

The IBM logo consists of the letters "IBM" in a bold, white, sans-serif font, centered within a solid black square.

Systems Reference Library

**IBM System/360 Operating System
Report Program Generator
Language**

This reference publication contains fundamentals of RPG programming and language specifications for the IBM System/360 Operating System, Report Program Generator.

Also included is the job setup information for executing RPG.



Preface

The System/360 Operating System, Report Program Generator (RPG) is a problem-oriented language designed to provide users with an efficient, easy-to-use technique for generating programs that can:

1. Obtain data records from single or multiple input files,
2. Perform calculations on data taken from input records or RPG literals,
3. Write reports,
4. Use Table Lookup,
5. Exit to a user's subroutine written in a language other than RPG,
6. Branch within the calculations,
7. Sequence check input records,
8. Maintain files.

RPG uses a set of specification sheets on which the user makes entries. The forms are simple, and the headings on the sheets are largely self-explanatory.

Although many reports use only one input file, RPG can combine data from multiple input files to create a report. The output may be a single report, or it may be several reports created simultaneously on different devices.

For information on the operating system that is beyond the purpose of this publication, refer to the following publications:

IBM System/360 Operating System, System Programmer's Guide (Form C28-6550)

IBM System/360 Operating System, Assembler (E) Programmer's Guide (Form C28-6595)

IBM System/360 Operating System, Concepts and Facilities (Form C28-6535)

IBM System/360 Operating System, Supervisor and Data Management Services (Form C28-6646)

IBM System/360 Operating System, Job Control Language (Form C28-6539)

IBM System/360 Operating System, Linkage Editor (Form C28-6538)

IBM System/360 Operating System, Supervisor and Data Management Macro-Instructions (Form C28-6647)

IBM System/360 Operating System, System Generation (Form C28-6554)

IBM System/360 Operating System, Storage Estimates (Form C28-6551)

For titles and abstracts of associated publications, see the IBM System/360 Bibliography (Form A22-6822).

FOURTH EDITION

This is a reprint of C24-3337-2 incorporating changes released in the following Technical Newsletters:

| Form Number | Pages Affected | Date | Form Number | Pages Affected | Date |
|-------------|--|-------------------|-------------|--|-------------------|
| N21-5088 | 61, 62, 65, 66, 69, 70, 77, 78, 78.1, 78.2, 139, 140, 143, 144, 149, 150 | January 10, 1968 | N21-5093 | 1-4, 31, 32, 32.1, 32.2, 41, 42, 42.1, 42.2, 57, 58, 61, 62, 65, 66, 69-72, 81, 82, 85, 86, 89, 90, 93, 94, 94.1, 94.2, 95-98, 98.1, 98.2, 99, 100, 111, 112, 121, 122, 125-128, 131-134, 139-146, 146.1, 146.2, 147, 148, 163, 164, 175, 176, 185, 186, 191-198 | November 15, 1968 |
| N21-5089 | 87, 88, 119, 120, 135, 136, 139, 140, 145, 146 | February 28, 1968 | | | |
| N21-5092 | iii, iv, 5, 6, 65, 66, 114.1-114.4, 135, 136, 139, 140, 143-146, 146.1, 146.2, 147, 148, 185, 186, 193-198 | May 1, 1968 | | | |

Changes are continually made to the specifications herein; any such change will be reported in subsequent revisions or Technical Newsletters.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Programming Publications, Department 425, Rochester, Minnesota 55901.

CONTENTS

| | | | |
|--|----|---|-------|
| OPERATING SYSTEM/360 REPORT | | LINE COUNTER SPECIFICATIONS SHEET. . . | 89 |
| PROGRAM GENERATOR. | 1 | How to use Line Counter Specifications | 89 |
| Compatibility of System/360 Operating System | 1 | FILE DESCRIPTION SPECIFICATIONS SHEET. | 91 |
| Function of RPG | 1 | Entries for File Description Sheet. | 97 |
| Using RPG | 1 | FILE EXTENSION SPECIFICATIONS SHEET. . | 99 |
| Machine Features Required | 2 | Entries on the File Extension Sheet | 102 |
| Additional Machine Features Supported | 2 | USING TABLES AND EXIT ROUTINES IN THE OBJECT MODULE. | 104 |
| FUNDAMENTALS OF RPG PROGRAMMING. | 4 | Using Tables in the Object Module. . . | 104 |
| Describing a Record and its Fields to the System | 4 | Retrieving Updated Tables | 106 |
| Addition and Subtraction. | 6 | Methods of Processing Tables. | 107 |
| Detail Printing | 7 | Example of Using Tables | 109 |
| Control Fields | 8 | Exit to a User's Subroutine. | 111 |
| Total Calculations. | 9 | General Information | 111 |
| Detail and Total Printing | 10 | How to Code Exit. | 111 |
| Group Printing. | 12 | Position of Exit in the Calculation Specifications. | 111 |
| Group Indication. | 12 | General Rules for Using Exit. | 111 |
| Overflow Printing | 13 | Use of Registers. | 112 |
| Summary Punching. | 14 | Using Indicators, Fields, and Tables in the Exit Routine. | 112 |
| Testing for Zero, Plus, and Minus Balance | 18 | Example of Exit to a Subroutine . . | 113 |
| Using Resulting Indicators. | 19 | USING THE DETAIL-TO-TOTAL BRANCH . . .114.1 | |
| Comparison of Two Fields. | 21 | Data Movement from Input Area to Field Locations | 114.1 |
| Multiplication and Division | 22 | Suppression of Normal Activity. . .114.1 | |
| Sequence-Checking | 24 | Branching | 114.1 |
| Correlation of the RPG Specifications Sheet. | 26 | Output. | 114.1 |
| Summary | 26 | Example | 114.1 |
| PROGRAM LOGIC. | 28 | DISK STORAGE CONCEPTS. | 115 |
| Problem Definition | 29 | Introduction and Terminology | 115 |
| Printer Spacing Chart | 29 | File Organization | 116 |
| GENERAL INFORMATION | 31 | File Processing. | 117 |
| RPG Specification Sheets. | 31 | File Processing in RPG | 118 |
| Common Fields | 33 | PROCESSING SINGLE INPUT FILES. | 120 |
| INPUT SPECIFICATIONS SHEET | 35 | Sequential File | 120 |
| General Information. | 35 | Processing an Indexed-Sequential File Between Limits | 120 |
| Record Identification Entries. | 36 | Random Processing of an Indexed-Sequential File | 121 |
| Field Description Entries. | 42 | Processing Files with Direct Organization. | 122 |
| Summary of Input Specifications . . | 53 | Creating Record Address Files (RAF) | 124 |
| CALCULATION SPECIFICATIONS SHEET . . . | 55 | PROCESSING MULTIPLE INPUT FILES. . . . | 127 |
| Specifying When Calculations are to be Performed. | 55 | Sequential Processing of Multiple Input Files (Matching). | 127 |
| Specifying the Kind of Calculation . . | 58 | Randomly Processing Multiple Input Files (Chaining). | 131 |
| Testing the Results of Calculations. . | 71 | Conversion of Chaining Fields for Direct File Organization. | 134 |
| Using the Calculation Specifications Sheet | 72 | Summary of Multiple File Processing | 136 |
| OUTPUT-FORMAT SPECIFICATIONS SHEET . . | 75 | | |
| General Information | 75 | | |
| File Identification and Control. . . . | 75 | | |
| Field Description. | 80 | | |
| Page Numbering. | 84 | | |

| | | | |
|---|-----|--|-----|
| RPG JOB PROCESSING | 138 | APPENDIX D. STERLING ROUTINES FOR THE REPORT PROGRAM GENERATOR. | 172 |
| Job Control Language. | 138 | APPENDIX E. CONVERSION ROUTINE OPERATION CODES. | 174 |
| Compiler Processing | 138 | APPENDIX F. SUMMARY OF RPG SPECIFI- CATION FORMS | 175 |
| Linkage Editor Processing | 140 | APPENDIX G. DIAGNOSTIC MESSAGES. . . | 183 |
| Load Module Execution | 142 | APPENDIX H. CONDITIONS THAT AUTO- MATICALLY TURN ON HALT INDICATOR (H0). | 192 |
| Using Cataloged Procedures. | 142 | APPENDIX I. RPG COMPILER RETURN CODES | 193 |
| Cataloged Procedures. | 144 | INDEX | 194 |
| RPG Source Program Deck Arrangement | 148 | | |
| RPG Control Card. | 148 | | |
| Executing RPG--Input Stream Variations | 149 | | |
| APPENDIX A. SAMPLE PROGRAMS | 152 | | |
| APPENDIX B. INDICATOR CHART | 168 | | |
| APPENDIX C. RPG LOGIC FLOWCHARTS. . . | 169 | | |

SYSTEM/360 OPERATING SYSTEM REPORT PROGRAM GENERATOR

RPG operates under control of the System/360 Operating System (referred to as the operating system). The operating system provides the RPG compiler with input and output services. Object programs generated by the RPG compiler also operate under operating system control and depend on it for similar services.

RPG supports the minimum configuration required by the operating system. The decimal arithmetic feature is required for RPG.

COMPATIBILITY OF SYSTEM/360 OPERATING SYSTEM

The RPG source language is upward compatible. The operating system Report Program Generator can compile a Model 20, a Basic Programming Support, Tape Operating System, Disk Operating System, or Basic Operating System source program that adheres to its language specifications.

The File Description Specifications Sheet of the operating system RPG may require additional information (record length and overflow indicators).

The control card of a Model 20 or Basic Programming Support program must be modified before the object program can be generated by the operating system RPG.

When a BOS, TOS, or DOS RPG program is used to retrieve records from a file with direct organization, it may be necessary to alter the method of obtaining the addresses of these records.

The operating system RPG does not use the Address Output Option of the Basic or Disk Operating System Disk Sort/Merge to retrieve records.

FUNCTION OF RPG

When RPG is used, the IBM System/360 actually performs the separate functions:

1. program-generating, and
2. data processing.

In the first function, program specifications defined by the user produce machine-language instructions. Storage areas are automatically assigned; constants or other reference factors are included; and linkages to routines for computations for input/output operations, and for other functions are produced.

In the second function, the machine-language instructions (created in the first

function) are combined with the user input-data files, and both are processed through the system to produce the desired reports or output files.

USING RPG

The preparation of a report by means of RPG consists of the general operations illustrated in Figure 1 and described below:

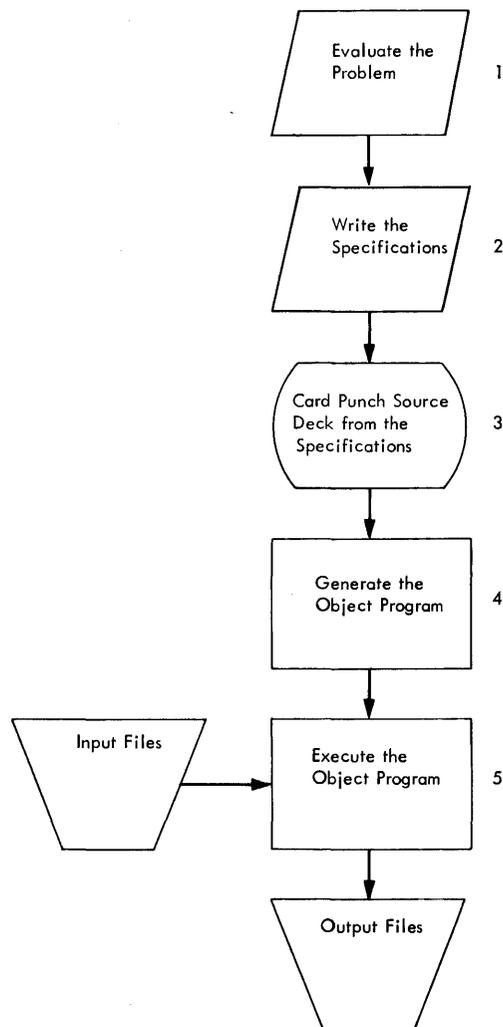


Figure 1. Producing Reports Using the RPG Program

1. The programmer must evaluate the report requirements to determine the format of the input files and the appearance of the finished report.
For example, he must know what fields in the input files are to be used, what kind of calculations are to take place, the location of the data in the output records, and the number and the kind of totals that must be accumulated. More specific information regarding the evaluation of report requirements is described in Problem Definition.
2. After the programmer has evaluated the requirements of the report, he provides the same information to the RPG program.
He must describe his input (record layout, fields used, etc.) by making entries on an Input Specifications sheet.
He must state what processing is to be done (add, subtract, multiply) by entries on a Calculation Specifications sheet.
He must state how the finished report is to look (printing position, carriage control, etc.) by making entries in the Output-Format Specifications sheet.
He must describe all files used by the object program (input files, output files, table files, etc.) by making entries on the File Description and File Extension Specifications sheets.
3. After the specifications have been written on the appropriate forms, cards are card punched with the data from the forms.
4. These punched cards (called a source deck) are combined with the processor control card and the operating system Job Control Statements. The source deck and the control cards are supplied to an input device and are processed under control of the operating system. At the end of this step (known as a compilation step), a program capable of preparing the report specified by the programmer has been produced. This program (known as an object module) is processed by the link edit step to create a loadable program. This program (known as the load module) contains all of the computer instructions and linkages to the control system necessary to prepare the desired report.
5. The input files can then be read into the system, and processing of the program will begin. This is known as the execution step.

At the end of the execution step, the report has been prepared and any other functions, such as file updating, are completed. Through facilities provided by the operating system, the object module can be retained for later runs without recompilation.

MACHINE FEATURES REQUIRED

Source Program Compilation

1. 32,768 main storage bytes
2. One of the following for input:
 - IBM 2540 Card Read-Punch
 - IBM 2501 Card Reader
 - IBM 2520 Card Read-Punch
 - IBM 1442 Card Read-Punch
 - IBM 2400 Series Magnetic Tape Unit
3. One of the following for output of object program:
 - IBM 2540 Card Read-Punch
 - IBM 2520 Card Read-Punch
 - IBM 1442 Card Read-Punch
 - IBM 2400 Series Magnetic Tape Unit
 - IBM 2311 Disk Storage Drive
4. One IBM 2311 Disk Storage Drive for system residence
5. One of the following for system utility files:
 - IBM 2400 Series Magnetic Tape Unit (Three)
 - Data Convert Feature (Required only if 7-track tape is used)
 - IBM 2311 Disk Storage Drive with three files
6. Standard and Decimal Instruction Sets
7. The minimum machine configuration required by the operating system

Object Module Execution

1. 32,768 main storage bytes
2. I/O units as requested by the specifications
3. Standard and Decimal Instruction Sets
4. Systems Residence only — IBM 2311 Disk Storage Drive
5. The minimum machine configuration required by the operating system

ADDITIONAL MACHINE FEATURES SUPPORTED

Source Program Compilation

1. 65,536; 131,072; 262,144; or 524,288 main storage bytes

2. One of the following for listings:
 - IBM 1443 Printer
 - IBM 1403 Printer
 - IBM 1404 Printer (continuous forms operation only)
 - One IBM 2400 Series Magnetic Tape Unit (9 track)

3. Two additional IBM 2311 Disk Storage Drives for utility files

Object Module Execution

1. 65,536; 131,072; 262,144 or 524,288 main storage bytes

Once the card columns of the field have been specified and the appropriate field name has been defined for the field, other references to this field are made by using its field name rather than writing down the specific card columns each time.

Although the input illustrated in Figure 3 is simplified, the entries shown will cause the contents of fields A, B, and C from the card to be read into the system during the processing of the object module. Calculations to be made from the input data are written on the Calculation Specifications sheet.

ADDITION AND SUBTRACTION

Problem

In this problem, fields A, B, and C from the previous example are used to calculate a new field (Field D). The calculation $A + B - C = D$ is to be performed.

Specifications

Figure 4 shows the required entries, using the fields from the previous example. The form required for this problem is the Calculation Specifications sheet.

The first line of the form tells the object program to add the contents of FLDB to the contents of FLDA and place the result in FLDD.

1. The 14 in Indicators defines the card type for which the calculations are to be performed as assigned by the Input Specifications sheet.
2. The card columns for FLDA and FLDB were defined by entries on the input forms and need not be specified again.
3. The result field FLDD is defined for the program by merely writing the name FLDD in Result Field (columns 43-48) and indicating in Field Length (columns 49-51) the number of positions that must be set aside for this field in the object program. Just as in the case of FLDA and FLDB, which were defined on the input form, the name FLDD can now

IBM

Date _____

Program _____

Programmer _____

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System 360

Form X24-3351-1
Printed in U.S.A.

