.		description	7.3.2.3	7-23
Collating sequence			test bit (TESTB)	7.3.2.3.2	7-24
alternate	See alternate collating sequence.		test numeric (TESTN)	7.3.2.3.3	7-25
definition	13.11	13-72	test zone (TESTZ)	7.3.2.3.4	7-25
field, alternate	4.2.7	4-5	Compilation		
RPG II	13.11.1	13-72	deck arrangement	Fig. 18-16	18-13
Combined edit cards			description	18.1	18-1
description	14.3.1.2	14-6	job control statements	18.2	18-2
listing	Table 14-1	14-6	main storage requirements	18.3	18-40
using	Table 14-2	14-7	Compilation mode field	4.2.1	4-1
Combined file	2.1.1	2-1	Compilation time messages	Appendix D	
	5.2.2	5-3	Compile the program		
Comments field			cards	1.3	1-25
calculations specifications	7.4.4	7-44	workstation	1.2	1-1
description	3.2.5	3-5	Compiler directives	3.2.4	3-5
file extension specifications	9.2.17	9-8	Compiler, main storage requirements	18.3	18-40
Common fields			Compiling source programs		
comments	3.2.5	3-5	IBM System/3	F.2.1	F-1
compiler directives	3.2.4	3-5	SPERRY UNIVAC 9200/9300		
description	3.2	3-2	System	F.3.1	F-6
form type	3.2.3	3-4	Concepts	Section 1	
line number	3.2.2	3-2	Conditions entries, calculation specifications form		
page number	3.2.1	3-2	AND/OR relationship	7.2.3	7-5
program identification	3.2.6	3-6	control level	Fig. 7-3	7-6
			indicators	7.2.1	7-1
			Configuration field	7.2.2	7-3
			Conservation techniques, main storage	11.2.2	11-3
			Constant field	Appendix E	
				8.3.8	8-11

Term	Reference	Page	Term	Reference	Page
Constants, using * PLACE to repeat	14.4.3 Fig. 14-6	14-12 14-13	Conversational reply	See transmit with reception of conversational reply.	
Continuation lines	5.2.18	5-20	Conversion, program requirements	Appendix F	
Continuity data area	See action programs.		Console files		
Control break	1.4.2	1-7	description	5.2.16	5-19
Control break examples	See unwanted control break examples.		interactive data entry	13.16	13-94
Control card specifications form			/COPY modifier statements		
action program	4.2.14	4-7	description	19.6.3	19-91
alternate collating sequence	4.2.7	4-5	file description entries	19.6.3.1	19-92
binary search	4.2.8	4-5	input field entries	19.6.3.2	19-95
CCA name	4.2.13	4-7	/COPY statement		
compilation mode	4.2.1	4-1	description	19.6	19-89
currency sign	4.2.5	4-4	examples of entries	19.6.2	19-91
description	4.1	4-1	form entries	19.6.1	19-90
error analysis dump	4.2.2	4-3	modifier statements	19.6.3	19-91
examples of entries	4.3 Fig. 4-2	4-8 4-8	Cross-foot (XFOOT) operation	7.3.2.1.9	7-12
file translation	4.2.12	4-7	Currency sign	4.2.5	4-4
forms alignment	4.2.10	4-6	Customer records	13.4.1	13-8
generate debug code	4.2.4	4-4	Cylinder overflow space percentage	5.2.22	5-28
indicator initialization	4.2.11	4-6			
inverted print	4.2.6	4-5			
operator control	4.2.3	4-3			
program identification	4.2.15	4-7			
sign handling	4.2.9	4-6			
subroutine	4.2.14	4-7			
Control level field					
calculations specifications form	7.2.1	7-1			
input format specifications form	6.3.5	6-15			
rules	6.3.5	6-15			
Control level indicators					
description	12.2.2	12-3			
unwanted control breaks	See unwanted control break examples.				
using	12.4.2	12-15			
Control level zero indicator	12.3.2	12-12			
Controlling RPG II					
general	12.1	12-1			
indicators defined on forms	12.2	12-1			
indicators not defined on forms	12.3	12-11			



Term	Reference	Page	Term	Reference	Page
<b>D</b>					
Data, entering	See entering data.		DCT 2000 Data Communications Terminal	17.3.1.1 Fig. 17-2	17-5 17-5
Data files, designation	2.1.2	2-2	Debug code field, generate	4.2.4	4-4
Data format field			DEBUG operation	7.3.2.8.8 C.2 Fig. C-1	7-36 C-2 C-4
file extension	9.2.9 9.2.14	9-6 9-7	Debug the program		
input	6.3.1	6-12	cards	1.3	1-25
output	8.3.7	8-10	workstation	1.2	1-1
Data formats			Decimal points, edit words	14.3.2.2	14-10
alphanumeric	2.4.1 Fig. 2-5	2-11 2-11	Decimal positions, edit codes	14.3.1.2	14-6
binary	2.4.2 Fig. 2-6	2-11 2-11	Decimal positions field		
packed numeric	2.4.3 Fig. 2-7	2-12 2-12	calculation specifications	7.3.3.3	7-41
unpacked numeric	2.4.4 Fig. 2-8	2-12 2-12	file extension specifications	9.2.10 9.2.15	9-6 9-8
Data structures	13.15	13-91	input format specifications	6.3.3	6-14
Date format, system	4.2.6	4-5	Deck arrangement for compiling, link editing, and execution	Fig. 18-18	18-41
D-*AUTO detail reports			Default options, standard	Fig. 18-1 Fig. 18-2	18-8 18-9
description	19.3	19-21	Deleting records from an existing indexed sequential file	8.2.5	8-4
examples of entries	19.3.2.7	19-52	Demand files		
field description and control entries	19.3.2	19-25	designation	2.1.2 5.2.3	2-2 5-5
field description, prints field and column heading	19.3.2.1	19-28	processing	13.7.1 13.7.3.1 Fig. 13-10	13-22 13-23 13-23
field description, prints field next to total	19.3.2.6	19-49	READ operation	13.7.3	13-22
field description, prints heading	19.3.2.2	19-34	SETLL operation	Fig. 13-10 13.7.3.2 Fig. 13-11	13-23 13-25 13-26
field description, prints heading next to total	19.3.2.5	19-47	Detail logic cycle description	A.1 Fig. A-1 A.3 Fig. A-2	A-1 A-2 A-6 A-7
field description, prints numeric field, column heading and accumulates totals	19.3.2.3	19-36	subroutines		
field description, prints second and third column heading line	19.3.2.4	19-45	Detail logic subroutines		
output file identification and control entries	19.5.1	19-84	chaining	A.3.2	A-8
DCT 500 Data Communications Terminal			look-ahead	A.3.4	A-8
formatting a printer and entering data	Fig. 17-7	17-18	matching fields	A.3.1	A-6
transmit with reception of conversation reply	17.3.2.2.1 Fig. 17-8	17-19 17-19	overflow	A.3.3	A-8
using	17.3.2.2	17-17	Detail reports	19.3	19-21
DCT 1000 Data Communications Terminal	17.3.1.2 Fig. 17-3	17-7 17-7			

Term	Reference	Page	Term	Reference	Page
Detail time	1.4.1	1-7			
Device field	5.2.16 Table 5-5	5-17 5-18			
Devices					
block length ranges	5.2.7	5-8			
record length ranges	5.2.8	5-10			
Digit, C/Z/D field	6.2.6.3	6-10			
Direct data interface (DDI)	17.4	17-22			
Direct files					
chaining	13.4.2 Fig. 13-5	13-13 13-14			
creating with CHAIN operation	13.4.3	13-13			
description	2.2.3 Fig. 2-2	2-3 2-6			
Display (DSPLY) operation	7.3.2.8.6 C.3	7-36 C-3			
Display file	2.1.1 5.2.2	2-1 5-3			
DIV operation	7.3.2.1.6	7-10			
Divide (DIV) operation	7.3.2.1.6	7-10			
Dollar sign, edit words	14.3.2.2	14-8			
DSPLY operation	7.3.2.8.6 C.3	7-36 C-3			
Dumps					
error analysis	See error analysis dump.				
unformatted	See unformatted dump.				
D?TE field name	14.4.1	14-10			
			<b>E</b>		
			EBCDIC collating sequence	4.2.7 13.11.1 Table 13-6	4-5 13-72 13-73
			Edit codes		
			combined	14.3.1.2 Table 14-1 Table 14-2	14-6 14-6 14-7
			description	14.3.1	14-5
			field	8.3.3	8-9
			simple	14.3.1.1	14-5
			Edit word field	8.3.9	8-11
			Edit words		
			description	14.3.2	14-6
			examples	Fig. 14-4	14-11
			format	14.3.2.1 Fig. 14-3	14-8 14-9
			using	14.3.2.2	14-8
			Editing		
			description	14.3	14-3
			edit codes	14.3.1	14-5
			edit words	14.3.2	14-6
			/EJECT compiler directive	3.2.4	3-5
			Elements		
			adding to existing table	13.9.3.4 Fig. 13-19	13-46 13-47
			array formats	13.10.1 Fig. 13-20	13-48 13-48
			first element positioning	13.9.1	13-33
			writing using EXCPT	13.10.3.7 Fig. 13-29	13-69 13-70
			End internal subroutine (ENDSR) operation	7.3.2.4.4 15.2	7-27 15-2
			End of file	5.2.4	5-6
			End position in output record field	8.3.5	8-10
			ENDSR operation		
			description	7.3.2.4.4	7-27
			internal subroutines	15.2	15-2
			Entering data		
			teletypewriter and DCT 500	Fig. 17-7	17-18
			UNISCOPE 100/200	Fig. 17-5	17-13

Term	Reference	Page	Term	Reference	Page
Entries, form	See examples of form entries.		Examples, array processing		
Entries per record, number	9.2.6	9-5	array definition	Fig. 13-21	13-53
Entries to avoid incorrect output, unwanted control break example	Fig. 12-3	12-4	calculating totals with arrays	Fig. 13-28	13-68
Entry length	9.2.8	9-5	calculating totals without arrays	Fig. 13-27	13-67
	9.2.13	9-7	loading an array using fixed indexes	Fig. 13-22	13-56
Error analysis dump			loading an array using input fields as indexes	Fig. 13-23	13-57
description	C.7	C-7	loading an array with calculations specifications form	Fig. 13-24	13-59
*ERROR	C.7.1.2	C-8	LOKUP operation	Fig. 13-25	13-62
example	Fig. C-3	C-12	using to format output records	Fig. 13-26	13-65
field	4.2.2	4-3	writing elements using EXCPT operation	Fig. 13-29	13-70
FLDS/CONSTS/TABLES/ARRAYS	C.7.1.5	C-10			
LINKAGE VECTOR	C.7.1.4	C-9	Examples, auto report		
option	18.3	18-40	/COPY statement	19.6.2	19-91
RECORD	C.7.1.3	C-9	D-*AUTO	19.3.2.7	19-52
REG SAVE AREA	C.7.1.1	C-8	H-*AUTO	19.2.2.3	19-20
			options specifications form	19.5.2	19-89
*ERROR field			T-*AUTO	19.4.2.7	19-82
description	C.6	C-7			
	Fig. C-2	C-7	Examples, file processing		
error analysis dump	C.7.1.2	C-8	ADDROUT file	Fig. 13-9	13-21
unformatted dump	C.8.1	C-10	chaining, CHAIN operation	Fig. 13-3	13-9
	C.8.1.1.2	C-11	chaining, C1 through C9	Fig. 13-4	13-10
			chaining, direct file	Fig. 13-5	13-14
Error indicator, permanent	11.2.7	11-4	controlling record selection during multifile processing	Fig. 13-12	13-30
Error messages			demand file, READ operation	Fig. 13-10	13-23
auto report	Appendix G		demand file, SETLL operation	Fig. 13-11	13-26
compilation	Appendix D		indexed sequential file between limits	Fig. 13-8	13-19
Error processing, telecommunications	17.6	17-24	indexed sequential file randomly using record address file	Fig. 13-7	13-18
Errors			matching records from secondary file	Fig. 13-13	13-32
compilation	1.2	1-1	workstations	13.13	13-82
keypunching	1.3	1-25	WORKSTN file	13.13.1	13-84
Examples, alternate collating sequence					
causing characters to be considered equal	Fig. 13-30	13-76	Examples, file translation		
changing position of characters	Fig. 13-32	13-78	translating characters for more than one but not all files	Fig. 13-34	13-82
inserting a character between two existing characters	Fig. 13-31	13-77	translating characters in a file	Fig. 13-33	13-82

Term	Reference	Page	Term	Reference	Page
Examples, form entries			EXEC statement	18.2	18-2
calculations specifications	7.5	7-44		18.2.3	18-31
	Fig. 7-4	7-45	Execute internal subroutine (EXSR) operation	7.3.2.4.5	7-28
control card specifications	4.3	4-8	Executing IBM System/3 program	F.2.2	F-5
	Fig. 4-2	4-8	Execution		
file description specifications	5.3	5-30	deck arrangement	Fig. 18-18	18-41
	Fig. 5-2	5-31	description	18.1	18-1
file extension specifications	9.3	9-8	job control statements	18.2	18-2
	Fig. 9-2	9-9	EXIT operation		
indexed sequential file creation	Fig. B-8	B-17	description	7.3.2.4.6	7-29
	Fig. B-20	B-41	external subroutines	15.3	15-4
indexed sequential file processing	Fig. B-11	B-22	Exit operations	7.3.2.4	7-26
	Fig. B-14	B-29	Exit to linked subroutine (EXIT) operation	7.3.2.4.6	7-29
indexed sequential file updating	Fig. B-17	B-35		15.3	15-4
input format specifications	6.4	6-20	Expansion, edit word format	14.3.2.1	14-8
	Fig. 6-5	6-21	EXSR operation		
line counter specifications	10.4	10-4	description	7.3.2.4.5	7-28
	Fig. 10-2	10-4	internal subroutines	15.2	15-1
sequential file processing	Fig. B-4	B-8	Extension or line counter code field	5.2.15	5-17
telecommunications	11.3	11-6	External indicators	12.3.1	12-11
	Fig. 11-2	11-7	External subroutine access operations		
Examples, interactive data entry			RPG II program to (ULABL)	7.3.2.4.8	7-30
adding MIRAM records	B.10.3	B-65	to RPG II program (RLABL)	7.3.2.4.7	7-29
	Fig. B-27	B-65	External subroutines		
creating MIRAM files	B.10.1	B-58	description	15.3	15-4
	Fig. B-25	B-58	specifying	15.3.1	15-5
four-column prompt screen	Fig. 13-44	13-94d		Fig. 15-2	15-5
function key 5 prompt screen	Fig. 13-39	13-94			
one-column prompt screen	Fig. 13-40	13-94b			
three-column prompt screen	Fig. 13-43	13-94c			
two-column prompt screen	Fig. 13-41	13-94b			
	Fig. 13-42	13-94c			
updating MIRAM records	B.10.2	B-61			
	Fig. B-26	B-61			
Examples, programming	Appendix B				
Examples, table processing					
adding elements	Fig. 13-19	13-47			
LOKUP operation	Fig. 13-16	13-41			
modifying a table	Fig. 13-18	13-46			
table definition	Fig. 13-15	13-39			
Exception lines (EXCPT) operation	7.3.2.8.7	7-36			
EXCPT operation					
description	7.3.2.8.7	7-36			
using	C.4	C-3			
writing array elements	13.10.3.7	13-69			
	Fig. 13-29	13-70			