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Central Level (C, O, P, L, B) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Half Adjust (H) | Resulting Indicators | | | Comments | | | |
|------|-----------|-------------------------------|------------|-----|-----|----------|-----------|----------|--------------|--------------|--------------------------------------|----------------------|-------|---------------|----------|---------------|--------------|----------------|
| | | | And | And | And | | | | | | | Plus | Minus | Zero or Blank | | Compare | | |
| | | | | | | | | | | | | | | | | High 1 > 2 | Low 1 < 2 | Equal 1 = 2 |
| | | | | | | | | | | | | | | | | | | |
| 0 1 | C | | 14 | | | FLDA | ADD | FLDB | FLDD | 80 | | | | | | | | |
| 0 2 | C | | 14 | | | FLDD | SUB | FLDC | FLDD | | | | | | | | | |
| 0 3 | C | | | | | | | | | | | | | | | | | |
| 0 4 | C | | | | | | | | | | | | | | | | | |
| 0 5 | C | | | | | | | | | | | | | | | | | |
| 0 6 | C | | | | | | | | | | | | | | | | | |
| 0 7 | C | | | | | | | | | | | | | | | | | |
| 0 8 | C | | | | | | | | | | | | | | | | | |
| 0 9 | C | | | | | | | | | | | | | | | | | |

Figure 4. Calculation Specifications Sheet

be used in other calculation operations or used in defining output specifications.

NOTE: The result of A + B is not placed back into FLDA. To do this would destroy the original contents of FLDA. In this example it is necessary to save the original value of FLDA so that it can be printed on an output report.

The second line in Figure 4 causes the object module to subtract the contents of FLDC from the result just previously obtained in FLDD and to store the new result back into FLDD.

Decimal Symbol Location

In this example there were no decimal positions in any of the three fields. In operations involving numbers containing decimal positions, the number of decimal positions in each of the fields is frequently not the same. When the number of decimal positions is not the same in both factors in an arithmetic operation, shifting the factors to align the decimal point is usually required.

RPG automatically shifts the factors.

The programmer must indicate the number of decimal positions contained in each factor used in calculations, and the number of decimal positions required in the result. If a field is to have arithmetic operations performed upon it, the decimal positions must be specified. A zero is coded for no decimal positions (See Figure 4). Assume that the fields in Figure 3 have decimal positions as indicated under Decimal Positions (Column 52) in the following example:

A set of values for these fields might appear to the program as follows:

| <u>FIELD</u> | <u>DECIMAL POSITION</u> | <u>CONTAINED IN CARDS</u> | <u>ACTUAL VALUE</u> |
|--------------|-------------------------|---------------------------|---------------------|
| A | 0 | 00126 | 126. |
| B | 2 | 01123 | 11.23 |
| C | 3 | 04264 | 4.264 |

The programmer must indicate the number of decimal positions required in the result. The calculation A + B - C = D, would result in the values being shifted like this:

```

126.000 = A
+11.230 = B
-----
137.230
-  4.264 = C
-----
132.966 = D

```

(Result field has three decimal positions).

The result 132.966 is stored as the value of the field called D. Entering the decimal positions on the input sheet and the calculation sheet enables the factors to be shifted automatically.

DETAIL PRINTING

Detail printing is the printing of information obtained from each record as it is read.

Problem

This problem illustrates how input fields A, B, and C, and the calculated result, field D, can be specified for listing.

Specifications

Figure 5 shows the output specifications required. The numbers in the following list refer to items circled in Figure 5.

- The output listing must be assigned its own specific file name. In this example it has been called DAILYRPT. The D in Type H/D/T indicates that the line being printed is a detail line. That is, it contains information from the record just read. (The other possible entries, H and T, indicating heading lines and total lines, are described later.) The 2 in Space After provides a double-spaced listing. (There is one blank line after each printed line.)
- The 14 in Output Indicators identifies the input record type present when detail printing occurs.

BUSINESS MACHINES CORPORATION
REPORT INPUT SPECIFICATIONS
 System/360

Page 1 2

| Position | Not (N) C/Z/D | Character | Shrinker Select | Packed (P) | Field Location | | Field Name | Decimal Positions | Control Level (1,1,0) | Matching Fields or Chaining Fields |
|----------|------------------|-----------|-----------------|------------|----------------|----|------------|-------------------|-----------------------|------------------------------------|
| | | | | | From | To | | | | |
| 46 | | | | | 50 | 00 | FLDA | | | |
| 56 | | | | | 60 | 02 | FLDB | | | |
| 66 | | | | | 70 | 03 | FLDC | | | |

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS
 Form X24-335-1
 Printed in U.S.A.

Form 360

Page 1 2

Program Identification 75 76 77 78 79

| Line | Form Type | Control Level (L1-L9) | Indicators | | | Field Location | Field Name | Operation | Factor 2 | Result Field | Field Length | Decimal Positions | Resulting Indicators | | | Comments |
|------|-----------|-----------------------|------------|-----|-----|----------------|------------|-----------|----------|--------------|--------------|-------------------|----------------------|-------|---------------|----------|
| | | | And | And | And | | | | | | | | Plus | Minus | Zero or Blank | |
| 01 | C | 14 | | | | FLDA | ADD | FLDB | FLDD | 70 | | | | | | |
| 02 | C | 14 | | | | FLDD | SUB | FLDC | FLDD | 70 | | | | | | |
| 03 | C | 14 | | | | TOTE | ADD | FLDD | TOTE | 80 | | | | | | |
| 04 | C | L1 | | | | TOTF | ADD | TOTE | TOTF | 80 | | | | | | |
| 05 | C | L2 | | | | TOTG | ADD | TOTF | TOTG | 80 | | | | | | |
| 06 | C | L3 | | | | FINTOT | ADD | TOTG | FINTOT | 80 | | | | | | |

Figure 7. Example of Control Field Sequence

operation $A + B - C = D$. The third line adds the result, contained in FLDD, from each detail card to a field called TOTE. Therefore, TOTE is the accumulated amount for each employee group. The first three specification lines are performed in the object module each time a detail card is read (see Figure 8, Indicator 14).

When the level-1 (L1) control break occurs (a card from the next employee group has been sensed), specification-line 4 adds the total of TOTE accumulated from each detail card into a field called TOTF. TOTF is used to accumulate the total for each employee group for each department.

When the level-2 control break occurs, TOTF is added into TOTG.

When the level-3 control break occurs, TOTG (which represents the accumulated amounts for each division) is added into FINTOT.

Accumulating totals at each control-break level is normally done when the corresponding totals are printed on an output report.

NOTE: In System/360 RPG—as in IBM punched card equipment—a control break at one level forces control breaks for all lower levels.

Specifications

Figure 8 contains the data shown in Figure 4. The additional information for this example is circled. During processing of the object module, lines 1 and 2 cause the

DETAIL AND TOTAL PRINTING

Problem

This example shows the specifications necessary to print the three controlling

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
 Form X24-3351-1
 Printed in U.S.A.

IBM

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic Punched

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Control Level (L0-L9) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions | Resulting Indicators | | | Comments |
|------|-----------|-----------------------|------------|-----|-----|----------|-----------|----------|--------------|--------------|-------------------|----------------------|-------|---------------|----------|
| | | | And | And | And | | | | | | | Plus | Minus | Zero or Blank | |
| 01 | C | 14 | | | | FLDA | ADD | FLDB | FLDD | 70 | | | | | |
| 02 | C | 14 | | | | FLDD | SUB | FLDC | FLDD | 70 | | | | | |
| 03 | C | 14 | | | | TOTE | ADD | FLDD | TOTE | 80 | | | | | |
| 04 | C | L1 | | | | TOTF | ADD | TOTE | TOTF | 80 | | | | | |
| 05 | C | L2 | | | | TOTG | ADD | TOTF | TOTG | 80 | | | | | |
| 06 | C | L3 | | | | FINTOT | ADD | TOTG | FINTOT | 80 | | | | | |

Figure 8. Specifying Control Levels on the Calculation Sheet

fields, the name, the accumulated amount in field D, the three accumulated totals, and the final total at the end of the object-module report.

Specifications

Figure 9 illustrates the specifications required for the report shown in Figure 10. The following numbers refer to the items circled in Figure 9:

1. This information is similar to the specifications in the previous example in Figure 5.
2. The data fields DIVSON, DEPT, EMPNO, NAME, and FLDD are specified for printing.
3. The specifications to print the total for the first control level shown on lines 7 and 8 are:
 - a. The T in H/D/T (column 15) indicates that the line is a total line. A total line is an operation

- b. A 3 in Space After (column 18) provides two blank lines after each printed line to make the report easier to read.
- c. The L1 in Output Indicators (columns 24-25) indicates that the line is to be printed only on a level-1 control break.
- d. The field to be printed (TOTE) is indicated under Field Name (columns 32-37).
- e. The Z in Zero Suppress (column 38) indicates that zeros to the left of significant digits are not to be printed.
- f. The B in Blank After (column 39) causes the core-storage positions containing field TOTE to be set to zeros after the total is printed.

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
 IBM System/360

Form X24-3352-1
 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

| Line | Form Type | Filename | Space | | | | Skip | | | | Output Indicators | | | Field Name | Zero Suppress (Z) | Blank After (B) | End Position in Output Record | Packed Field (P) | Constant or Edit Word | Sterling Sign Position |
|------|-----------|-----------|--------|-------|--------|-------|------|-----|-----|-----|-------------------|-----|-----|------------|-------------------|-----------------|-------------------------------|------------------|-----------------------|------------------------|
| | | | Before | After | Before | After | And | And | And | And | Not | Not | Not | | | | | | | |
| 0 1 | O | DAILYRPTD | | | | | | | | | | | | | | | | | | |
| 0 2 | O | | | | | | | | | | | | | | DIVSON | | 5 | | | |
| 0 3 | O | | | | | | | | | | | | | | DEPT | | 10 | | | |
| 0 4 | O | | | | | | | | | | | | | | EMPNO | | 17 | | | |
| 0 5 | O | | | | | | | | | | | | | | NAME | | 33 | | | |
| 0 6 | O | | | | | | | | | | | | | | FLDD | Z | 64 | | | |
| 0 7 | O | | T | | | | | | | | L1 | | | | | | | | | |
| 0 8 | O | | | | | | | | | | | | | | TOTE | ZB | 66 | | | |
| 0 9 | O | | T | | | | | | | | L2 | | | | | | | | | |
| 1 0 | O | | | | | | | | | | | | | | TOTF | ZB | 66 | | | |
| 1 1 | O | | T | | | | | | | | L3 | | | | | | | | | |
| 1 2 | O | | | | | | | | | | | | | | TOTG | ZB | 66 | | | |
| 1 3 | O | | T | | | | | | | | LR | | | | | | | | | |
| 1 4 | O | | | | | | | | | | | | | | FINTOTZ | Z | 66 | | | |
| 1 5 | O | | | | | | | | | | | | | | | | | | | |

Figure 9. Specifications on the Output-Format Sheet

| DIVSON | DEPT | EMPNO | NAME | | |
|--------|------|--------|-------|------|----------|
| 0112 | 0246 | 011426 | SMITH | 101 | ← FLDD |
| 0112 | 0246 | 011426 | SMITH | 100 | |
| 0112 | 0246 | 011426 | SMITH | 102 | |
| 0112 | 0246 | 011426 | SMITH | 101 | |
| | | | | 404 | ← TOTE |
| 0112 | 0246 | 011428 | JONES | 120 | |
| 0112 | 0246 | 011428 | JONES | 122 | |
| 0112 | 0246 | 011428 | JONES | 123 | |
| 0112 | 0246 | 011428 | JONES | 121 | |
| | | | | 486 | |
| 0112 | 0246 | 001430 | BROWN | 100 | |
| 0112 | 0246 | 001430 | BROWN | 103 | |
| 0112 | 0246 | 001430 | BROWN | 102 | |
| 0112 | 0246 | 001430 | BROWN | 104 | |
| | | | | 409 | |
| | | | | 1299 | ← TOTF |
| 0112 | 0310 | 011296 | GREEN | 121 | |
| 0112 | 0310 | 011296 | GREEN | 120 | |
| 0112 | 0310 | 011296 | GREEN | 144 | |
| 0112 | 0310 | 011296 | GREEN | 102 | |
| | | | | 487 | |
| 0112 | 0310 | 011298 | BLAND | 98 | |
| 0112 | 0310 | 011298 | BLAND | 86 | |
| | | | | 184 | |
| | | | | 671 | |
| | | | | 1970 | ← TOTG |
| 0114 | 0069 | 001262 | ADAMS | 146 | |
| 0114 | 0069 | 001262 | ADAMS | 237 | |
| 0114 | 0069 | 001262 | ADAMS | 184 | |
| 0114 | 0069 | 001262 | ADAMS | 197 | |
| | | | | 764 | |
| 0114 | 0069 | 001278 | JAMES | 182 | |
| 0114 | 0069 | 001278 | JAMES | 176 | |
| 0114 | 0069 | 001278 | JAMES | 160 | |
| 0114 | 0069 | 001278 | JAMES | 164 | |
| | | | | 682 | |
| | | | | 1446 | |
| | | | | 1446 | |
| | | | | 3416 | ← FINTOT |

Figure 10. Detail-Printed Report

This is done so the total for one group is not added to the total of the previous group.

The specifications to print the second and third control levels are essentially the same as those for the first level.

- The specifications for printing the final total contain the code LR (Last Record) in Output Indicators (columns 24-25). The indicator LR is turned on automatically by the program when the

last card of a file has been sensed. This indicator is used to cause the final total to be printed.

GROUP PRINTING

In group-printing operations, only one line is printed for each group of detail cards. This line usually contains the control fields and the totals of the quantity fields.

An example of a group-printed report is shown in Figure 11.

The detail-printed report specifications in Figure 9 could be altered to provide a group-printed report as illustrated by the specifications in Figure 12.

The differences between Figures 9 and 12 that provide for the two types of reports are:

- The detail line specified in Figure 9 (line 01) has been changed to a total line conditioned on an L1 break (Figure 12, line 01).
- The total line in Figure 9 (line 07) has been combined with the first total line 01 of Figure 12.
- The spacing on lines 09 and 11 of Figure 9 have been changed from 3 to 1 space-after in Figure 12.

GROUP INDICATION

In group-indication operations, each detail card is processed; however, only the control fields that identify the specific detail card are printed. An example of a group-indication report is illustrated in Figure 13. In this example the employee name and number are printed for each control change. The fields, division and department, are printed only when there is a control change for the division and department fields, respectively.

| | | | | |
|------|------|--------|-------|------|
| 0112 | 0246 | 011426 | SMITH | 404 |
| 0112 | 0246 | 011428 | JONES | 486 |
| 0112 | 0246 | 011430 | BROWN | 409 |
| | | | | 1299 |
| 0112 | 0310 | 011296 | GREEN | 487 |
| 0112 | 0310 | 011298 | BLAND | 184 |
| | | | | 671 |
| | | | | 1970 |
| 0114 | 0069 | 001262 | ADAMS | 764 |
| 0114 | 0069 | 001278 | JAMES | 682 |
| | | | | 1446 |
| | | | | 1446 |
| | | | | 3416 |

Figure 11. Group-Printed Report

Specifications

The specifications for this example are shown in Figure 22. The program logic for this example is shown in Figure 23.

Line 1 specifies that FLDB is to be added to FLDA and that the result is to be placed in FLDD. Line 2 specifies that FLDC is to be subtracted from FLDD, that the result is to be placed in FLDD and that FLDD is to be tested to determine if the result is minus. In this example it is assumed that if a minus balance occurs, the calculation has no meaning. Therefore, if the result is minus, two things must be accomplished:

1. The result must not be added into field TOTE.

2. The contents of field FLDD must be reset to zeros. (This step might be required if FLDD were used in a subsequent step and the minus balance remaining would give incorrect results.)

This is accomplished on the specification sheet (Figure 22) by placing an indicator code (19) in Resulting Indicators: Minus on the specification-line 2. Indicator 19 will be turned on for a minus condition.

The function of adding TOTE to FLDD is specified on line 3. It is accomplished only if there is a no-minus condition resulting from the test on line 2. The specifications on line 4 cause FLDD to be reset to zeros only on a minus condition.