Term	Reference	Page	Term	Reference	Page
<b>F</b>			Fields		
Factor 1 and factor 2 fields	7.3.1	7-6	chaining	6.3.6.2	6-18
Fetch overflow	8.2.3	8-3	common	See common fields.	
Field description and control entries	12.2.5	12-10	control card specifications		
blank after	Fig. 12-5	12-10	form	4.2	4-1
constant			look-ahead	13.8	13-28
data format			matching	13.8.1	13-28
description				See matching fields.	
edit codes			File, last	11.2.9	11-5
edit word			File, primary	13.3.1	13-2
end position in output record			File addition field	5.2.21	5-27
field name	8.3.5	8-10	Table 5-8	5-28	
output indicators - fields	8.3.5	8-10	File characteristics	2.1	2-1
	8.3.1	8-8	File conditioners	5.2.25	5-30
Field description entries			File conditioning indicators	12.4.1	12-15
control level	6.3.5	6-15	File description specifications		
data format	6.3.1	6-12	form		
decimal position	6.3.3	6-14	array definition	13.10.2.1	13-50
description	6.3	6-12	Table 13-4	13-51	
field indicators	6.3.8	6-20	block length	5.2.7	5-7
field location	6.3.2	6-13	change keys	5.2.18.6	5-26
field name	6.3.4	6-14	cylinder overflow space		
field record relation	6.3.7	6-19	percentage	5.2.22	5-28
matching fields or chaining fields	6.3.6	6-17	description	5.1	5-1
Field indicator	12.2.3	12-7	Fig. 5-1	5-2	
Field length field	7.3.3.3	7-41	device	5.2.16	5-17
Field location field	6.3.2	6-13	Table 5-5	5-18	
Field name field			duplicate record	5.2.18.5	5-25
input	6.3.4	6-14	end of file	5.2.4	5-6
output	8.3.2	8-8	examples of entries	Fig. 5-2	5-31
Field names, special	See special field names.		extension or line counter		
Field record relation	6.3.7	6-19	code	5.2.15	5-17
Field record relation indicators	12.4.3	12-15	file addition	5.2.21	5-27
			Table 5-8	5-28	
			file conditioners	5.2.25	5-30
			file designation	5.2.3	5-4
			file format	5.2.6	5-7
			file name	5.2.1	5-1
			file organization	5.2.12	5-14
			file processing mode	5.2.9	5-13
			file type	5.2.2	5-3
			file use	Table 5-1	5-4
			key length (MIRAM files)	5.2.18.4	5-25
			key or record address		
			field length	5.2.10	5-13
			labels	5.2.17	5-17

Term	Reference	Page	Term	Reference	Page
File description specifications form (cont)			table definition	13.9.2.1	13-36
name of label exit or name of user device routine	5.2.19	5-26	Table 13-3		13-38
number of bytes in main storage to be reserved for ISAM index	5.2.20	5-27	9.2.5		9-4
overflow indicator	5.2.13	5-16	9.2.12		9-7
record address type	5.2.11	5-13	9.2.4		9-3
sequence	5.2.5	5-6	File format		
summary of file format block length, and record length entries	Table 5-2	5-12	allowable	Table 2-1	2-7
summary of file processing mode, record address type, and file organization entries	Table 5-3	5-15	description	2.3	2-7
summary of required entries, action programs	Table 16-2	16-5	field	5.2.6	5-7
table definition	13.9.2.1	13-36	Table 5-2		5-12
tape rewind option	5.2.24	5-29	fixed-length records	2.3.1	2-8
workstation entries	13.13.1	13-84	variable-length records	Fig. 2-3	2-8
File designation	2.1.2	2-2		2.3.2	2-9
File designation field	5.2.3	5-4	Fig. 2-4		2-10
File extension specifications form			File identifications, output	See output file identification and control entries.	
array definition	13.10.2.1	13-50	File, MIRAM		
comments	Table 13-5	13-52	adding records	B.10.3	B-65
data format	9.2.17	9-8	creating files	Fig. B-27	B-65
decimal positions	9.2.9	9-6	updating records	B.10.1	B-58
description	9.2.14	9-7		Fig. B-25	B-58
example of entries	9.2.10	9-6		B.10.2	B-61
from file name	9.2.15	9-8		Fig. B-26	B-61
length of entry	9.1	9-1	File name field		
number of entries per record	Fig. 9-1	9-2	file description specifications form	5.2.1	5-1
number of entries per table or array	9.3	9-8	input format specifications form	6.2.1	6-3
number of the chaining field	Fig. 9-2	9-9	line counter specifications form	10.2.1	10-1
record sequence of chaining file	9.2.3	9-3	output format specifications form	8.2.1	8-1
sequence	9.2.8	9-5	telecommunications specifications form	11.2.1	11-1
summary of from and to file name entries	9.2.13	9-7	File name fields, file extension specifications form		
	9.2.6	9-5	from	9.2.3	9-3
	9.2.7	9-5	to	Table 9-1	9-4
	9.2.2	9-1		9.2.4	9-3
	9.2.1	9-1		Table 9-1	9-4
	9.2.11	9-6	File organization		
	9.2.16	9-8	action program, summary	Table 16-3	16-6
	Table 9-1	9-4	description	2.2	2-2
			direct	2.2.3	2-3
			field	Fig. 2-2	2-6
			indexed sequential record retrieval and processing methods	5.2.12	5-14
			sequential	Table 5-3	5-15
				2.2.2	2-3
				Table 13-1	13-1
				2.2.1	2-3

Term	Reference	Page	Term	Reference	Page
File processing			File processing operations		
ADDROUT file	13.6	13-21	debug (DEBUG)	7.3.2.8.8	7-36
alternate collating sequence	Fig. 13-9	13-21	description	7.3.2.8	7-33
arrays	See alternate collating sequence.		display (DISPLY)	7.3.2.8.6	7-35
chaining	See arrays.		exception lines (EXCPT)	7.3.2.8.7	7-36
controlling record selection during multifile processing	See chaining.		next workstation input (NEXT)	7.3.2.8.9	7-37
demand file	13.8.3.1	13-29	override normal record selection (FORCE)	7.3.2.8.5	7-35
demand file, READ	Fig. 13-12	13-30	read record (READ)	7.3.2.8.3	7-34
demand file, SETLL	13.15	13-91	retrieve record from a chained file (CHAIN)	7.3.2.8.1	7-33
demand files and READ operation	13.7	13-22	select key structure (SETK)	7.3.2.8.2	7-34
description	13.7.1	13-22	set lower limits (SETLL)	7.3.2.8.4	7-35
file organization, record retrieval	13.7.3.1	13-23	File sharing	5.2.18	5-20
FORCE operation	Fig. 13-10	13-23		Table 5-7	5-25
interactive data entry	13.7.3.2	13-25	File translation		
look-ahead fields	Fig. 13-11	13-26	control card field	4.2.12	4-7
matching record indicator	13.7.3	13-22	defining table	13.12.1	13-78
matching records	13.1	13-1	description	13.12	13-78
matching records, secondary files	Table 13-1	13-1	translating characters for more than one but not all files	13.12.2.2	13-82
mode	13.8	13-28		Fig. 13-34	13-82
modes, remote terminals	13.8.2	13-28	translating characters in a file	13.12.2.1	13-81
multikey MIRAM files	13.16	13-94		Fig. 13-33	13-82
primary file	13.8	13-28	using table	13.12.2	13-81
READ operation	13.8.1	13-28	File type field	5.2.2	5-3
record address	13.3.4	13-7	Files		
record selection	13.3	13-2	demand	See demand files.	
sequential	13.8.3.2	13-32	direct	See direct files.	
specifying matching fields	Fig. 13-13	13-32	indexed sequential	See indexed sequential files.	
tables	Table 5-3	5-15	line counter	See line counter files.	
translation	17.3	17-2	sequential	See sequential files.	
workstation	13.4	13-7	telecommunications	17.2	17-1
WORKSTN file	13.3.1	13-2	types	Fig. 17-1	17-3
See also examples, file processing.	13.7.2	13-22	use	2.1.1	2-1
	See record address file processing.		See also file processing.	Table 5-1	5-4
	13.3.3	13-3	First page forms alignment	4.2.10	4-6
	Fig. 13-2	13-4	First page indicator	12.3.2	12-12
	13.2	13-2		Fig. A-1	A-2
	13.3.2	13-2		A.2	A-4
	Fig. 13-1	13-3			
	See tables.				
	See file translation.				
	13.13	13-82			
	13.13.1	13-84			





Term	Reference	Page	Term	Reference	Page
IMS - RPG II action programs	See action programs.		permanent error record identifying	11.2.7 6.2.5 12.2.1 Fig. 12-1	11-4 6-4 12-1 12-2
IMS - RPG II interface areas	See action programs.		reset resulting	See action programs.	
Incorporating subroutines in your program	15.1	15-1	reset resulting, form entries	12.2.4	12-7
Indexed sequential files			setting on and off	See resulting indicator entries.	
adding records	8.2.5	8-4	setting on via system console	Table 12-3	12-22
chaining	13.4.1 Fig. 13-3	13-8 13-9	setting on via system console use, summary	12.6 Table 12-2	12-23 12-21
description	2.2.2 Fig. 2-1	2-3 2-4	Input deck sequence, job control statements	18.2.3 Fig. 18-16	18-31 18-37
processing between limits	13.5.2 Fig. 13-8	13-18 13-19	Input fields, loading an array	13.10.3.2 Fig. 13-23	13-57 13-57
processing randomly	13.5.1 Fig. 13-7	13-17 13-18	Input file	2.1.1 5.2.2	2-1 5-3
Indexes			Input format specifications form		
fixed, loading array	13.10.3.1 Fig. 13-22	13-55 13-56	/COPY statement description	19.6.3.2 6.1 Fig. 6-1	19-95 6-1 6-2
input fields, loading array	13.10.3.2 Fig. 13-23	13-57 13-57	examples	6.4 Fig. 6-5	6-20 6-21
LOKUP	13.10.3.4	13-60	loading an array using fixed indexes	13.10.3.1 Fig. 13-22	13-55 13-56
using arrays	13.10.3	13-55	loading an array using input fields as indexes	13.10.3.2 Fig. 13-23	13-57 13-57
Indicator setting operations			record identification	See record identification entries.	
description	7.3.2.5	7-30	workstation entries	13.13.2	13-85
set off (SETOF)	7.3.2.5.2	7-30	Input message area	See action programs.	
set on (SETON)	7.3.2.5.1	7-30	Input records, unwanted control break example	Fig. 12-2	12-3
Indicators			Inquiry request, IR	12.6.2	12-26
calculation conditioning	12.4.4 Fig. 12-6	12-16 12-17	Integrated communications access method (ICAM)		
calculations specifications from field	7.2.2	7-3	error processing	17.6	17-24
control level	12.2.2 12.4.2	12-3 12-15	network definition	17.4	17-22
definition, summary	Table 12-1	12-20	program execution	17.5	17-24
external	12.3.1	12-11	Interactive data entry	13.16	13-94
field	6.3.8 12.2.3	6-20 12-7			
field record relation	12.4.3	12-15			
file conditioning	12.4.1	12-15			
initialization	4.2.11	4-6			
internal	12.3.2	12-12			
matching record	13.3.4	13-7			
output	See output indicators.				
output conditioning	12.4.5 Fig. 12-7	12-17 12-19			
overflow	See overflow indicator.				



Term	Reference	Page	Term	Reference	Page
<b>L</b>					
Label exit	5.2.19	5-26	Link editing		
Labels field	5.2.17	5-19	deck arrangement	18.2.4	18-40
Last file field	11.2.9	11-5	description	18.1	18-1
Last record indicator (LR)	5.2.4	5-6	job control statements	18.2	18-2
	12.3.2	12-14	Linkage editor	1.3	1-25
	Fig. A-1	A-2	18.1	18-1	
	A.2	A-4	Linkage registers	5.2.19	5-26
Length of entry field	9.2.8	9-5	LINKAGE VECTOR, error		
	9.2.13	9-7	analysis dump	C.7.1.4	C-9
Limits			Linked subroutine, exit	7.3.2.4.6	7-29
demand file processing	13.7.3.2	13-25	Load module	1.3	1-25
indexed sequential file			Logic cycle	See detail	
processing	13.5.2	13-18	logic cycle.		
	Fig. 13-8	13-19	LOKUP operation		
Line counter code	5.2.15	5-17	arrays	13.10.3.4	13-60
Line counter files			Fig. 13-25	13-62	
description	14.5	14-13	binary search field	4.2.8	4-5
printing	14.5.2	14-14	description	7.3.2.7.1	7-32
specifying	Fig. 14-7	14-14	tables	13.9.3	13-41
specifying lines associated				13.9.3.1	13-41
with channel numbers	Fig. 14-8	14-14		Fig. 13-16	13-41
using	14.5.1	14-13	Look-ahead fields	13.8	13-28
Line counter specifications form			13.8.1	13-28	
channel number	10.2.3	10-3	Look-ahead subroutine	A.3.4	A-8
description	10.1	10-1	Fig. A-2	A-7	
	Fig. 10-1	10-2	Look-up operations		
examples of entries	10.4	10-4	description	7.3.2.7	7-32
	Fig. 10-2	10-4	look-up (LOKUP)	7.3.2.7.1	7-32
file name	10.2.1	10-1	resulting indicator	12.2.4	12-7
for IBM System/3 mode	10.3	10-3	Lower limits, set	7.3.2.8.4	7-35
line number	10.2.2	10-3			
Line insertion example	Fig. 3-1	3-3			
Line number field					
description	3.2.2	3-2			
line counter specifications					
form	10.2.2	10-3			
line insertion example	Fig. 3-1	3-3			
Link-edit errors	1.3	1-25			
Link-edit the program	1.3	1-25			
	Fig. 1-1	1-2			