The 0 in Factor 2 on line 4 is known as a literal and is used to set FLDD to zeros. (A literal is the actual value to

| IBM | | | INTERNATIONAL BUSINESS MACHINES CORPORATION | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | Form X24-3351-1 Printed in U. S. A. | | | |
|------------------|-----------|--------------------------|---|-----|------|----------|-----------|----------|--------------|--------------|--------------------------------------|----------------------|-------|---------------|----------|---------------|--------------|----------------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|----------------------|--|---------|--|-------|--|----------|--|--|--|--|--|--|--|
| | | | REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | IBM System/360 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Date _____ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | Punching Instruction | | Graphic | | Punch | | Page 1 2 | | Program Identification 75 76 77 78 79 80 | | | | | |
| Program _____ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Programmer _____ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Line | Form Type | Central Level (0019, 18) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Half Adjust (H) | Resulting Indicators | | | Comments | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | No | And | No | | | | | | | Plus | Minus | Zero or Blank | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | Compare | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | High 1 > 2 | Low 1 < 2 | Equal 1 = 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 0 | C | 14 | | FLDA | ADD | FLDB | FLDD | 70 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 0 | C | 14 | | FLDD | SUB | FLDC | FLDD | 70 | | 19 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 0 | C | 14 | 19 | FLDD | ADD | TOTE | TOTE | 80 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 0 | C | 19 | 14 | | Z-ADD | 0 | FLDD | 70 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 0 | L1 | | | TOTE | ADD | TOTF | TOTF | 80 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 0 | L2 | | | TOTF | ADD | TOTG | TOTG | 80 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 0 | L3 | | | TOTG | ADD | FINTOT | FINTOT | 80 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 22. Testing for a Minus Condition

be used in a calculation rather than the name of the location of the data to be used.) The remaining specifications on the form in Figure 22 are the same as those from previous examples.

By using Indicator 19, it is possible to suppress detail-card printing when the calculation D - C results in a minus balance.

COMPARISON OF TWO FIELDS

The calculation specifications in this example illustrate the ability to compare two fields and govern processing according to the result of the comparison. The comparison may be tested for a high, low, or equal condition. The result of the test can then be used to bypass some specifications or to process other specifications only when a particular condition is present.

Specifications

The calculation specifications for this example are shown in Figure 24. The program logic for the example is shown in Figure 25.

Line 1 specifies that FLDA is compared to FLDB. If the two fields are equal, Indicator 18 is turned on, and FLDA is moved to HOLD (line 2). If the two fields

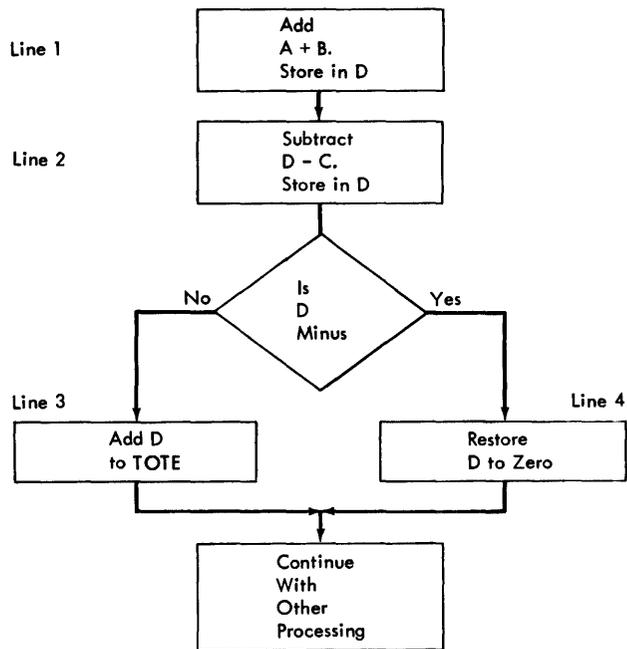


Figure 23. Testing Indicators to Govern Processing

are not equal (line 3), FLDB is subtracted from FLDA and the result is placed in SAVE. A literal may also be used in a comparison specification. In the lower half

| Line | | Farm Type | Control Level | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions | Resulting Indicators | | | Comments |
|------|---|-----------|---------------|------------|-------|------|----------|------------|----------|--------------|-----------------|-------------------|----------------------|---------------|--|----------|
| | | | | And | And | | | | | | Half Adjust (H) | Plus | Minus | Zero or Blank | | |
| | | | | Not | Not | | | | | | | High | Low | Equal | | |
| | | | | 1 > 2 | 1 < 2 | | | | | | | 1 = 2 | | | | |
| 01 | 0 | C | | | | FLDA | COMP | FLDB | | | | | | 18 | | |
| 02 | 0 | C | | 18 | | | MOVE | FLDA | HOLD | 102 | | | | | | |
| 03 | 0 | C | | N18 | | FLDA | SUB | FLDB | SAVE | 102 | | | | | | |
| 07 | 0 | C | | | | DATE | COMP | '02-14-64' | | | | | | 19 | | |

Figure 24. Specifying a Comparison

part number may be present. In this example, multiple cards for issues, receipts, and adjustments may be present.

The letter O in Option in the specifications means that the records are optional. That is, a record may or may not be present.

If the letter O is not specified it means that the particular record must be present. This requirement applies only if Sequence has been specified as numeric.

CORRELATION OF THE RPG SPECIFICATIONS SHEETS

Figure 30 shows a file-to-file sample program. (This type of report is often called an 80/80 listing.) It illustrates the relationships of the RPG specification sheets.

Assume the contents of an input file are to be transferred to another file. In this example the input file is a card file, and the contents of the cards are to be printed. Three specifications sheets are required for this program: File Description, Input, and Output-Format Specifications.

File Description Specifications Sheet

The two files are described on this sheet. The card-input file is assigned the name INPUT, and the printed output file is named OUTPUT. Page and line sequence numbers are entered in columns 1-5. An F in column 6 indicates that each entry is a File Description Entry. The file names are entered in columns 7-14 (Filename). Column 15 contains an I or an O to indicate whether the file has an input or an output function.

Column 16 of the input file of the File Description Specifications Sheet contains a P because the file described is the primary input file for the job; the E in column 17 indicates that the end-of-file condition for this file occurs when this file is depleted.

Column 19 contains an F and a V to indicate that the file formats are fixed-length and variable-length respectively.

Block length (columns 20-23) is 80 for the input file because each card is a block of data. Record length (columns 24-27) is also 80 for the input file because each card is an unblocked record. For the output file, the block length and the record

length is 132, which is the maximum length of a printer line. The program identification is entered into columns 75-80.

If the input file was read in by an IBM 1442 Card Read-Punch (as it is in this example) the code in Device (columns 40-46) would be READ42. The output printer would have a Device code of PRINTER.

Input Specifications Sheet

The Input Specifications sheet also has the program identification entered in columns 75-80 and the page and line numbers in columns 1-5. An I in column 6 of each card indicates an input specification entry.

The filename is again entered into columns 7-14. Columns 15-16 contain the sequence code AA. Indicator 01, entered into columns 19-20, will be on throughout the job to show that records from its associated file are being processed. The second line describes the field name CARDIN. The field consists of card columns 1-80.

Output-Format Specifications Sheet

On the Output-Format Specifications sheet, the filename of the output file is entered in columns 7-14. The D in column 15 indicates that each line printed in this file is a detail line. A single space after printing is specified by the entry in column 18. Indicator 01 from the Input Specifications sheet is specified in columns 24-25. When Indicator 01 is on, a record will be printed.

The second output line has the name of the field to be written in the output record entered in columns 32-37. Data from the field labeled CARDIN will be printed (end position of 80) in the output record as specified in columns 40-43.

SUMMARY

This completes the general description of some of the functions that can be performed with RPG. Some of the fields of the specifications sheets were not explained and some additional operations that can be performed with RPG remain to be described. At this point, the reader should be able to determine the scope of the RPG program.

More specific information about each specification sheet is contained under RPG Specification Sheets.

PROGRAM LOGIC

Each program generated by RPG uses the same general logic, and for each record to be processed the program goes through the same general cycle of operations. Within that cycle, there are two different instances in time when operations specified on the Calculation and Output-Format Specifications sheets are performed. These instances are called detail and total time.

For the illustration of this concept, a generalized flowchart of an RPG generated program is shown in Figure 31.

The following numbers correspond to the numbers on Figure 31. A program cycle begins with item 1 and continues through item 11. Steps 6 and 7 are referred to as total time. Steps 1 and 11 are referred to as detail time.

1. Before the first record is read, the program prepares and writes any heading information to be put out on the first page. After the first record has been read, the program prepares and writes heading and detail information which is not conditioned on overflow.
2. The generated program tests for any halt indicators. If any halt indicators are on, the program branches to item 12.
3. The generated program tests for the end-of-file conditions. If the end-of-file condition has occurred, the program branches to item 13.
4. The generated program then reads an input record.
5. All control level indicators and all resulting indicators (specified) in columns 19-20 of the input sheet) are turned off. Then, starting with line 1 of the Input Specifications sheets, and with the record just read, the generated program uses the record identification code to identify the record. When the identification code matches an entry on the input sheet, the program turns on the resulting indicator that has been specified for the record. When a control-field break occurs, appropriate control-level indicators are turned on.
6. Next, all total calculations are performed. (This step is bypassed for the initial control break which is caused by reading the first input record.)
7. Next, all total output records which are not conditioned on overflow are prepared and put out. (This step is also bypassed for the first control break.)
8. The generated program tests for the last record indicator (LR). If it

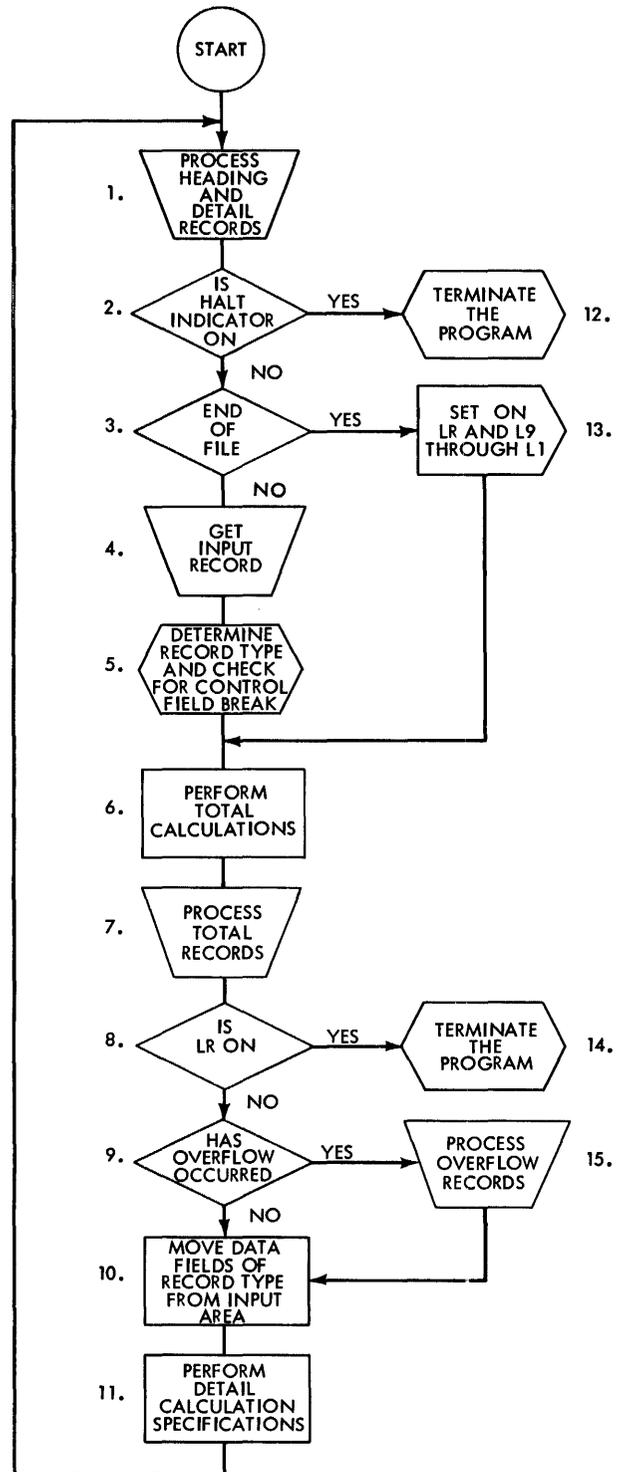


Figure 31. General Logic Flow of a Program Generated by RPG

- is on, the program branches to item 14.
9. The object module tests for an overflow condition. If an overflow condition has occurred, the program branches to item 15. Overflow is defined as the condition existing whenever any of the indicators OA-OG, and OV are on. (This step is also bypassed for the first control break.)
 10. The data fields contained in the input record just read are moved into storage. These fields are specified by field entries on the Input Specifications Sheet.
 11. Any detail calculations are performed, and processing continues with item 1.
 12. Program execution is terminated.
 13. The Last-Record indicator (LR) is set ON and all control-level indicators L1-L9 are set ON. Then the program branches to item 6.
 14. Program execution is terminated.
 15. If overflow has occurred, total lines, heading lines and detail lines (in that order) conditioned by overflow are printed. The program then branches to item 10.

PROBLEM DEFINITION

The programming examples in the preceding section were intended to introduce the reader to the use of RPG and were therefore kept simple. More complex applications may require a thorough analysis of the existing or proposed system before a program can be written.

This analysis should include a description of source data and its format, and how this data should be processed to develop the report and other necessary output information.

The following types of information must be defined before coding the program:

1. The available data
2. The input and output formats to be used
3. The information required in the input and output formats
4. The codes to be used to identify the various inputs and outputs and their elements
5. The handling of the various transactions and exceptions.

After all application data has been gathered, document it for easy reference during the writing of the specification forms. One method of documenting an application is to lay out the complete format of the report on a printer spacing chart. This method also provides a pictorial representation of the final product.

PRINTER SPACING CHART

Before the report specifications are written, the programmer should have a clear picture of what he wants as the final product. If the report is to be printed, he must know the number of fields to be placed on each line of the report, the spacing between lines, and the positioning of the information within each line of the report.

Although no cards for the source deck are punched directly from the entries on this chart, the representation serves as a guide for completing the specification sheets. It plays an important role in writing report specifications. If the final product is written on magnetic tape or direct-access storage devices or if it is punched in cards, the user must know where the information is to be located. A tape layout chart or a direct-access storage-device layout chart can be used.

Layout of Lines and Fields

The two most important functions of a printer spacing chart are

1. To establish the positions of the data to be printed and to indicate the spacing between printed lines.
2. To assign each line an identification code representing the type of line. Figure 32 shows an example of the printer spacing chart (Form X24-6436).

The numbers across the top and bottom of the spacing chart represent the print positions. The numbers down the left side are the line numbers. The programmer selects the line number and print positions for a particular field and makes his notation in the selected positions.

Headings and other constant information are spelled out completely in the print positions assigned to them. Variable information is represented by Xs and, where applicable, includes credit symbols, punctuation, etc. The position in an amount field where zero suppression ends is indicated by a zero rather than an X.

Line-Identification Code

The line-identification code specifies the type of line to be printed. The identification codes are H for a heading line, D for a detail line, and T for a total line. All lines must be identified as belonging to one of these categories.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
PRINTER SPACING CHART
IBM 407, 408, 409, 1403, 1404, and 1443

| LINE DESCRIPTION | FIELD HEADINGS/WORD MARKS | | | | | | | | | | | | 6 Lines Per Inch | | | | | | | | | | | | |
|------------------|---------------------------|---|---|---|---|---|---|---|---|---|----|----|--|---|---|---|---|---|---|---|---|---|----|----|----|
| | | | | | | | | | | | | | ← IBM 1403 Models 1 and 4 | | | | | | | | | | | | |
| | | | | | | | | | | | | | ← IBM 407, 408, 409, and 1403 Model 7 | | | | | | | | | | | | |
| | | | | | | | | | | | | | ← IBM 1403 Models 2, 3, 5, N1 and 1404 | | | | | | | | | | | | |
| | | | | | | | | | | | | | ← IBM 1443 Models 1 and N1 | | | | | | | | | | | | |
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| GLUE | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 13 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 14 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 16 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 17 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 18 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 19 | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 32. Printer Spacing Chart

Cross-References

To make this reference manual a more effective learning tool, numerous cross-references have been placed in the manual. They are located wherever it was thought that readers not familiar with disk storage processing and related functions would have difficulty with these unfamiliar subjects.

Disk storage, table lookup, matching field, and chaining field operations and related functions are described apart from the detailed descriptions of specifications for them.

These general introductory descriptions, contained at the back of the manual, can be used by the reader as he encounters the related specifications for them throughout the manual.

To facilitate locating them in the manual, all cross-references used are listed in the Index under Cross-References.

The section Disk Storage Concepts provides a general introduction to disk file organization and processing including terminology associated with these functions. Readers not familiar with these concepts may wish to review that section before beginning with this section.

Left- and Right-Justification

When making entries on the RPG sheets, it is important that the entry be right-justified or left-justified as required. Justifying an entry means having it begin in the first position of the specification (left-justified), or having it end in the last position of the specification (right-justified).

Alphanumeric entries (composed of both alphabetic and numeric entries) are always left-justified. Numeric entries are always right-justified.

Information regarding the correct justification is provided in the description of the entry, in those cases where it may not be clear to the reader as to whether the entry is alphanumeric or numeric.

Alphabetic Characters

In this manual, the references to alphabetic character designate the letters A through Z, the dollar sign (\$), the pound sign (#), and the at sign (@).

Numeric Field Format

The System/360 packed decimal format allows two decimal digits to be represented in one core storage byte. The RPG object program automatically converts all numeric input data from unpacked to packed format. Unless otherwise specified, all numeric data is unpacked before it is output. In this manual, numeric length refers to the unpacked length although the data is actually stored in the packed format.

Sterling Routines

Sterling Routines are included in RPG to provide users with a convenient and time-saving means of handling amount fields that are punched in the format of Pound Sterling monetary units.

The presence of sterling fields is indicated to the RPG program by additional entries in the input and output specifications sheets and in the RPG Control Card. The other specifications sheets are not affected. All calculations are done in the Pence unit of measure.

RPG SPECIFICATION SHEETS

This section provides detailed explanations of each specification field contained in the six RPG forms.

The forms are listed in the sequence in which they are discussed in this publication (see Figure 33).

- Input Specifications
- Calculation Specifications
- Output-Format Specifications
- Line Counter Specifications
- File Description Specifications
- File Extension Specifications

Input Specifications. (Refer to Figure 34)
This form is used to

1. Specify the file or files to be read into the system
2. Identify the different types of records contained in each file
3. Describe the location of the data fields in each record

Calculation Specifications. This form specifies the operations to be performed

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS
 IBM System 360 Form 324-3347-1
 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

Punching Instruction: Punched Graphic
 Punched Punch

Page 1 of 2
 Program Identification: 15 16 17 18 19 20

| Line | Item Type | Filename | File Type | File Description | Length of Record | Number of Records | Device | Symbolic Device | Name of Label Exit | Extent Exit for DAM | Comments |
|------|-----------|----------|-----------|------------------|------------------|-------------------|--------|-----------------|--------------------|---------------------|----------|
| 1 | 1 | | | | | | | | | | |
| 2 | 2 | | | | | | | | | | |
| 3 | 3 | | | | | | | | | | |
| 4 | 4 | | | | | | | | | | |
| 5 | 5 | | | | | | | | | | |
| 6 | 6 | | | | | | | | | | |
| 7 | 7 | | | | | | | | | | |
| 8 | 8 | | | | | | | | | | |
| 9 | 9 | | | | | | | | | | |
| 10 | 10 | | | | | | | | | | |
| 11 | 11 | | | | | | | | | | |
| 12 | 12 | | | | | | | | | | |
| 13 | 13 | | | | | | | | | | |
| 14 | 14 | | | | | | | | | | |
| 15 | 15 | | | | | | | | | | |
| 16 | 16 | | | | | | | | | | |
| 17 | 17 | | | | | | | | | | |
| 18 | 18 | | | | | | | | | | |
| 19 | 19 | | | | | | | | | | |
| 20 | 20 | | | | | | | | | | |
| 21 | 21 | | | | | | | | | | |
| 22 | 22 | | | | | | | | | | |
| 23 | 23 | | | | | | | | | | |
| 24 | 24 | | | | | | | | | | |
| 25 | 25 | | | | | | | | | | |
| 26 | 26 | | | | | | | | | | |
| 27 | 27 | | | | | | | | | | |
| 28 | 28 | | | | | | | | | | |
| 29 | 29 | | | | | | | | | | |
| 30 | 30 | | | | | | | | | | |
| 31 | 31 | | | | | | | | | | |
| 32 | 32 | | | | | | | | | | |
| 33 | 33 | | | | | | | | | | |
| 34 | 34 | | | | | | | | | | |
| 35 | 35 | | | | | | | | | | |
| 36 | 36 | | | | | | | | | | |
| 37 | 37 | | | | | | | | | | |
| 38 | 38 | | | | | | | | | | |
| 39 | 39 | | | | | | | | | | |
| 40 | 40 | | | | | | | | | | |
| 41 | 41 | | | | | | | | | | |
| 42 | 42 | | | | | | | | | | |
| 43 | 43 | | | | | | | | | | |
| 44 | 44 | | | | | | | | | | |
| 45 | 45 | | | | | | | | | | |
| 46 | 46 | | | | | | | | | | |
| 47 | 47 | | | | | | | | | | |
| 48 | 48 | | | | | | | | | | |
| 49 | 49 | | | | | | | | | | |
| 50 | 50 | | | | | | | | | | |
| 51 | 51 | | | | | | | | | | |
| 52 | 52 | | | | | | | | | | |
| 53 | 53 | | | | | | | | | | |
| 54 | 54 | | | | | | | | | | |
| 55 | 55 | | | | | | | | | | |
| 56 | 56 | | | | | | | | | | |
| 57 | 57 | | | | | | | | | | |
| 58 | 58 | | | | | | | | | | |
| 59 | 59 | | | | | | | | | | |
| 60 | 60 | | | | | | | | | | |
| 61 | 61 | | | | | | | | | | |
| 62 | 62 | | | | | | | | | | |
| 63 | 63 | | | | | | | | | | |
| 64 | 64 | | | | | | | | | | |
| 65 | 65 | | | | | | | | | | |
| 66 | 66 | | | | | | | | | | |
| 67 | 67 | | | | | | | | | | |
| 68 | 68 | | | | | | | | | | |
| 69 | 69 | | | | | | | | | | |
| 70 | 70 | | | | | | | | | | |
| 71 | 71 | | | | | | | | | | |
| 72 | 72 | | | | | | | | | | |
| 73 | 73 | | | | | | | | | | |
| 74 | 74 | | | | | | | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS
 IBM System 360 Form 324-3348-1
 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

Punching Instruction: Punched Graphic
 Punched Punch

Page 1 of 2
 Program Identification: 15 16 17 18 19 20

| Line | Item Type | From Filename | To Filename | Table Name | Number of Table Entries Per Record | Length of Table Entry | Table Name | Length of Table Entry | Comments |
|------|-----------|---------------|-------------|------------|------------------------------------|-----------------------|------------|-----------------------|----------|
| 1 | 1 | | | | | | | | |
| 2 | 2 | | | | | | | | |
| 3 | 3 | | | | | | | | |
| 4 | 4 | | | | | | | | |
| 5 | 5 | | | | | | | | |
| 6 | 6 | | | | | | | | |
| 7 | 7 | | | | | | | | |
| 8 | 8 | | | | | | | | |
| 9 | 9 | | | | | | | | |
| 10 | 10 | | | | | | | | |
| 11 | 11 | | | | | | | | |
| 12 | 12 | | | | | | | | |
| 13 | 13 | | | | | | | | |
| 14 | 14 | | | | | | | | |
| 15 | 15 | | | | | | | | |
| 16 | 16 | | | | | | | | |
| 17 | 17 | | | | | | | | |
| 18 | 18 | | | | | | | | |
| 19 | 19 | | | | | | | | |
| 20 | 20 | | | | | | | | |
| 21 | 21 | | | | | | | | |
| 22 | 22 | | | | | | | | |
| 23 | 23 | | | | | | | | |
| 24 | 24 | | | | | | | | |
| 25 | 25 | | | | | | | | |
| 26 | 26 | | | | | | | | |
| 27 | 27 | | | | | | | | |
| 28 | 28 | | | | | | | | |
| 29 | 29 | | | | | | | | |
| 30 | 30 | | | | | | | | |
| 31 | 31 | | | | | | | | |
| 32 | 32 | | | | | | | | |
| 33 | 33 | | | | | | | | |
| 34 | 34 | | | | | | | | |
| 35 | 35 | | | | | | | | |
| 36 | 36 | | | | | | | | |
| 37 | 37 | | | | | | | | |
| 38 | 38 | | | | | | | | |
| 39 | 39 | | | | | | | | |
| 40 | 40 | | | | | | | | |
| 41 | 41 | | | | | | | | |
| 42 | 42 | | | | | | | | |
| 43 | 43 | | | | | | | | |
| 44 | 44 | | | | | | | | |
| 45 | 45 | | | | | | | | |
| 46 | 46 | | | | | | | | |
| 47 | 47 | | | | | | | | |
| 48 | 48 | | | | | | | | |
| 49 | 49 | | | | | | | | |
| 50 | 50 | | | | | | | | |
| 51 | 51 | | | | | | | | |
| 52 | 52 | | | | | | | | |
| 53 | 53 | | | | | | | | |
| 54 | 54 | | | | | | | | |
| 55 | 55 | | | | | | | | |
| 56 | 56 | | | | | | | | |
| 57 | 57 | | | | | | | | |
| 58 | 58 | | | | | | | | |
| 59 | 59 | | | | | | | | |
| 60 | 60 | | | | | | | | |
| 61 | 61 | | | | | | | | |
| 62 | 62 | | | | | | | | |
| 63 | 63 | | | | | | | | |
| 64 | 64 | | | | | | | | |
| 65 | 65 | | | | | | | | |
| 66 | 66 | | | | | | | | |
| 67 | 67 | | | | | | | | |
| 68 | 68 | | | | | | | | |
| 69 | 69 | | | | | | | | |
| 70 | 70 | | | | | | | | |
| 71 | 71 | | | | | | | | |
| 72 | 72 | | | | | | | | |
| 73 | 73 | | | | | | | | |
| 74 | 74 | | | | | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR LINE COUNTER SPECIFICATIONS
 IBM System 360 Form 324-3349-1
 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

Punching Instruction: Punched Graphic
 Punched Punch

Page 1 of 2
 Program Identification: 15 16 17 18 19 20

| Line | Item Type | Filename | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|------|-----------|----------|---|---|---|----|---|---|---|---|---|----|----|----|
| 1 | 1 | | | | | | | | | | | | | |
| 2 | 2 | | | | | | | | | | | | | |
| 3 | 3 | | | | | | | | | | | | | |
| 4 | 4 | | | | | | | | | | | | | |
| 5 | 5 | | | | | | | | | | | | | |
| 6 | 6 | | | | | | | | | | | | | |
| 7 | 7 | | | | | | | | | | | | | |
| 8 | 8 | | | | | | | | | | | | | |
| 9 | 9 | | | | | | | | | | | | | |
| 10 | 10 | | | | | | | | | | | | | |
| 11 | 11 | | | | | | | | | | | | | |
| 12 | 12 | | | | | | | | | | | | | |
| 13 | 13 | | | | | | | | | | | | | |
| 14 | 14 | | | | | | | | | | | | | |
| 15 | 15 | | | | | | | | | | | | | |
| 16 | 16 | | | | | | | | | | | | | |
| 17 | 17 | | | | | | | | | | | | | |
| 18 | 18 | | | | | </ | | | | | | | | |

File Extension Specifications
 Line Counter Specifications
 Input Specifications
 Calculation Specifications
 Output-Format Specifications

Line (3-5)

Each specification line may be identified by a line number. The first two digits of the line number are pre-printed on the form. The third position (column 5) is used when it becomes necessary to insert an additional line between two previously written lines. The line to be inserted is written following line 15. It is given an appropriate number (and subnumber in column 5).

The page number and line number have no direct effect on the program and need not be written. These columns are for the convenience of the programmer to indicate the proper order of the RPG source program cards. For example, the specification cards for a program could be placed in numeric sequence (if, for example they were accidentally dropped or upset) by sorting or arranging them in sequence by page number and line number.

Form Type (6)

Each form has an appropriate type-code pre-printed in column 6. This code must be punched into all specifications cards. The codes are

- I Input specifications
- C Calculation specifications
- O Output-Format specifications
- L Line Counter specifications
- F File Description specifications
- E File Extension specifications

Comments (*Column 7)

This feature enables the programmer to insert an identifying comment in the specification sheets. This facility can be used, for example, to identify the end of one section of a program. These comments are written on the specification line, preceded by an asterisk in column 7. During

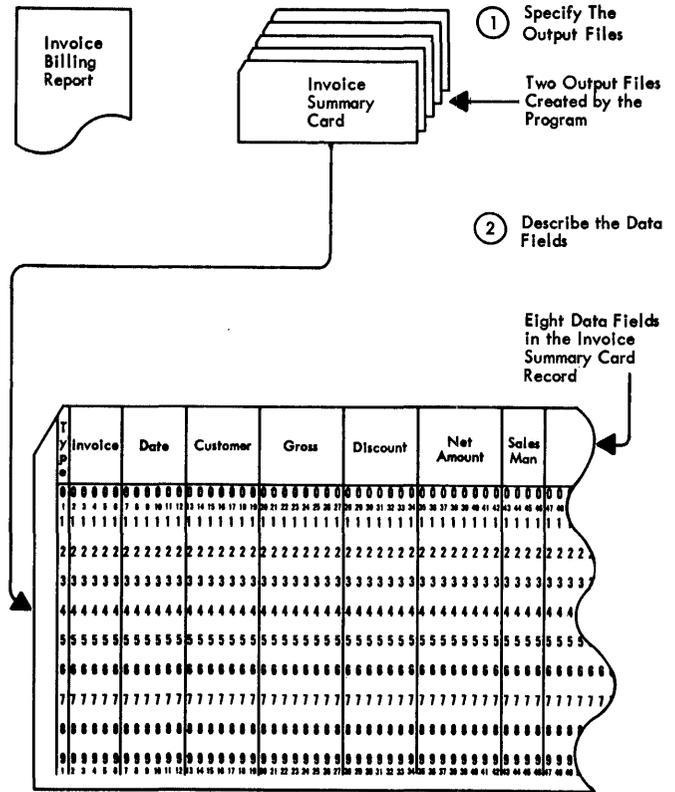


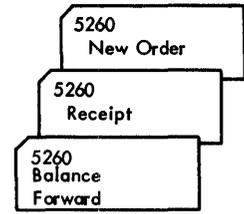
Figure 35. Functions of Output-Format Specifications

the generation of the object program, the asterisk in column 7 identifies the comment so that it is not considered a specification.

Program Identification (75-80)

This identification is located in the upper right-hand corner of the sheets. This entry identifies the specification cards for a particular program or for a specific section of a large program.

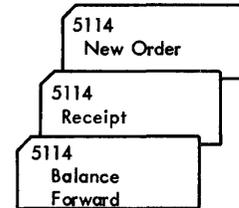
input list, then, as each card is read, all specifications for the exceptions must be examined first before the specifications for the normal processing are found. Thus a great amount of processing time would be wasted if the card file contained only two or three exception cards but the exception specifications had to be examined for all 3000 cards.



RECORD IDENTIFICATION ENTRIES

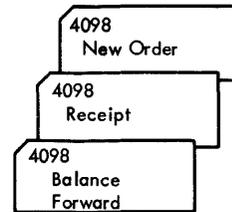
FILENAME (7-14)

A file name must be given to each input file. The file name must be left-justified (that is, it must start in column 7) and it must begin with an alphabetic character. The remaining characters of the name may be alphameric, but must not contain special characters or embedded blanks. The file name may be eight characters or fewer. (Embedded blanks are blank positions falling between other characters of the name.)



NOTE: In this publication, alphabetic character refers to the letters A through Z, the dollar sign, pound sign, and the at sign (\$, #, and @).

The file name must be entered only with the first record-identification line of the appropriate file.