Term	Reference	Page	Term	Reference	Page
<b>M</b>					
Main storage			MIRAM files, manipulating		
conservation techniques	Appendix E		adding records	B.10.3	B-65
requirements	18.3	18-40	creating files	Fig. B-27	B-65
reserving for ISAM index	5.2.20	5-27	updating records	B.10.1	B-58
				Fig. B-25	B-58
Matching field subroutine	A.3.1	A-6		B.10.2	B-61
	Fig. A-2	A-7		Fig. B-26	B-61
Matching fields			MLHZO operation	7.3.2.2.5	7-21
description	6.3.6.1	6-17	MLLZO operation	7.3.2.2.4	7-20
file record relation	12.4.3	12-15	Mode		
sequence	5.2.5	5-6	compilation	4.2.1	4-1
specifying	13.3.2	13-2	file processing	5.2.8	5-10
	Fig. 13-1	13-3	operator control	C.5	C-5
Matching fields or chaining fields field	6.3.6	6-17	Modes, file processing	17.3	17-2
Matching record indicator (MR)	12.3.2	12-14	Modifier statements		
	13.3.4	13-7	description	19.6.3	19-91
Matching records, file processing			file description specifications	19.6.3.1	19-92
description	13.3	13-2	input specifications	19.6.3.2	19-95
indicator	13.3.4	13-7	Move array (MOVEA) operation	7.3.2.2.2	7-16
primary file	13.3.1	13-2	Move high zone to high zone (MHHZO) operation	7.3.2.2.7	7-22
record selection	13.3.3	13-3	Move high zone to low zone (MHLZO) operation	7.3.2.2.6	7-22
	Fig. 13-2	13-4	Move left (MOVEL) operation	7.3.2.2.3	7-17
secondary files	13.8.3.2	13-32	Move low zone to high zone (MLHZO) operation	7.3.2.2.5	7-21
	Fig. 13-13	13-32	Move low zone to low zone (MLLZO) operation	7.3.2.2.4	7-20
specifying matching fields	13.3.2	13-2	Move operations		
	Fig. 13-1	13-3	description	7.3.2.2	7-12
Messages			move (MOVE)	7.3.2.2.1	7-13
compilation time	C.5.1	C-5	move array (MOVEA)	7.3.2.2.2	7-16
	Appendix D		move high zone to high zone (MHHZO)	7.3.2.2.7	7-22
displayed for operator control	C.5.3	C-6	move high zone to low zone (MHLZO)	7.3.2.2.6	7-22
execution time	C.5.2	C-5	move left (MOVEL)	7.3.2.2.3	7-17
Message text format	19.9	19-107	move low zone to high zone (MLHZO)	7.3.2.2.5	7-21
MHHZO operation	7.3.2.2.7	7-22	move low zone to low zone (MLLZO)	7.3.2.2.4	7-20
MHLZO operation	7.3.2.2.6	7-22	Move remainder (MVR) operation	7.3.2.1.7	7-10
MINUS field	6.3.8	6-20			
Minus sign, edit word	14.3.2.2	14-10			
MIRAM files	2.2.4	2-7			
	18.2.3	18-31			





Term	Reference	Page	Term	Reference	Page
Printing techniques			Program logic	1.4	1-27
description	14.1	14-1	Program testing aids		
edit codes	14.3.1	14-5	DEBUG operation	C.2	C-2
edit words	14.3.2	14-6	description	Fig. C-1	C-4
editing	14.3	14-3	DSPLY operation	C.1	C-1
line counter files	14.5	14-13	error analysis dump	C.3	C-3
printer format chart	14.2	14-1	*ERROR field	C.7	C-7
special field names	Fig. 14-1	14-2	EXCPT operation	Fig. C-3	C-12
	Fig. 14-2	14-4	operator control	C.6	C-7
	14.4	14-10	unformatted dump	Fig. C-2	C-7
Procedure call statements, job control			C.5	C.5	C-5
AUTO statement	18.2.2	18-18	C.5	C.5	C-5
	Fig. 18-9	18-25	C.8	C.8	C-10
	Fig. 18-11	18-27	Fig. C-4	Fig. C-4	C-25
AUTORPG statement	18.2.2	18-18	Programming at workstation	13.13	13-82
	Fig. 18-13	18-29		B.9.3	B-57
AUTORPGL statement	18.2.2	18-18	Programming examples	Appendix B	
AUTORPGLG statement	18.2.2	18-18	Programs		
defining keyword parameters	18.2.1.2	18-10	compiling	1.2	1-4
	Fig. 18-3	18-10		18.1	18-1
description	18.2.1	18-2	controlling RPG II	12.1	12-1
formats	18.2.1	18-3	debugging	1.2	1-5
RPG statement	18.2.1.1	18-8	executing	1.2	1-2
	Fig. 18-1	18-8		1.3	1-25
	Fig. 18-2	18-9		18.1	18-1
	Fig. 18-4	18-11	incorporating subroutines	See incorporating subroutines in your program.	
RPGL statement	18.2.1.3	18-12		1.2	1-1
	Fig. 18-5	18-12		1.3	1-25
	Fig. 18-6	18-13		18.1	18-1
RPLG statement	18.2.1.4	18-14	link-edit	1.2	1-1
	Fig. 18-7	18-14		1.3	1-25
	Fig. 18-8	18-16	requirements	1.2	1-1
Processing, file	See file processing.			1.3	1-25
Program check island code	4.2.9	4-6	type and number of specifications forms required	3.1	3-1
Program conversion requirements	Appendix F		writing	1.2	1-3
Program execution telecommunications	17.5	17-24		1.3	1-25
Program identification	3.2.6	3-6	Prompt screens		
	4.2.15	4-7	four-column, 70 fields	Fig. 13-44	13-94d
Program information block	See action programs.		function key 5	Fig. 13-39	13-94
			one-column, 23 fields	Fig. 13-40	13-94b
			three-column, 47 fields	Fig. 13-43	13-94c
			two-column, 24 fields	Fig. 13-41	13-94b
			two-column, self-adjusting fields	Fig. 13-42	13-94c

Term	Reference	Page	Term	Reference	Page
<b>R</b>					
Randomly processing an indexed sequential file	13.5.1 Fig. 13-7	13-17 13-18	Record identification entries, input format specifications form		
Randomly retrieving records	13.4.2 Fig. 13-5	13-13 13-14	AND/OR	6.2.8	6-12
READ operation			file name	6.2.1	6-3
demand files	13.7.3 Fig. 13-10	13-22 13-23	number	6.2.3	6-3
demand files using SETLL description	13.7.3.2 7.3.2.8.3	13-25 7-34	optional	6.2.4	6-4
Receive a file, then transmit a file mode	17.3	17-4	providing for undefined record types	Fig. 6-4	6-9
Receive only mode description using DCT 2000	17.3 17.3.1.1 Fig. 17-2	17-3 17-5 17-5	record identification codes	6.2.6	6-8
RECORD, error analysis dump	C.7.1.3	C-9	record identifying indicator	6.2.5	6-4
Record address field length	5.2.10	5-13	sequence	6.2.2	6-3
Record address file	2.1.2 5.2.3	2-2 5-5	stacker select	6.2.7	6-11
Record address file processing indexed sequential file between limits	13.5.2 Fig. 13-8	13-18 13-19	summary of sequence, number, and optional field entries	Table 6-1	6-4
indexed sequential file randomly	13.5.1 Fig. 13-7	13-17 13-18	Record identifying indicator	12.2.1 Fig. 12-1	12-1 12-2
Record address type	5.2.11 Table 5-3	5-13 5-15	Record identifying indicator field	6.2.5	6-4
Record identification codes field			Record length field	2.3.2 5.2.8 Table 5-2	2-9 5-10 5-12
C/Z/D characters	6.2.6.3 6.2.6.4	6-10 6-11	Record numbers, relative	13.4.2	13-13
description	6.2.6	6-8	Record position ranges, UNISCOPE 100/200	Table 17-1	17-12
interpretation of &, -, and blanks	Table 6-2	6-10	Record retrieval chaining	13.4.2 Fig. 13-5	13-13 13-14
N=NOT	6.2.6.2	6-10	file processing	Table 13-1	13-1
position	6.2.6.1	6-10	Record selection		
providing for undefined record types	Fig. 6-4	6-9	controlling during multifile processing	13.8.3.1 Fig. 13-12	13-29 13-30
			matching	13.3.3 Fig. 13-2	13-3 13-4
			override normal	7.3.2.8.5	7-35
			Record sequence of chaining file field	9.2.1	9-1
			Record types, undefined	Fig. 6-4	6-9



Term	Reference	Page	Term	Reference	Page
<b>Records</b>					
adding to indexed sequential file	8.2.5	8-4	Retrieve record from a chained file (CHAIN) operation	7.3.2.8.1	7-33
alternate collating sequence	13.11.2	13-72	Rewinding tape	5.2.24	5-29
customer	13.4.1	13-8	RLABL operation		
file format	2.3	2-7	description	7.3.2.4.7	7-29
file translation table	13.12.1	13-76	external subroutines	15.3	15-4
fixed-length	See fixed-length records.		RPG II		
input	See input records.		controlling	See controlling RPG II.	
matching	See matching records.		cards	1.3	1-25
output	See output records.		workstation	Fig. 1-2	1-24
output indicators	8.2.8	8-7		1.2	1-1
variable-length	See variable-length records.			Fig. 1-1	1-23
REFER operation	7.3.2.8.1A	7-34	RPG II - IMS action programs	See action programs.	
REG SAVE AREA	C.7.1.1	C-8	RPG II program to external subroutine access (ULABL) operation	7.3.2.4.8	7-30
Registers, linkage	5.2.19	5-26	RPG job control procedure call statement		
Relative record numbers	13.4.2	13-13	defining keyword parameters	18.2.1.2	18-10
Remote device field	11.2.10	11-5	description	Fig. 18-3	18-10
	Table 11-1	11-6	format	18.2	18-2
Remote terminal field	11.2.6	11-4	job control statements generated, nonstandard options	18.2.1	18-2
	Table 11-1	11-6	job control statements generated, standard default options	18.2.1	18-3
Remote terminals			job control statements generated, nonstandard options	Fig. 18-4	18-11
batch	See batch terminals.		job control statements generated, standard default options	Fig. 18-2	18-9
description	17.1	17-1	RPGL job control procedure call statement		
field	11.2.6	11-4	description	18.2.1	18-3
file processing modes	17.3	17-2	format	18.2.1	18-3
interactive	See interactive terminals.		job control statements generated	Fig. 18-6	18-13
Reports, special field names	14.4	14-10	using	18.2.1.3	18-12
Restrictions, RPG II features	See action programs.		using	Fig. 18-5	18-12
Result field	7.3.3	7-40	RPGLG job control procedure call statements		
Resulting indicator	12.2.4	12-7	description	18.2.1	18-2
Resulting indicator entries			format	18.2.1	18-3
comments	7.4.4	7-44	job control statements generated	Fig. 18-8	18-16
description	7.4	7-42	using	18.2.1.4	18-14
0, 1 = 2, or equal	7.4.3	7-43		Fig. 18-7	18-14
-, 1 < 2, or low	7.4.2	7-43			
+, 1 > 2, or high	7.4.1	7-42			

Term	Reference	Page	Term	Reference	Page
<b>S</b>					
Sample job control stream, auto report	19.8	19-107	SETK operation	7.3.2.8.2	7-34
Sample program, workstation	13.13.6	13-86	Set lower limits (SETLL) operation	7.3.2.8.4	7-35
Screen formatting	8.3.6 Fig. 17-5	8-10 17-13	Set off (SETOF) operation description resulting indicator	7.3.2.5.2 12.2.4	7-30 12-7
Screen line numbers, UNISCOPE 100/200	Table 17-1	17-12	Set on (SETON) operation description resulting indicator	7.3.2.5.1 12.2.4	7-30 12-7
Screens, prompt			SETLL operation description processing a demand file	7.3.2.8.4 13.7.3.2 Fig. 13-11	7-35 13-25 13-26
four-column prompt screen, 70 fields	Fig. 13-44	13-94d	Setting indicators on and off	Table 12-3	12-22
function key 5 prompt screen	Fig. 13-39	13-94	Setting indicators via system console		
one-column prompt screen, 23 fields	Fig. 13-40	13-94b	PF key subroutine - SUBR89	12.6.1	12-23
three-column prompt screen, 47 fields	Fig. 13-43	13-94c	unsolicited inquiry request subroutine - SUBR95	12.6.2	12-26
two-column prompt screen, 24 fields	Fig. 13-41	13-94b	SHTDN operation	7.3.2.10	7-38
two-column prompt screen, self-adjusting fields	Fig. 13-42	13-94c	Sign handling field	4.2.9	4-6
Search conditions			Sign status	14.3.2.1	14-8
arrays	13.10.3.4	13-60	Simple edit codes	14.3.1.1	14-5
tables	13.9.3	13-41	Skip channel number translation	Table 8-1	8-6
Secondary data files			Skip fields	8.2.7	8-5
file designation	2.1.2 5.2.3	2-2 5-5	SKIP keyword parameter	18.2.3	18-34
matching records	13.8.3.2 Fig. 13-13	13-32 13-32	Source program deck	1.3	1-25
Sequence checking			Space fields	8.2.6	8-5
line number field	3.2.2	3-2	/SPACE n compiler directive	3.2.4	3-5
page number field	3.2.1	3-2	Special field names description	14.4	14-10
when performed during compilation	Table 3-1	3-4	D?TE, UDATE, UDAY, UMONTH, and UYEAR	14.4.1	14-10
Sequence field			PAGE, PAGE 1-7	14.4.2	14-12
file description	5.2.5	5-6	*PLACE	14.4.3	14-12
file extension	9.2.11 9.2.16	9-6 9-8	printing	Fig. 14-5 Fig. 14-6	14-13 14-13
input format	6.2.2 Table 6-1	6-3 6-4		19.2.1.5	19-8
Sequential files	2.2.1	2-3			
Sequential processing	13.2 13.6	13-2 13-21			
Serious error messages	Appendix D				
Set bit off (BITOF) operation	7.3.2.6.2	7-32			
Set bit on (BITON) operation	7.3.2.6.1	7-31			