SEQUENCE (15-16)

This specification is used to check the sequence of cards within a control group. Figure 37 illustrates a card file containing three types of cards for each part-number control group. In this example, to assure correct accumulation of the values, the balance-forward card must be the first card in the control group, the receipt card the second, and the new order card the last.

The specifications for checking the sequence of these cards is shown in Figure 38. (Field description specifications are not shown.)

If the file is not in sequence the halt indicator H0 is turned on. Unless this H0 indicator is turned off by a SETOF OPERATION (See Turning Indicators On or Off) in the calculation specifications, the program will terminate before the next input record is read.

The cards are specified on the form in the same sequence in which they are to be read by the object module.

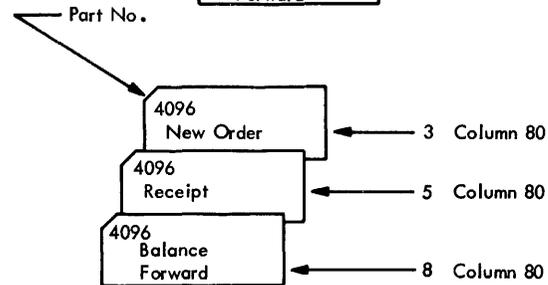


Figure 37. Card Types within Control Groups

NOTE: The entries in Sequence must begin with 01 in each file and be consecutive in ascending order.

Alphabetic codes must be placed in Sequence if the input records do not have to be in sequence within a control group or if it is not necessary to stop processing when the records are not in sequence. Any two alphabetic characters can be used.

Within a given file, header cards or other cards that are not in sequence must be specified on the form before specifications that must be sequence-checked.

| REPORT PROC | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------------|----------|---|---|----|----|----|-----------------------|------------|---------------------|----------|----|----|----|---------|-------|-----------|----------|----|----|----|----|----|----|----|----|----|----|----|
| Record Ident | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Form Type | Filename | | | | | | Sequence Number (1-N) | Option (O) | Resulting Indicator | Position | | | | Net (N) | C/Z/D | Character | Position | | | | | | | | | | | |
| 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 |
| | W | E | K | L | R | P | T | Ø | 1 | 1 | 2 | 2 | 8 | Ø | C | 8 | | | | | | | | | | | | |
| | | | | | | | | | Ø | 2 | 1 | 0 | 2 | 3 | 8 | Ø | C | 5 | | | | | | | | | | |
| | | | | | | | | | Ø | 3 | 1 | 0 | 2 | 4 | 8 | Ø | C | 3 | | | | | | | | | | |

Figure 38. Example of Sequence-Checking within Control Groups

NOTE: If a numeric specification is given in Sequence then specifications must also be provided in Number and Option.

NUMBER (17)

If a numeric code is assigned under Sequence, an entry must also be made in Number (column 17). If an alphabetic code has been assigned under Sequence, this column must be blank.

This specification indicates whether only one record of a specific record-type should exist in each control group or whether one or more than one record of a specific record-type may exist in each control group. For example, Figure 39 illustrates two control groups of cards. In this example, there can only be one balance-forward record in each control group, but there may be one or more new orders or receipt records.

The entry for the specification Number is either:

- 1 if only one record of a type may exist in a control group, or
- N if one or more records of a type may exist in a control group.

The specifications required for the example in Figure 39 are illustrated in Figure 40.

OPTION (18)

This specification is used only with numeric-sequenced record types. If the presence of a record is optional, the letter O is entered in this column. If a specific record type must be present in order to perform an operation, or if records are non-sequential, this column must be left blank. In both Figures 38 and 40, the specifications under Option indicate that there must be a balance-forward card for each control group, but there may or may not be a new-order or receipt card.

RESULTING INDICATOR (19-20)

This specification is used in conjunction with the next specification Record Identification Codes (21-41). It has two purposes:

- 1. To establish a 2-digit code for each input record type.

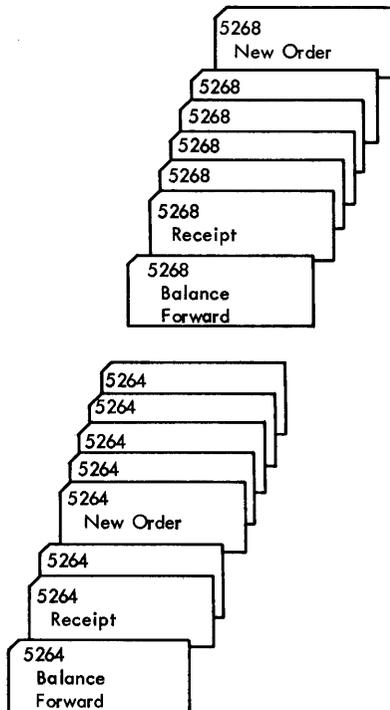


Figure 39. Number of Record Types within a Control Group

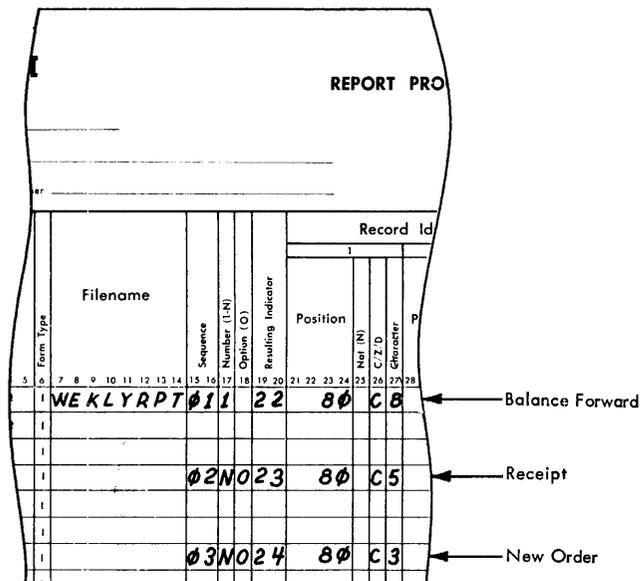


Figure 40. Example of Using Number Specification

- To set up a special condition in the object program each time the input record is read into the system. The object program may consider this condition during the processing of the calculation and output specifications.

As an example of the first function, assume that a certain card type is identified by the following codes:

- Digit 5 in column 40
- 11-punch in column 79
- No 12-punch in column 80

By assigning a two-digit resulting indicator to represent all of these codes it is much easier to refer to this card type during the writing of the calculation and output specifications.

As an example of the second (and more important) function of this specification, resulting indicators can be compared to selectors in punched-card machines, or to internal or external switches on electronic data processing machines. The use of resulting indicators (like the use of selectors and switches), is to permit certain operations to occur only on specific conditions.

Figure 41 illustrates, by symbols, how resulting indicators are used in the object program. In this example, a payroll file contains three types of cards.

| Card Type | Resulting Indicator |
|------------------|---------------------|
| current earnings | 14 |
| deduction | 15 |
| adjustment | 16 |

When one of these cards is read into the system during the object run, the appropriate resulting indicator is turned on, and those specifications pertaining to the record are performed. The detail specifications for these record types are indicated on the calculation and output specifications forms and are controlled by one of these three indicators. Specifications associated with other record types are not performed.

The input specifications required to establish the three indicators shown above are illustrated in Figure 42. (Field description specifications are not shown.)

Resulting indicators—from input records—are turned ON and OFF, during the processing of the object program, as the various record types are read by the system. However, only one resulting indicator can normally be on at one time. When a resulting indicator is turned on, all other resulting indicators are turned off. (See Chaining for an exception to this rule.)

Other indicator conditions that can be established in the program are Field

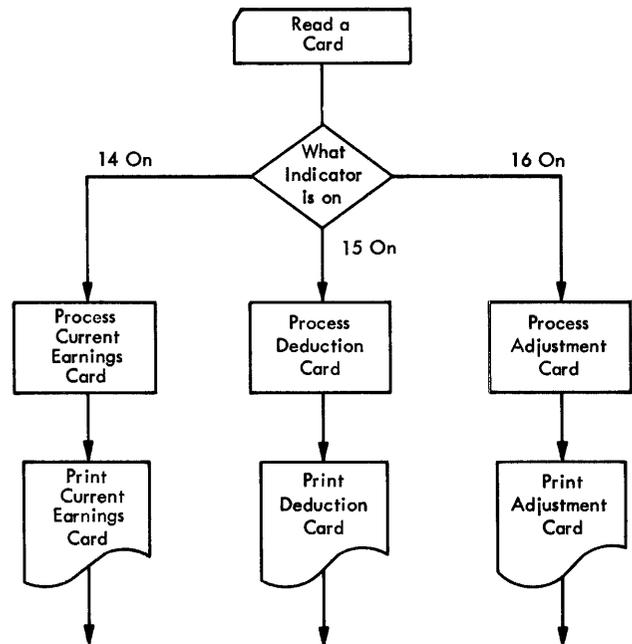


Figure 41. Sample Logic Flow Using Resulting Indicators

| IBM | | | | | | | | | | | | | | | | | | INTERNAL REPORT PROGRAM | | | | | | | | | | | | | |
|------------------|-----------|----------|-----------------------|------------|---------------------|-----------------------|----|----|----|-----------|----|----|----|----------|----|----|----|-------------------------|----|----|----|----|----|----|----|----|----|----|----|----|--|
| Date _____ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Program _____ | | | | | | | | | | | | | | | | | | Punching Instruction | | | | | | | | | | | | | |
| Programmer _____ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Line | Form Type | Filename | Sequence Number (1-N) | Option (O) | Resulting Indicator | Record Identification | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | Position | | | | Character | | | | Position | | | | | | | | | | | | | | | | | |
| 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 | |
| 0 1 | I | PAYROLL | AA | | 14 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 2 | I | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 3 | I | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 4 | I | | 88 | | 15 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 5 | I | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 6 | I | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 7 | I | | CC | | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 8 | I | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 9 | I | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 42. Specifying Resulting Indicators on Input Specifications

Indicators (columns 65-70) of the input specifications and Resulting Indicators (columns 54-59) of the calculation specifications. These functions are described later, but one aspect of their use is of interest at this time.

All three types of indicators are assigned a 2-digit number in the range of 01 through 99. Any of these 99 codes can be assigned to any one of the three types of indicators. Also the indicator codes do not have to be assigned in any sequence. For example, four different card types that are read into the system could be assigned codes 40, 62, 99, 02.

RECORD IDENTIFICATION CODES (21-41)

This specification provides a way of identifying each different record type used in the generated program. As mentioned previously, once the record type has been defined on the Input Specifications sheet, references to the record are made by its resulting indicator.

These columns provide for the entry of one to three identifying codes as indicated by the number 1, 2, and 3 on the Input Specifications sheet. It is possible to specify more than three record identification codes by using more than one line. If

only one record type is in the input file, the identification codes can be left blank but a resulting indicator and a sequence number must be specified.

Each of the three sets of entries is the same, so only the first set (columns 21-27) is described. Each set is divided into four categories:

- Position (21-24)
- Not (25)
- C/Z/D (26)
- Character (27)

Position (21-24)

Enter in these columns the position in each data record of the character that contains the identifying code. The position must be right-justified in columns 21-24, and leading zeros may be omitted.

Not (25)

Enter an N in this column if the code described must not be present in the specified record position. Otherwise leave this column blank.

C/Z/D (26)

The object module identifies the different record-types of a program by comparing the character written in Character against the codes contained in the records. The entry in Column 26 specifies whether the object program is to compare against the entire character (C), against only the zone portion (Z), or against only the digit portion (D). Enter a C, Z or D in column 26.

Zone Record Identification. Testing a zone means that the zone of the character located in the position specified in columns 21-24 of the input sheet is used to identify the record. (The word "Zone" in this case refers to the meaning of this word in punched-card data processing systems. It does not refer to bits 0-3 of the System/360 EBCDIC character coding.)

The ampersand, the minus, and the blank are exceptions. An ampersand will be identified as a 12-zone, a minus will be identified as an 11-zone, and a blank will be identified as a no-zone. The four common zones are:

1. 12-zone or plus zone (0, A-I, and &)
2. 11-zone or minus zone (0, J-R, and minus)
3. 0-zone (S-Z)
4. No zone (0-9 and Blank).

| Input Unit | Stacker Number | Stacker Select Code |
|--------------------------|----------------|---------------------|
| IBM 1442 Card Read-Punch | 1 2 | 1 or Blank 2 |
| IBM 2501 Card Reader | - | Blank |
| IBM 2520 Card Read-Punch | 1 2 | 1 or Blank 2 |
| IBM 2540 Card Read-Punch | R1 R2 | 1 2 |

Figure 45. Summary of Stacker Select Specifications (Input)

stacker-select entry must be written on each OR-record identification line.

FIELD DESCRIPTION ENTRIES

As mentioned previously, the Input Specifications sheet consists of two categories: record identification and field description.

On the record identification portion of the Input Specifications sheet, one line represents the specifications for one record type. On the field-description portion of the sheet, one line represents the specifications for one field of a record.

The following information concerns the individual field descriptions of one record type. Field descriptions are always written on the specification line immediately below the specification line that identifies the record type.

Field description entries describe the fields of the input record to be used in the report. Each field of the record requires one line on the Input Specifications sheet. Columns 7-42 of the line must be blank.

NOTE: Unused record fields should not be described, since this would waste core storage and processing time.

PACKED (43)

Packed format in the System/360 means that two decimal digits can be represented in one core storage byte. This is the data format used for numeric fields in RPG. Because input data is usually represented in the unpacked format—one digit in one core

storage byte—the RPG program automatically converts numeric input data from the unpacked format to the packed decimal format. Because the packed decimal format permits greater utilization of storage capacities (card-tape-disk) the RPG program permits numeric data to be put out in the packed decimal format. (See Output-Format Specifications Sheets.)

In order to utilize this numeric output data in subsequent processing runs, RPG permits data in packed decimal format to be read into the RPG program.

Enter a P in this column if the numeric input field is in the packed decimal format. Otherwise leave this column blank. (The letter P causes the RPG program to bypass the normal conversion of unpacked format to packed decimal format.)

The implied field length for determining the length of fields in calculation specifications for input in packed decimal format is:

$$2n - 1$$

n = number of input record positions used

FIELD LOCATION (44-51)

Columns 44-51 of the Input Specifications sheet are used to describe the location of each field in the record. The maximum field length for a numeric field is 15 digits. The maximum field length for an alphanumeric field is 256 characters.

From (44-47). This specification contains the location of the first position (leftmost position) of the field.

To (48-51). This specification contains the location of the last position (rightmost position) of the field.

NOTE: The entries in columns 44-47 and 48-51 must be right-justified. Leading zeros may be omitted.

Figure 46 illustrates a card input record. The field location specifications necessary to read this card into the system are shown in Figure 47.

The fields of the record may be listed in any sequence. (In this example they are shown in the same sequence as they appear on the card to make the example easier to understand.)

The specifications in Decimal Positions and Field Name are also included in Figure 47 because all three fields are closely

related. These specifications are explained following the description of Field Location.

Records in an OR-Relationship

It is possible to specify two different record types with just one set of specifications. This is known as an OR relationship. There are three types of OR relationships:

1. One or more record types have the same fields in the same positions of the record. (For example, all fields in one record type are in the same relative positions in another record type.)
2. One or more record types have the same fields, but the fields are in different positions of the record. (For example, unit cost is in positions 21-25 in one record type and in positions 31-35 in another record type.)
3. One or more record types have different fields in the same positions or in different positions of the record. (For example, two record types with ten fields in the same relative positions, but with three fields in different positions.)

It is of value to specify an OR relationship for the types in items 2 and 3 above only if there are more fields that are alike than fields that are unlike.

As an example of the first type of OR relationship assume that there were two detail labor cards in the preceding example; perhaps the second having been created during the previous week's reporting. If the second card had the same fields as the first, but with a different record identification code, it would not be necessary to repeat all of the field specifications for the second card.

Figure 48 shows the only additional specifications required in order to specify both detail labor cards. The specification OR is entered in columns 14 and 15, and columns 16-18 are left blank. The last OR line (if there is more than one OR relation) is followed by the field description entries.

Figures 49 and 50 illustrate how to specify two records in an OR relationship when the field locations are not the same. The input record in Figure 49 is similar to the record in Figure 46 except that FLDC is located in positions 61-65, instead of positions 66-70. By specifying an OR relationship it is possible to specify both record types with one set of specifications.

Figure 50 illustrates the specifications for this example. The numbers in the margin of the subsequent text refer to the numbers circled in Figure 50.

1. When field locations are not the same for both types, it is necessary to provide a separate resulting indicator code for the second record type.
2. Each field that is not located in the same positions in both record types must be specified twice. Each of the two specifications must be related to the appropriate record type. This is accomplished by specifying the appropriate resulting indicator code in Field-Record Relation.

For example, FLDC located in positions 66-70 is related to resulting indicator 14 (the record type identified by a 5 in column 35); and field FLDC in positions 61-65 is related to resulting indicator 16 (the record type identified by a 6 in column 35). Thus, if resulting indicator 14 is on, FLDC will be taken from positions 66-70; if resulting indicator 16 is on, FLDC will be taken from positions 61-65.

It is also possible to take advantage of an OR relationship to specify control fields that are not in the same positions in two different record types. In addition, specifying record types in an OR relationship is not limited to just two records. As many records as required can be specified as having an OR relationship.

DECIMAL POSITIONS (52)

This specification performs two functions:

1. It is used by the object program to determine the number of decimal positions contained in the field specified in Field Location.
2. It causes the field specified in Field Location to have 0-(zero), 11-, or 12-zone bits removed from all positions except the rightmost position.

If the field specified in Field Location is to have calculations or edit functions performed upon it or if it will be zero-suppressed in the output, there must be a specification in Decimal Positions. If there are no decimal positions in an arithmetic field, a zero must be specified. This specification must be left blank if the field is alphameric.

In Figure 51, the fields DIVSON, DEPT, and EMPNO contain numeric information, but

| Field Location | | Field Name | Control Level (L1-L9) | Matching Fields or Chaining Fields | Field-Record Relation | Field Indicators | | | Sterling Sign Position |
|----------------|----|------------|-----------------------|------------------------------------|-----------------------|------------------|-------|---------------|------------------------|
| From | To | | | | | Plus | Minus | Zero or Blank | |
| 1 | 40 | DIVNO | L3 | | | | | | |
| 5 | 8 | DEPT | L2 | | | | | | |
| 9 | 14 | MANNO | L1 | | | | | | |
| 15 | 28 | NAME | | | | | | | |
| 29 | 32 | ORDRNO | | | | | | | |
| 33 | 40 | BALNCE | | | 01 | | | | |
| 41 | 48 | PARTNO | | | | | | | |
| 49 | 55 | FIELD1 | | | | | | | |
| 56 | 60 | FIELD2 | | | | 02 | | | |
| 61 | 69 | FIELDA | | | | | | | |
| 70 | 74 | CUSTNO | | | | | | | |
| 76 | 79 | AMTDUE | | | | | | 03 | |
| 28 | 32 | CUSNO | L4 | | | | | | |
| 15 | 20 | ACTNO | L4 | | | | | | |
| 50 | 52 | REGNO | L4 | | | | | | |

Figure 52. Field Names

Two input files may have fields with the same field names. A field name is assigned only once by RPG. The two input files will use the same storage location for fields with identical names. This procedure is permissible when using RPG, but the programmer should be aware that possible errors in calculations or output may occur when two input files have the same field names. This is especially important when chaining fields are specified on the Input Specifications sheet.

Specifying the Same Data Field as Alpha-meric and Numeric

If the same field from an input record is to be used as both an alphameric and a numeric field, the field must be specified twice by assigning two different field names to the same location in the record.

(If no decimal positions are specified for the field, it is considered to be an alphameric field.)

Using Input Data Fields as Constants

The term constant is frequently used to describe information that is not changed with each record. It is usually not altered during the object run. Examples of constants are date cards, or other data that may be changed for each run.

The technique for getting this information to the program is to define a special record type and assign a field name that is not otherwise used. This technique will also permit entering of constants that are too large to be specified as literals in the source program.

When only a single input file is processed, the date or constant card(s) may be placed at the front of the file preceding the first data record.

If an additional file is required for entering the constant information, it may be designated as a primary file, and the data file as a secondary file. Using this technique, the primary file will be processed first and the constant information will be entered prior to the data records.

If multiple input files are processed, the constant card(s) may appear as the first records on any file. The constant records should be defined in the input specifications for the associated file and should not have matching fields. (See Matching Fields.) In a job with multiple files, the constant file could be designated as secondary and since its associated input specifications have no matching fields, its records will be processed first.

CONTROL LEVEL (59-60)

As the object program is processed, a change in a control field causes all processing indicated by this change to be initiated. Columns 59-60 are used to provide a convenient and simple method for specifying all control functions.

Up to nine control levels can be used by RPG. These levels are designated from low to high as L1, L2, L3, ..., L9. An indicator similar to a resulting indicator is associated with each control-level designation. They are used to control functions specified on the calculation and output specification sheets. When the field specified in Field Location is a control

field, its appropriate control level must be specified in columns 59-60. The first three lines in Figure 52 show the entries for the three control fields.

The field DIVNO (first entry in the example) is the highest level of control.

The indicator L3 could be used to specify when the calculation specifications for the control field DIVNO are to take place, or when the totals for the field DIVNO are to be printed or summarized. More information regarding the use of the control level indicators is presented in the descriptions of the calculation and output specifications sheets.

NOTE: Whenever control levels are used, a control break will occur on the first record of a record type which has control levels. Total calculations, total lines and overflow lines are bypassed until after the control break occurs.

Additional Functions of Control Level Specifications

If a field specified in Field Location also has an entry in Control Level, the object program places the field into two storage areas. One area, known as a control-field holding area, is used for the controlling functions of the field, and the other is used for any other uses of the field (such as for printing or for arithmetic calculations).

When an alphameric field is used as a control field, the zone bits in the field are used in the comparison of one record to the next. To use an alphameric field for controlling functions without considering the zone bits, specify the field as numeric by making an entry in Decimal Positions. In Figure 52 the field DIVNO is used for controlling functions. The entry L3 causes the value of the field DIVNO to be stored in the control-field holding area in packed format with a plus sign of C. The sign of the value is C regardless of the sign of the input field.

Split Control Fields

Several fields in an input record can be specified as one control field. In the lower half of Figure 52, three fields, which are not in adjacent record positions, are specified with the same L4 control

level. The three fields are treated as one control field:

| <u>CUSNO</u> | <u>ACTNO</u> | <u>REGNO</u> |
|--------------|--------------|--------------|
|--------------|--------------|--------------|

The first field defined on the input sheet is placed in the high-order position, and the last field is placed in the low-order position. All fields are placed according to the sequence in which they are defined on the Input Specifications sheet.

Using Split Control Fields with Field-Record Relation

The use of the Field-Record Relation specification in conjunction with control level indicators permits the programmer to define the same control level for split or non-split control fields in various record types. This is illustrated in Figure 53 and Figure 54.

The following points must be considered when using field-record relation indicators with split control field specifications.

1. The overall field length for each control level must be the same in each group.
2. The sum of the split control fields cannot be greater than 256 bytes.
3. Split control fields of any one level are arranged in storage in the same sequence as they appear in the input specifications.
4. If a field-record relation indicator is specified for a control level, the related field location is used for control purposes whenever that indicator is on. The examples that are identified by the circled numbers in Figure 53 illustrate this point.
 - 1 When either indicator 93 or 94 is on, L1 is composed of positions 1-5 and 6-10.
 - 2 When indicator 91, 93, or 94 is on, L2 is composed of positions 11-20.
 - 3 When indicator 92, 93, or 94 is on, L3 is composed of positions 21-40. A control level with a blank field-record relation indicator is used for control purposes when all indicators that condition any field with the same control level are off.
 - 4 To specify a portion of a split control field as being common to several record types, repeat that portion of the field definition with each record type indicator.

IBM CORPORATION
INPUT SPECIFICATIONS
 Form X24-3350-1
 Printed in U. S. A.

Page
 Program Identification

| Character Stacker Select Packed (P) | Field Location | | Field Name | Control Level (1-15) | Matching Fields or Chaining Fields | Field-Record Relation | Field Indicators | | | Sterling Sign Position |
|--|-------------------|----|------------|----------------------|---------------------------------------|--------------------------|---------------------|-------|---------------------|------------------------------|
| | From | To | | | | | Plus | Minus | Zero or Blank | |
| 11 | | | | | | | | | | |
| 12 | | | | | | | | | | |
| 13 | | | | | | | | | | |
| 14 | | | | | | | | | | |
| 15 | | | | | | | | | | |
| 16 | | | | | | | | | | |
| 17 | 10 | 15 | NAME | | | | | | | |
| 18 | 11 | 15 | NUM | | | | | | | |
| 19 | 17 | 19 | DEPT | | M1 | | | | | |
| 20 | 20 | 22 | REGION | | M2 | | | | | |
| 21 | 23 | 25 | DIVSON | | M3 | | | | | |

Figure 55. Using Matching Fields to Sequence Check in a Single Input File

descending sequence. In Figure 56 assume that the file has been specified in ascending sequence. The number 003051008 is lower than 005025003. Thus, the file is in ascending sequence.

NOTE: A chaining file may also be sequence checked. However, if it is the chaining field that is to be sequence checked, it must be defined twice using two different field names. The chaining indicator is entered in one field definition and the matching fields indicator in the other.

If the file is not in sequence the Halt indicator H0 is turned on. Unless this H0 indicator is turned off by a SETOF operation (See Turning Indicators On or Off) in the calculation specifications, the program terminates before the next input record is read.

Exit to External Translate Subroutine.
 If the sequence of the matching fields is not the same as the collating sequence of the System/360, the RPG program can provide an automatic exit to an external user subroutine that translates the sequence of the matching fields to the collating sequence of the system.

An entry in the RPG Processor Control Card is all that is required to cause the RPG program to branch to the subroutine. The automatic branch occurs after the input card is read in and before the RPG program checks the sequence of the matching field. The subroutine to translate the matching fields must use the predefined label ALTSEQ. The register conventions for this subroutine are the same as those for the EXIT operation. (See Exit to a Subroutine.) The address of the matching-field holding area will be contained in Register 1. The subroutine must place the translated fields back into the matching-fields hold area before it returns control to RPG.

Chaining Fields

The use of chaining files is explained in the section Chaining. Up to nine chaining fields are permitted in a record. In these columns enter the code that identifies the chaining field (C1 through C9).

FIELD-RECORD RELATION (63-64)

This specification is used when there are records in an OR relationship and the fields of the records are not in the same location. Enter in columns 63-64 the appropriate resulting indicator which will

Data Contained in First Record

DEPT 008
 REGION 051
 DIVSON 003

Data Contained in Second Record

DEPT 003
 REGION 025
 DIVSON 005

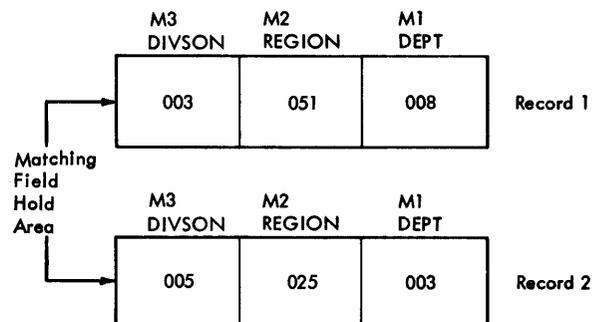


Figure 56. Comparing Matching Fields

be on when the field is used. An explanation of the use of this specification was contained in Records in an OR Relationship, and an example of this specification is provided in Figure 50.

Using Field-Record Relation with Chaining Files

An additional function of this specification is to selectively control chaining operations. (See Chaining for a general explanation of chaining. In order to understand this function, readers should be familiar with chaining operations and with the use of the Calculation Specifications sheet.)

If a chaining field is specified on a field description line and Field-Record Relation is blank, the chained record will be obtained whenever the record type (for the chaining field) is present. However, if a resulting indicator is placed in Field-Record Relation, then the chained record will be obtained only if the record-type is present and the resulting indicator (in Field-Record Relation) is on.

This feature provides the programmer with the ability to utilize chaining files on a selective basis. The function of this feature of Field-Record Relation is similar to the function of controlling calculations by the status of resulting indicators.

A variation of the use of this function of Field-Record Relation is to control two chained files with one chaining field. For example, the chaining field would be specified twice on the Input Specifications sheet. Each specification line would be conditioned by a separate resulting indicator. The particular chained record obtained would depend upon the status (on or off) of the appropriate resulting indicator.

FIELD INDICATORS (65-70)

This specification is used to test the status of a field when it is read into the system. Depending upon the status of the field— plus, minus, or zero or blank — it turns on an indicator that can be used to control calculation and output specifications, or even to stop the processing of the object program.

The entry for the specification is an indicator which will be turned on when the field specified on the line is plus, minus, or zero or blank.

Field Status Conditions

Plus. A plus condition occurs when the value of a numeric field is greater than 0.

Minus. A minus condition occurs when the value of the numeric field is less than zero.

Zero or Blank. A zero-or-blank condition occurs if a numeric field contains all zeros or blanks, or if an alphameric field contains all blanks. It is turned on if a numeric field is either +0 or -0.

NOTE: For alphameric fields, columns 65 through 68 must be blank.

Types of Indicator Codes Used

A 2-digit field indicator code is used for this specification. These codes, ranging from 01 to 99 can be defined one or more times on the form. If they are defined more than once, the second specification of this indicator resets it from the status it may have had by the previous specification for it.

NOTE: "Defining" these indicators means specifying them on the input form in Resulting Indicators or in Field Indicators. This should not be confused with "using" these indicators. "Using" these indicators means specifying them in Indicators on the calculation form or in Output Indicators on the output form as many times as required. In the latter case, they are merely tested to determine their status and not reset by the test.

Halt Indicators

There are ten additional indicator codes, known as Halt indicators that can be used in the RPG program. These indicators, designated as H0 through H9, halt the processing of the object program when error conditions (as determined by the programmer) have been detected. These indicators can also be used in calculation and output specifications sheets to control specifications and stop processing. For example, the status of a field can be tested, and depending upon the results of the tests, a halt indicator may be set on which terminates the object program.

If one of these indicators has been turned on during the processing of a record,

the object program terminates at the completion of the processing of that record. However, processing will not be interrupted if a halt indicator that has been turned on is turned off (in the program) before the program attempts to read the next input record.

The HO indicator can also be turned on automatically by the RPG program. The conditions that cause HO to be turned on automatically are listed in Appendix H. Unless HO is turned off by a SETOF operation (see Turning Indicators On or Off) in the calculation specifications, the program terminates before the next input record is read.

Use of Field Indicators

The field indicator 07 in Figure 57 is used to determine if FLDB contains zeros or blanks, and Indicator 06 is used to determine if the value contained in FLDA is positive. Indicator 07 and 06 would both be used to control functions in the calculation and output specifications that must be altered or modified depending on the status of FLDA and FLDB.

NOTE: Use of Indicators HO-H9 as field indicators testing zero or blank (69-70)

causes the program to terminate as the first record is read because any indicators used in (69-70) are initialized on.

How Field Indicators are Turned Off and On

Indicators used for testing for plus and minus conditions, are "set" (turned on or off) if their respective conditions occur when a record is read into the system. Each field indicator is related to only one record type. Therefore the indicators are not reset (turned on or off) until the related record type is read again or until the indicator number is defined in some other specification. One or more field indicators can be on at one time.

Indicators used for testing for zero or blank conditions are set in the same manner as those used for testing for plus and minus conditions. They can, however, be reset by one other condition. The output specification Blank After causes a field to be set to blanks or zeros (depending upon whether the field is alphabetic or numeric) after execution of the output operation specified. If the field being reset to blanks or zeros is an input field that is being tested for zero or blank, the field indicator specified for it is turned on when the field is set to blanks or zeros by the Blank After specification.

Any plus or minus indicators associated with the field are not turned off by virtue of the Blank After specification. All zero or blank indicators are initialized on at the beginning of the program to reflect the initial status of the associated fields. (All fields defined in an RPG program are initialized to zero if numeric or blank if alphameric.) The indicators remain on until the status of the associated fields change as a result of an input record being read or a calculation being performed.

STERLING (71-74)

Enter in these columns the position in the record that contains the sign of the sterling field. If the sign is in the normal position, enter an S in column 74. Leading zeros may be omitted. Leave these columns blank if the sterling specification is not used. Additional information on Sterling is in Appendix D. Sterling Routines for the Report Program Generator.