Term	Reference	Page	Term	Reference	Page
Specifications forms			Subtract (SUB) operation	7.3.2.1.3	7-9
auto report options	19.5	19-84	Summary, auto report		
calculation	Section 7		placement, H-*AUTO page		
common fields	3.2	3-2	heading lines	19.7.5	19-102
control card	Section 4		placement, D/T-*AUTO column		
file description	Section 5		heading lines over fields	19.7.6	19-103
file extension	Section 9		placement, D/T-*AUTO fields	19.7.7	19-105
input format	Section 6		placement, D/T-*AUTO headings		
line counter	Section 10		or fields next to totals	19.7.8	19-106
output format	Section 8		skipping	19.7.9	19-106
telecommunications	Section 11		spacing, H-*AUTO page heading		
type and number required	3.1	3-1	lines	19.7.1	19-100
use, action programs	16.2	16-2	spacing, D-*AUTO lines	19.7.2	19-100
Split control fields, rules	6.3.5	6-15	spacing, D/T-*AUTO column		
Spread card feature			heading lines	19.7.4	19-102
description	6.2.5.2	6-5	spacing, T-*AUTO lines	19.7.3	19-101
examples	Fig. 6-2	6-6	Switched field	11.2.5	11-3
	Fig. 6-3	6-8	System date format	4.2.6	4-5
Square root (SQRT) operation	7.3.2.1.8	7-12	System shutdown (SHTDN)	7.3.2.10	7-38
Stacker select/fetch overflow field	8.2.3	8-3			
Stacker select field	6.2.7	6-11			
Standard default options, RPG statement	Fig. 18-1	18-8			
	Fig. 18-2	18-9			
Statements, job control	See job control statements and procedure call statements.				
Statements, using arrays to reduce number	13.10.3.6	13-66			
Station, type	11.2.3	11-3			
SUB operation code	7.3.2.1.3	7-9			
Subroutine field	4.2.14	4-7			
Subroutines					
detail logic	A.3	A-6			
	Fig. A-2	A-7			
external	15.3	15-4			
incorporating in your program	15.1	15-1			
internal	15.2	15-1			
operations	See branching and exit operations.				

Term	Reference	Page	Term	Reference	Page
<b>T</b>					
Table, file translation	See file translation.		T-*AUTO total reports		
Table file	2.1.2 5.2.3	2-2 5-5	description	19.4	19-54
Table linkage field, unformatted dump	C.8.1.1.4	C-11	examples of entries	19.4.2.7	19-82
Table or array, number of entries	9.2.7	9-5	field description and control entries	19.4.2	19-59
Table or array name field	9.2.5 9.2.12	9-4 9-7	field description, prints field and column heading	19.4.2.1	19-62
Tables, file processing			field description, prints field next to total	19.4.2.6	19-79
adding elements	13.9.3.4 Fig. 13-19	13-46 13-47	field description, prints heading	19.4.2.2	19-66
binary search	4.2.8	4-5	field description, prints heading next to total	19.4.2.5	19-76
computing payroll	Fig. 13-17	13-44	field description, prints numeric field, column heading, and accumulates totals	19.4.2.3	19-68
definition	13.9.2	13-35	field description, prints second and third column heading line	19.4.2.4	19-74
definition examples	13.9.2.2 Fig. 13-15	13-39 13-39	output file identification and control entries	19.4.1	19-54
description	13.9	13-33	Telecommunications		
file description form entries	13.9.2.1 Table 13-2	13-36 13-37	CCA name field	4.2.13	4-7
file extension form entries	13.9.2.1 Table 13-3	13-36 13-38	description	17.1	17-1
first element positioning	13.9.1	13-33	error processing	17.6	17-24
formats	13.9.1 Fig. 13-14	13-33 13-34	file processing modes	17.3	17-2
LOKUP operation	13.9.3 13.9.3.1 Fig. 13-16	13-41 13-41 13-41	network definition requirements for RPG II programs	17.4	17-22
modifying	13.9.3.3 Fig. 13-18	13-46 13-46	program requirements	17.5	17-24
using	13.9.3	13-41	specifying and describing communications files	Fig. 17-1	17-3
using data from LOKUP	13.9.3.2	13-43	using	17.2	17-1
Tag file	2.1.2 5.2.3	2-2 5-5	using batch remote terminals	17.3.1	17-5
TAG operation			using interactive remote terminals	17.3.2	17-11
description	7.3.2.4.2	7-26	Telecommunications specifications form		
internal subroutines	15.2	15-2	configuration	11.2.2	11-3
Tape rewind option	5.2.24 Table 5-9	5-29 5-29	description	11.1	11-1
			examples of entries	Fig. 11-1	11-2
			file name	11.3	11-6
			last file	Fig. 11-2	11-7
			permanent error indicator	11.2.1	11-1
			remote device	11.2.9	11-5
			remote terminal	11.2.7	11-4
			summary of remote terminal and remote device entries	11.2.10	11-5
			switched	11.2.6	11-4
			terminal name	Table 11-1	11-6
			transparency	11.2.5	11-3
			type of station	11.2.11	11-6
			wait time	11.2.4	11-3
				11.2.3	11-3
				11.2.8	11-5

Term	Reference	Page	Term	Reference	Page
Teletypewriter and DCT 500 formatting a printer and entering data	Fig. 17-7	17-18	Total time	1.4.2	1-7
transmit with reception of conversational reply	17.3.2.2.1	17-19	Transaction buffer area (TBA)	See action programs.	
using	Fig. 17-8 17.3.2.2	17-19 17-17	Translating characters for more than one but not all files	13.12.2.2	13-82
Terminal name field	11.2.11	11-6	in a file	Fig. 13-34 13.12.2.1 Fig. 13-33	13-82 13-81 13-82
Terminals			Translation, file	See file translation.	
BCS	17.3.1.3	17-9	Translation table, file translation	4.2.11	4-6
DCT 1000	17.3.1.2	17-7	Transmit a file, then receive another file mode		
DCT 2000	17.3.1.1	17-5	description	17.3	17-2
UNISCOPE	See remote terminals. 17.3.2.1	17-11	using BCS remote terminal	17.3.1.3 Fig. 17-4	17-9 17-9
See also batch terminals and interactive terminals.			Transmit only mode		
Test bit operation			DCT 1000	17.3.1.2	17-7
description	7.3.2.3.2	7-24	description	Fig. 17-3	17-7
resulting indicator	12.2.4	12-7		17.3	17-2
Test numeric operation			Transmit with reception of conversational reply		
description	7.3.2.3.3	7-25	description	17.3	17-3
resulting indicator	12.2.4	12-7	using four UNISCOPE 100s	17.3.2.1.1	17-14
Test operations, compare and	See compare and test operations.		using two DCT 500s and a teletypewriter terminal	Fig. 17-6	17-14
Test zone operation			Transparency field	11.2.4	11-3
description	7.3.2.3.4	7-25	Type field	8.2.2	8-3
resulting indicator	12.2.4	12-7	Type of station	11.2.3	11-3
TESTB operation	7.3.2.3.2	7-24	Types of files	2.1.1	2-1
Testing aids	See program testing aids.				
TESTN operation	7.3.2.3.3	7-25			
TESTZ operation	7.3.2.3.4	7-25			
Time of day operations	7.3.2.9	7-38			
TIME operation	7.3.2.9.1	7-38			
/TITLE compiler directive	3.2.4	3-5			
TO file name field	9.2.4 Table 9-1	9-3 9-4			
Total reports	19.4	19-54			