SUMMARY OF INPUT SPECIFICATIONS

This concludes the description of the input specifications. Additional information on

ES CORPORATION Form X24-3350-1 Printed in U. S. A.

INPUT SPECIFICATIONS

Page 1 2 Program Identification 75 76 77 78 79 80

| Field Location | Field Name | | Field Indicators | Sterling Sign Position |
|----------------|------------|-----------|------------------|------------------------|
| | From | To | | |
| 1 | 4 | DIVSON L3 | | |
| 5 | 8 | DEPT L2 | | |
| 9 | 14 | EMPNO L2 | | |
| 15 | 28 | NAME | | |
| 46 | 50 | FLDA | 06 | |
| 56 | 60 | FLDB | | 07 |
| 66 | 70 | FLDC | | |

Figure 57. Example of Field Indicators Specifications

matching or chaining fields may be found in Processing Multiple Input Files. The input specifications listed below are used with the calculation and output forms.

Resulting Indicator
Field Name
Control Level

Matching Fields Field Indicators

The use and function of these specifications may become more apparent to the reader after the descriptions of the calculation and output specifications have been read.

SPECIFYING THE KIND OF CALCULATION

In order to describe the kind of calculations to be performed, the following conditions must be considered.

1. The fields to be used
2. The arithmetic operation to be performed
3. The disposition of the results

FACTOR 1 (18-27)

This specification can be a field name or a literal. If it is a field name it must have been defined on the Input Specifications sheet, or it must be defined in the result field of a calculation. The field name must be left-justified, and the first character must be alphabetic.

Literal

A literal is the actual data to be operated on, rather than a name representing the location of data. Literals may be numeric or alphabetic. Numeric literals may be up to ten characters long. Alphameric literals may be up to eight characters long.

Numeric Literals

A numeric literal can consist of any combination of the numbers 0-9. One decimal symbol and/or one plus sign or one minus sign may also be used.

Rules for Forming Numeric Literals

1. Blanks must not appear within a numeric literal.
2. The sign, if present, must be the left-most character. If a literal is unsigned, it is treated as a positive literal.
3. The decimal symbol can appear anywhere in the literal.
4. Numeric literals must not be enclosed in apostrophe symbols.

Alphameric Literals

Any set of consecutive characters enclosed in a set of apostrophe symbols is treated as an alphameric literal. (The apostrophe is a 5,8 punch.) Alphameric literals may not be used in arithmetic operations.

Rules for Forming Alphameric Literals

1. Any character may be used in an alphameric literal. Blanks are treated as valid characters in the body of the literal.
2. Alphameric literals must be enclosed in a set of apostrophe symbols.
3. An apostrophe symbol may be contained within a literal by entering two consecutive apostrophe symbols within the literal. For example, the literal o'clock would be coded as 'o'clock'.

Figure 61 illustrates entries for Factor 1. The numbers in circles refer to the items listed below.

1. GROSS could be a field name specified on the input sheet.
2. NETAMT could have been specified as the result field of the previous calculation.
3. This numeric literal could be used in the program to determine if specific fields in the input records were higher or lower than this number. The position of the decimal symbol must be indicated if the number is not a whole number.
4. Alphameric literals, like the one in this example, can be used to compare against a data field in the input records to perform certain types of calculations only upon records representing the month of January.

The description of Factor 1 also applies to Factor 2.

| IBM | | INTERNATIONAL BUSINESS | | REPORT PROGRAM GENERATOR | | IBM Sy | | |
|------------------|-----------|---------------------------|------------|--------------------------|-----|---------------|-----------|----------|
| Date _____ | | Program _____ | | Punching Instruction | | Graphic Punch | | |
| Programmer _____ | | | | | | | | |
| Line | Form Type | Control Level (00,10, LB) | Indicators | | | Factor 1 | Operation | Factor 2 |
| | | | And | And | | | | |
| | | | Not | 2 | Not | | | |
| | | | 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 | | | | |
| 0 1 | C | | | | | GROSS | | ① |
| 0 2 | C | | | | | NETAMT | | ② |
| 0 3 | C | | | | | 12500.00 | | ③ |
| 0 4 | C | | | | | 'JANUARY' | | ④ |
| 0 5 | C | | | | | | | |

Figure 61. Factor 1

Summary of Factor 1 and Factor 2

1. Enter either the name or the literal that is Factor 1 or Factor 2 in columns 18-27 or 33-42.
2. If Factor 1 or Factor 2 contains the name of a field, the field must be defined in either:
 - a. Columns 53-58 (Field Name) of the Input Specifications sheet.
 - b. Columns 43-48 (Result Field) of the Calculation Specifications sheet.
3. A name cannot exceed six characters. Special characters and blanks must not be used.
4. A numeric literal cannot exceed ten characters; an alphameric literal cannot exceed eight characters.
5. Entries in Factor 1 or Factor 2 must be left-justified.

OPERATION (COLUMNS 28-32)

Entries in these columns specify the operations to be performed using Factor 1, Factor 2, and Result Field. Each operation is specified by placing the operation code in Operation (columns 28-32).

Arithmetic Operations

Add
 Zero and Add
 Subtract
 Zero and Subtract
 Multiply
 Divide
 Move Remainder

Code

ADD
 Z-ADD
 SUB
 Z-SUB
 MULT
 DIV
 MVR

Move Operations

Move
 Move Left
 Move High-to-Low Zone
 Move Low-to-High Zone
 Move High-to-High Zone
 Move Low-to-Low Zone

Code

MOVE
 MOVEL
 MHLZO
 MLHZO
 MHHZO
 MLLZO

Testing or Compare Operations

Compare
 Test Zone

Code

COMP
 TESTZ

Branching and Exit Operations

Branching (or GOTO)
 Providing a Label for GOTO
 Exit to a Subroutine
 RPG Label
 User Label

Code

GOTO
 TAG
 EXIT
 RLABL
 ULABL

Turning Indicators On or Off

Set Indicators On
 Set Indicators Off

Code

SETON
 SETOF

Table Operations

Table Lookup

Code

LOKUP

Conversion Routine Operations

RPG Conversion Routine
 End of RPG Conversion
 External Conversion Routine
 Record Key

Code

RPGCV
 ERPGC
 EXTCV
 KEYCV

Arithmetic Operations

The fields or literals involved in these operations may contain numeric characters only. All arithmetic operations are performed with automatic decimal alignment. Resulting indicators can be used with all arithmetic operations.

No arithmetic overflow will be sensed by RPG. The length of a field involved in arithmetic operations can be up to 15 digits (this includes decimal alignment when necessary). The resulting field length after decimal alignment must not be greater than 15 digits.

Add (ADD)

This operation causes the contents of the field or the literal in Factor 2 to be added, algebraically, to the contents of the field or literal in Factor 1. The result of the operation is placed in the result field specified in Result Field (columns 43-48).

Zero and Add (Z-ADD)

This operation causes the field specified in Result Field to be set to zeros and then causes the data contained in the numeric literal or the field in Factor 2 to be placed in the Result Field. Factor 1 is not used in this operation.

Subtract (SUB)

This operation causes the contents of the field or literal in Factor 2 to be subtracted, algebraically, from the contents of the field or literal in Factor 1. The result of this operation is placed in the Result Field specified.

Zero and Subtract (Z-SUB)

This operation causes the negative of the number contained in the literal or the field in Factor 2 to be placed in the result field specified. This operation is performed after the result field has been set to zeros. Factor 1 is not used in this operation.

Multiply (MULT)

This operation causes the contents of the field or literal in Factor 1 to be multiplied algebraically by the contents of the field or the literal in Factor 2. The result of this operation is placed in the result field specified.

Divide (DIV)

This operation causes the contents of the field or literal in Factor 1 to be divided by the contents of the field or literal in Factor 2. The result of this operation (quotient) is placed in the specified result field. The contents of the field or the literal in Factor 2 cannot be zero.

If factor 2 is zero, a program check and an abnormal end-of-job will result. The following field length restrictions apply to this operation:

$$L_1 + (D_2 - D_1 + D_r) \leq 15$$

$$L_2 - (D_2 - D_1 + D_r) \leq 15$$

and if half-adjusting is specified

$$L_1 + (D_2 - D_1 + D_r) \leq 14$$

where

L_1 = length of Factor 1 (dividend)

L_2 = length of Factor 2 (divisor)

D_1 = decimal positions of Factor 1

D_2 = decimal positions of Factor 2

D_r = decimal positions of Result Field

NOTE: Invalid results are obtained if the formula is violated.

Any remainder that results from this operation is lost unless the move-remainder operation is specified as the next operation in the program.

NOTE: If a move-remainder operation follows a divide operation, the result in the divide operation cannot be half-adjusted.

Move Remainder (MVR)

This operation is used to move the remainder from a divide operation to a separate field. If MVR is used, it must immediately follow the divide operation. The divide may not be half adjusted. Figure 62 shows an example of the MVR operation. The remainder is placed in a field named STORE.

| Line | | Form Type | Control Level (O, L, E) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Half Adjust (H) | Resulting Indicators | | | Comments |
|------|---|-----------|-------------------------|------------|------------|-----------|-------------|-----------|----------|--------------|--------------|-----------------------------------|----------------------|-------|---------------|----------|
| 3 | 4 | | | And | And | Not | | | | | | | Plus | Minus | Zero or Blank | |
| 5 | 6 | 1 | 2 | 3 | High 1 > 2 | Low 1 < 2 | Equal 1 = 2 | | | | | | | | | |
| 01 | 0 | C | | 12 | | | FIELD1 | DIV | FIELD2 | SAVE | | | | | | |
| 02 | 0 | C | | | | | | MVR | | STORE | | | | | | |
| 03 | | C | | | | | | | | | | | | | | |

Figure 62. Using MVR Operation Code

The field that is to contain the remainder must be specified in Result Field.

The value of the remainder can be determined by the following formula:

$$R = (\text{Dividend}) - [(\text{Divisor}) \times (\text{Quotient})]$$

For the above equation to be valid in a divide operation involving factors containing decimal positions, the result field that is to contain the remainder must provide for the decimal positions in the remainder based on the sum of ($D_2 + D_r$) or D_1 , whichever is greater.

NOTE: Resulting indicators can be specified for the MVR operation.

Move Operations

For the MOVE and MOVEL operations, numeric fields may be changed to alphameric fields and alphameric fields may be changed to numeric fields. To change a numeric field to an alphameric field, Factor 2 is numeric, and the result field is specified as alphameric. To change an alphameric field to a numeric field, Factor 2 is alphameric, and the result field is specified as numeric. No decimal alignment is performed when a move operation is used.

Resulting indicators can be used with all move operations.

Move (MOVE)

This operation code causes data characters (starting at the rightmost position) to be moved from the field or literal contained in Factor 2 to the rightmost positions of the result field.

If Factor 2 is longer than the result field, the excess leftmost positions of Factor 2 are not moved as illustrated in Figure 63.

If the result field is longer than the field specified by Factor 2, the positions to the left of the data that is moved remain undisturbed as illustrated in Figure 64.

Factor 1 is not used in this operation.

Move Left (MOVEL)

This operation code causes data characters (starting at the leftmost position) to be moved from the field or literal contained in Factor 2 to the leftmost positions of the result field.

If Factor 2 is longer than the result field, the excess rightmost positions of Factor 2 are not moved. This is illustrated in Figure 65.

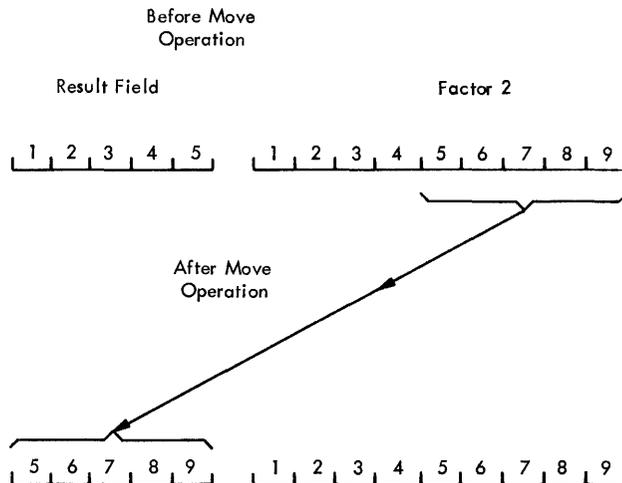


Figure 63. Move Operation -- Factor 2 Longer than Result Field

If the Result Field is longer than the field specified by Factor 2, the positions in the Result Field to the right of the data that was moved remain undisturbed as shown in Figure 66, insert A.

When moving data to a numeric field, the determination of the sign is an important consideration. In Figure 66, inserts B and C, the underlined digits in Factor 2 indicate the position of the sign. If the Result Field is longer than Factor 2 as shown in Figure 66, insert B, the original sign of the Result Field is retained. However, if Factor 2 is as long or longer than the Result Field as shown in Figure 66, insert C, the sign of Factor 2 is assumed by the Result Field and is indicated by the underlined digit in the Result Field.

Factor 1 is not referenced by this operation.

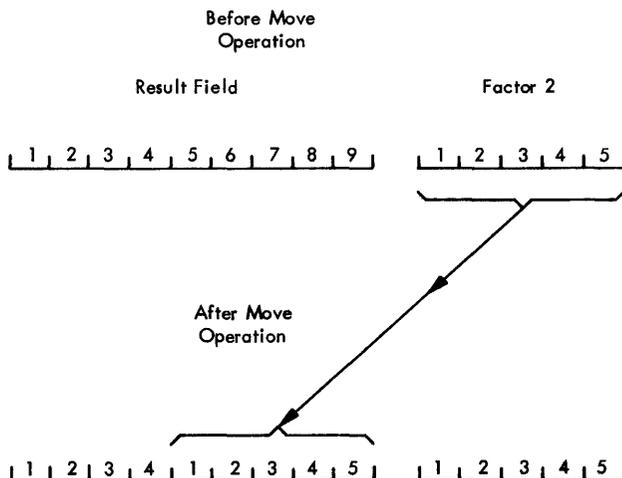


Figure 64. Move Operation -- Factor 2 Shorter than Result Field

| Operator | Operation | Factor 2 | Result Field | Field Length | Decimal Position Half Adjust (H) | Resulting Indicators | | | Column | |
|----------------|----------------|-------------------------------|-------------------|--------------|-------------------------------------|----------------------|--------------|----------------|--------|-------------------------|
| | | | | | | Plus | Minus | Zero | | |
| | | | | | | Compare | | | | |
| | | | | | | High 1 > 2 | Low 1 < 2 | Equal 1 = 2 | | |
| 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 | 58 59 | 60 61 62 63 64 65 66 67 |
| | MOVE | FLDA | FLDB | 4 | 0 | | | | | |

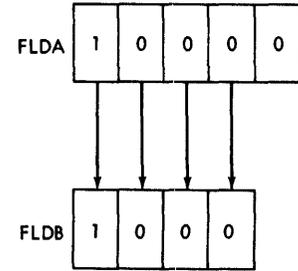


Figure 65. Using the MOVE Operation Code

numeric data, this operation can still be performed; however, the numeric field must have been specified as an alphameric field on the Input Specifications sheet.

The result field can be numeric or alphameric. A result field specified as numeric contains an F zone for a plus sign or a D zone for a minus sign after this operation.

Figure 68 illustrates a Move High-to-Low Zone operation (alphameric to alphameric).

Move High-To-Low Zone (MHLZO)

This operation moves the zone at the leftmost position of Factor 2 to the rightmost position of the result field.

Figure 67 illustrates the movement of zones for all four move zone operations.