Term	Reference	Page	Term	Reference	Page
<b>U</b>					
U UPDATE, UDAY, UMONTH, and UYEAR field names	14.4.1	14-10	UPSI byte external indicators setting, compilation time	12.3.1 D.1	12-11 D-1
ULABL operation description external subroutines	7.3.2.4.8 15.3	7-30 15-4	User device routine	5.2.19	5-26
Undefined record types	Fig. 6-4	6-9	Using RPG II cards workstation	Fig. 1-2 1.2 Fig. 1-1	1-24 1-1 1-23
Unformatted dump description example locating data field locating *ERROR field locating indicator locating information locating table linkage field for table or array using	C.8 Fig. C-4 C.8.1.1.3 C.8.1.1.2 C.8.1.1.1 C.8.1.1 C.8.1.1.4 C.8.1	C-10 C-25 C-11 C-11 C-10 C-10 C-11 C-10			
UNISCOPE 100 and UNISCOPE 200 Display Terminals formatting a screen and entering data transmit with reception of conversational reply using	Fig. 17-5 17.3.2.1.1 Fig. 17-6 17.3.2.1	17-13 17-14 17-14 17-11			
Unordered load field	5.2.21	5-27			
Unpacked numeric data format	2.4.4 Fig. 2-8	2-12 2-12			
Unsolicited inquiry request subroutine - SUBR95	12.6.2	12-26			
Unwanted control break examples entries to avoid incorrect output input records outputs	Fig. 12-3 Fig. 12-2 Fig. 12-4	12-4 12-3 12-6			
Update file	2.1.1 5.2.2	2-1 5-3			
Updating an indexed sequential file by chaining	B.7	B-33			
			<b>V</b>		
			Variable-length records	2.3	2-7

Term	Reference	Page	Term	Reference	Page
<b>W</b>			<b>Z</b>		
Wait time field	11.2.8	11-5	Z-ADD operation	7.3.2.1.2	7-8
Writing a program	1.2 Fig. 1-1	1-1 1-2	Z-SUB operation	7.3.2.1.4	7-9
Workstation			*ZERO	7.3.1	7-6
as primary or demand file	5.2.3	5-5	*ZEROS	7.3.1	7-6
entering a program	B.9.3	B-57	Zero and add (Z-ADD) operation	7.3.2.1.2	7-8
file processing	13.13	13-80	Zero and subtract (Z-SUB) operation	7.3.2.1.4	7-9
format name	8.3.10	8-13	ZERO or BLANK field	6.3.8	6-20
general information	1.3	1-5	Zeros		
length of format name	8.3.6	8-10	edit codes	14.3.1.1	14-5
screen formats	5.2.16	5-19	edit words	14.3.2.2	14-8
multiple workstations	B.9.3	B-57	Zone, C/Z/D field	6.2.6.3	6-10
	5.2.18	5-20			
Workstation options					
IND	5.2.18.2	5-22			
INFDS	5.2.18.2	5-22			
INFSR	5.2.18.2	5-22			
NUM	5.2.18.2	5-22			
SAVDS	5.2.18.2	5-22			
WORKSTN file	13.13	13-82			
<b>X</b>					
XFOOT operation	7.3.2.1.9	7-12			







## USER COMMENTS

We will use your comments to improve subsequent editions.

NOTE: Please do not use this form as an order blank.

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update Level)*

### Comments:

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage is necessary if mailed in the U.S.A.)  
Thank you for your cooperation



FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY CORPORATION**

**ATTN: Documentation Quality Control Group  
C/O SYSTEM PUBLICATIONS**



P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19422-9990



FOLD

**USER COMMENT SHEET**

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

**Comments:**

Cut along line.

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY UNIVAC**

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD

## USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

### Comments:

Cut along line.

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY UNIVAC**

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



CUT

FOLD

## USER COMMENT SHEET

Your comments concerning this document will be welcomed by Sperry Univac for use in improving subsequent editions.

*Please note: This form is not intended to be used as an order blank.*

---

*(Document Title)*

---

*(Document No.)*

*(Revision No.)*

*(Update No.)*

**Comments:**

Cut along line.

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage stamp is necessary if mailed in the U.S.A.)  
Thank you for your cooperation



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

---

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

---

POSTAGE WILL BE PAID BY ADDRESSEE

SPERRY UNIVAC

ATTN.: SYSTEMS PUBLICATIONS

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424







## USER COMMENT SHEET

We will use your comments to improve subsequent editions.

NOTE: Please do not use this form as an order blank.

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update No.)*

### Comments:

**From:**

---

*(Name of User)*

---

*(Business Address)*

CUT

FOLD

NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

---

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

---

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY CORPORATION**

ATTN.: SOFTWARE SYSTEMS PUBLICATIONS

P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19424



FOLD



## USER COMMENTS

We will use your comments to improve subsequent editions.

NOTE: Please do not use this form as an order blank.

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update Level)*

### Comments:

**From:**

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage is necessary if mailed in the U.S.A.)  
Thank you for your cooperation

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

---

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

---

POSTAGE WILL BE PAID BY ADDRESSEE

**SPERRY CORPORATION**

**ATTN: SYSTEM PUBLICATIONS**



P.O. BOX 500  
BLUE BELL, PENNSYLVANIA 19422-9990



FOLD

## USER COMMENTS

We will use your comments to improve subsequent editions.

NOTE: Please do not use this form as an order blank.

---

*(Document Title)*

---

*(Document No.)*

---

*(Revision No.)*

---

*(Update Level)*

### Comments:

From:

---

*(Name of User)*

---

*(Business Address)*

Fold on dotted lines, and mail. (No postage is necessary if mailed in the U.S.A.)  
Thank you for your cooperation



FOLD



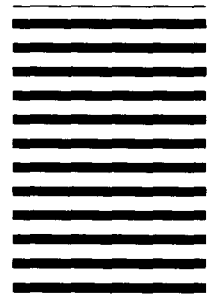
NO POSTAGE  
NECESSARY  
IF MAILED IN THE  
UNITED STATES

**BUSINESS REPLY MAIL**

FIRST CLASS PERMIT NO. 21 BLUE BELL, PA.

POSTAGE WILL BE PAID BY ADDRESSEE

Unisys Corporation  
E/MSG Product Information Development  
PO Box 500 C1-NE6  
Blue Bell, PA 19422-9990



FOLD