Factor 2 can only be alphameric. If the zone to be moved is located over

Move Low-To-High Zone (MLHZO)

This operation moves the zone at the rightmost position of Factor 2 to the leftmost position of the result field. Factor 2 can be numeric or alphameric, but the result field must be alphameric.

| | Punching Instruction | Graphic Punch | Operation | Factor 2 | Result Field | Field Length | Decimal Position Half Adjust (H) | Resulting Indicators | | | Page |
|---|----------------------|---------------|-----------|----------|--------------|--------------|-------------------------------------|----------------------|--------------|----------------|------------------------------|
| | | | | | | | | Plus | Minus | Zero | |
| | | | | | | | | Compare | | | |
| | | | | | | | | High 1 > 2 | Low 1 < 2 | Equal 1 = 2 | |
| Ⓐ | | | MOVE | 'DATA' | FLDA | 6 | | | | | |
| | | | | | | | | | | | Factor 2 |
| | | | | | | | | | | | Result Field (after move) |
| | | | | | | | | | | | DATA |
| | | | | | | | | | | | DATAxx |
| Ⓑ | | | MOVE | FLDA | FLDB | 8 | 1 | | | | 9 8 7.6 5 |
| | | | | | | | | | | | 9 8 7 6 5 x x.x |
| Ⓒ | | | MOVE | 1357.65 | FLDC | 4 | 2 | | | | 1 3 5 7.6 5 |
| | | | | | | | | | | | 1 3.5 7 |

Figure 66. Additional Functions of the MOVE Operation

| | | |
|-----------------------|--------------------------|---|
| MLLZO | | |
| Factor 2 Result Field | Alphameric Alphameric | Bits 0-3 of rightmost byte of Factor 2 are moved to bits 0-3 of rightmost byte of Result Field. |
| Factor 2 Result Field | Alphameric Numeric | Bits 0-3 of rightmost byte of Factor 2 are moved to bits 4-7 of the rightmost byte of the Result Field. |
| Factor 2 Result Field | Numeric Alphameric | Bits 4-7 of rightmost byte of Factor 2 are moved to bits 0-3 of rightmost byte of the Result Field. |
| Factor 2 Result Field | Numeric Numeric | Bits 4-7 of rightmost byte of Factor 2 are moved to bits 4-7 of rightmost byte of the Result Field. |
| MHLZO | | |
| Factor 2 Result Field | Alphameric Numeric | Bits 0-3 of leftmost byte of Factor 2 are moved to bits 4-7 of rightmost byte of Result Field. |
| Factor 2 Result Field | Alphameric Alphameric | Bits 0-3 of leftmost byte of Factor 2 are moved to bits 0-3 of rightmost byte of the Result Field. |
| MLHZO | | |
| Factor 2 Result Field | Alphameric Alphameric | Bits 0-3 of rightmost byte of Factor 2 are moved to bits 0-3 of leftmost byte of Result Field. |
| Factor 2 Result Field | Numeric Alphameric | Bits 4-7 of rightmost byte of Factor 2 are moved to bits 0-3 of leftmost byte of the Result Field. |
| MHHZO | | |
| Factor 2 Result Field | Alphameric Alphameric | Bits 0-3 of leftmost byte of Factor 2 are moved to bits 0-3 of leftmost byte of Result Field. |

Figure 67. Move Zone Operations

Move High-To-High Zone (MHHZO)

This operation moves the zone at the leftmost position of Factor 2 to the leftmost position of the result field. Factor 2 and the result field must be alphameric.

Move Low-To-Low Zone (MLLZO)

This operation moves the zone at the rightmost position of Factor 2 to the rightmost position of the result field. Factor 2 and the result field are alphameric or

numeric. A result field specified as numeric contains an F zone for a plus sign or a D zone for a minus sign after this operation.

Testing or Compare Operations

Compare (COMP)

This operation causes the contents of the field or the literal in Factor 1 to be compared against the contents of the field

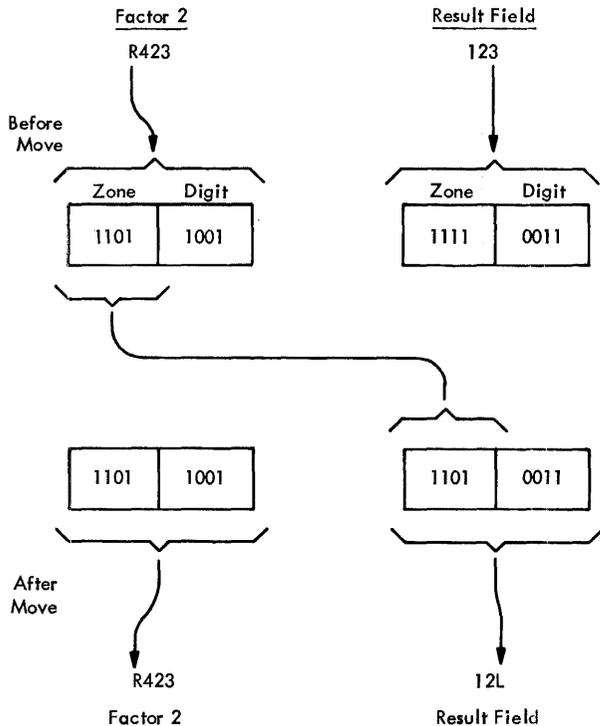


Figure 68. Move High-to-Low Zone Operation

or literal in Factor 2. The outcome of this operation can be used to turn on an indicator that has been specified in columns 54-59 (Resulting Indicators High, Low, or Equal). These indicators are turned on as follows:

High: Factor 1 is greater than Factor 2.
 Low: Factor 1 is less than Factor 2.
 Equal: Factor 1 is equal to Factor 2.

This operation is used to make comparisons to alter or modify subsequent calculations. No result field is specified.

- The Factor-1 and Factor-2 fields are aligned according to whether they are numeric or alphameric. If numeric fields are compared, fields of unequal length are aligned to the implied decimal point.
- Missing digits in numeric fields are assumed to be zeros.
- If alphameric fields are compared, fields of unequal length are aligned to their leftmost characters and the unused positions are filled with blanks.

- The alphameric compare operation is based upon the internal collating sequence of the system.
- For equal alphameric fields, the maximum field length is 256 characters.
- For unequal alphameric fields, the maximum field length is 200 characters.
- An alphameric field and a numeric field should not be compared because the results of such a comparison are unpredictable.

All numeric comparisons are algebraic. An absolute comparison can be performed by means of a short routine programmed to meet the user's requirements. Figure 69 shows an example of comparing the absolute value of a sum to a literal.

Test Zone (TESTZ)

This operation is used to test the zone of the leftmost position of the alphameric field that is entered in the result field. The format of a Test Zone operation is shown in Figure 70.

If the result of the test is a 12-zone (&, A through I, 0), the indicator specified in columns 54-55 will be turned on. If the result of the test is an 11-zone (-, J through R, 0), the indicator specified in columns 56-57 is turned on. Any other zone turns on the indicator specified in columns 58-59.

Figure 71 shows an example of this operation. When Indicator 25 is on, the field DATA is tested. If the leftmost position has a 12-zone, Indicator 01 is turned on. If the position has an 11-zone, Indicator 02 is turned on.

Branching and Exit Operations

Exit to a Subroutine (EXIT)

This operation code enables the programmer to transfer control from the RPG program to a user subroutine. Factor 2 contains the name of the subroutine. The name of the subroutine cannot be greater than six alphameric characters; the first character must be alphabetic. Factor 1 and the Result Field are not used. See EXIT to a User's Routine for a complete discussion of this operation.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
IBM System/360

Form X24-3351-1
Printed in U.S.A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Control Level (00-19, LR) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Half Adjust (H) | Resulting Indicators | | | Comments | | | |
|------|-----------|---------------------------|------------|-----|-------|----------|-----------|----------|--------------|--------------|-----------------------------------|----------------------|-------|---------------|----------|------------|-----------|-------------|
| | | | Not | And | And | | | | | | | Plus | Minus | Zero or Blank | | | | |
| | | | | | | | | | | | | | | | | Compare | | |
| | | | | | | | | | | | | | | | | High 1 > 2 | Low 1 < 2 | Equal 1 = 2 |
| 0 1 | C | | | | FACT1 | ADD | FACT2 | FACT2 | 82 | | | 21 | | | | | | |
| 0 2 | C | | 21 | | | Z-SUB | FACT2 | CFLD | 82 | | | | | | | | | |
| 0 3 | C | | W21 | | | Z-ADD | FACT2 | CFLD | | | | | | | | | | |
| 0 4 | C | | | | CFLD | COMP | 10000.0 | | | | | 313233 | | | | | | |
| 0 5 | C | | | | | | | | | | | | | | | | | |

Figure 69. Example of an Absolute Compare Routine

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
IBM System/360

Form X24-3351-1
Printed in U.S.A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Control Level (00-19, LR) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Half Adjust (H) | Resulting Indicators | | | Comments | | | |
|------|-----------|---------------------------|------------|-----|-------|----------|-----------|----------|--------------|--------------|-----------------------------------|----------------------|-------------|---------------|----------|------------|-----------|-------------|
| | | | Not | And | And | | | | | | | Plus | Minus | Zero or Blank | | | | |
| | | | | | | | | | | | | | | | | Compare | | |
| | | | | | | | | | | | | | | | | High 1 > 2 | Low 1 < 2 | Equal 1 = 2 |
| 0 1 | C | | | | | TESTZ | | FLDNAM | | | | | | | | | | |
| 0 2 | C | | | | | | | | | | | | Other Zones | | | | | |
| 0 3 | C | | Optional | | Blank | | | Blank | | | | | 11-Zone | | | | | |
| 0 4 | C | | | | | | | | | | | | 12-Zone | | | | | |
| 0 5 | C | | | | | | | | | | | | Blanks | | | | | |
| 0 6 | C | | | | | | | | | | | | Required | | | | | |

Figure 70. Format of the TESTZ Operation

The LOKUP operation code causes the contents of the field or literal contained in Factor 1 to be used as the search argument. Factor 2 contains the name of the argument table to be searched, and the Result Field contains the table name from which the associated function will be obtained. Decimal alignment is not performed for this operation. (The Result Field may be left blank if the associated function is to be located but not retrieved from the table.)

The use of two resulting indicators causes the RPG program to look up that table entry that is high or equal, or low or equal in relation to the search argument.

After the lookup operation is completed, the function that is retrieved is placed in a special holding area for the function table. The name of this area is the same as the name of the function table.

To utilize the function in another operation (for example in an arithmetic operation), the name of the function table is specified in Factor 1 or Factor 2. In this case, the function table name (in Factor 1 or Factor 2) refers to the special holding area in the function table.

To update the function table (for example, a move operation to replace the old function with the new updated function) the name of the function table is specified in the Result Field. The new function is then placed in its proper place in the function table and in the special holding area.

After each table lookup operation, the retrieved function should be used (or moved from the special holding area to another location) before the next table lookup operation is performed. Each subsequent lookup operation overlays the function obtained from the previous lookup operation.

See Using Tables in the Object Module for additional information and examples of table lookup operations.

Conversion Routine Operations

RPG Conversion (RPGCV)

This operation code is used to indicate that the track-address conversion routine is coded on the RPG Calculation Specifications sheet. Factor 1 contains the name (label) of the conversion routine. This name must also be specified on the File Extension sheet in columns 27-32. The Result Field contains the name of the field that

contains the relative track address. This field must be alphameric and must have a length of 3. Indicators and Factor 2 are not used.

End of RPG Conversion (ERPGC)

This entry terminates the conversion-step entries that have been coded on the Calculation Specifications sheet. No other entries are necessary.

Indicators, Factor 1, Factor 2, and Result Field are not used.

External Conversion Routine (EXTCV)

This entry is used to indicate that the track-address conversion routine is external to the RPG language.

Factor 1 contains the label specified on the File Extension sheet in columns 27-32.

The Result Field contains the name of the field that will contain the track address. This name is defined in the RPG program by this operation and must not be defined in the external routine. The field must be alphameric and must have a length of 3.

Factor 2 must contain the name of the external conversion subroutine that the RPG program branches to. The specification Indicators is not used.

Record Key (KEYCV)

This operation code establishes the name of the field that is to contain the key of the disk record. (It is used only when records are retrieved using record key data.) The code KEYCV is placed in Operation (columns 28-32) and the name of the field is placed in Result Field (columns 43-48). The field length and decimal positions must be specified if the field has not been defined previously. Indicators, Factor 1, and Factor 2 are not used.

The operation must follow the RPGCV or EXTCV operation. The name of the field that will contain the record key is defined in the RPG program by this operation and must not be defined in the external conversion routine.

Table 1 is a summary of the specification entries required for the operation codes just described.

Table 1. Summary of Operation Specifications

O = Optional R = Required b = blank

| Operation | Control Level | Indicators | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions | Half Adjust | Resulting Indicators |
|-----------------------------|---------------|------------|----------|-----------|----------|--------------|--------------|-------------------|-------------|----------------------|
| Add | O | O | R | ADD | R | R | O | O | O | O |
| Zero and Add | O | O | b | Z-ADD | R | R | O | O | O | O |
| Subtract | O | O | R | SUB | R | R | O | O | O | O |
| Zero and Subtract | O | O | b | Z-SUB | R | R | O | O | O | O |
| Multiply | O | O | R | MULT | R | R | O | O | O | O |
| Divide | O | O | R | DIV | R | R | O | O | O | O |
| Move Remainder | O | O | b | MVR | b | R | O | O | b | b |
| Move | O | O | b | MOVE | R | R | O | b | b | b |
| Move Left | O | O | b | MOVEL | R | R | O | b | b | b |
| Move High-to-Low Zone | O | O | b | MHLZO | R | R | O | b | b | b |
| Move Low-to-High Zone | O | O | b | MLHZO | R | R | O | b | b | b |
| Move High-to-High Zone | O | O | b | MHHZO | R | R | O | b | b | b |
| Move Low-to-Low Zone | O | O | b | MLLZO | R | R | O | b | b | b |
| Compare | O | O | R | COMP | R | b | b | b | b | R |
| Test Zone | O | O | b | TESTZ | b | R | R | b | b | R |
| Exit to a Subroutine | O | O | b | EXIT | R | b | b | b | b | b |
| RPG Label | O | b | b | RLABL | b | R | O | O | b | b |
| User's Label | O | b | b | ULABL | b | R | R | R | b | b |
| Branching or GOTO | O | O | b | GOTO | R | b | b | b | b | b |
| Providing a Label for GOTO | O | b | R | TAG | b | b | b | b | b | b |
| Set Indicators ON | O | O | b | SETON | b | b | b | b | b | R |
| Set Indicators OFF | O | O | b | SETOF | b | b | b | b | b | R |
| Table Lookup | O | O | R | LOKUP | R | O | O | O | b | R |
| RPG Conversion | O | b | R | RPGCV | b | R | R | R | b | b |
| End of RPG Conversion | O | b | b | ERPGC | b | b | b | b | b | b |
| External Conversion Routine | O | b | R | EXTCV | R | R | R | b | b | b |
| Record Key | O | b | b | KEYCV | b | R | O | O | b | b |

module considers the number of decimal positions in both factors of an arithmetic operation and automatically "shifts" the factors or the results to provide the correct number of decimal positions.

In Figure 73 each result field has two decimal positions.

If the result field is alphameric, this column must be left blank.

HALF ADJUST (53)

This specification is used to half-adjust, or round, the result field. Enter an H in this column if the data in the result field is to be half-adjusted.

Half-adjusting is accomplished in the object program by adding an absolute value of 5 to the right of the last position retained in the result field.

In Figure 73, DISCNT is half-adjusted.

If the result field is an alphameric field, this specification must be left blank.

If the number of decimal positions in the arithmetic result is less than or equal to the decimal positions specified for the pertinent result field, the half-adjustment specification has no effect.

This completes the description of the specifications required for determining the kind of calculations to be performed.

TESTING THE RESULTS OF CALCULATIONS

The last category of specifications for the calculation form is Resulting Indicators.

RESULTING INDICATORS (54-59)

This specification may be used to test the value of a result field after the completion of an operation. As the result of this test, an indicator is turned on which can be used to control subsequent calculation operations or to control output operations. The specification is used in five ways:

1. To determine whether the result of an arithmetic operation is plus, minus, or zero. (In the case of half-adjustment, the resulting indicator refers to the result after half-adjustment.)
2. To test the result of a compare operation to determine if:

Factor 1 > Factor 2 -- High
Factor 1 < Factor 2 -- Low
Factor 1 = Factor 2 -- Equal

3. To define the type of LOKUP operation to determine:
 - a. If the argument next-higher than the search argument is found.
 - b. If the argument next-lower than the search argument is found.
 - c. If the argument equal to the search argument is found.

NOTE: If an equal-search resulting indicator is specified, it takes precedence over either high or low indicators if an equal-table value exists.

4. To define a TESTZ operation as to what type of zone is to be tested.
5. To define SETOF and SETON operations as to what indicators are to be turned off or on.

The entries for this specification can be any of the indicator codes 01 through 99 and the halt indicators H0 through H9. They can be defined one or more times on the form. If these indicators are defined more than once, a subsequent specification of the indicator resets it from the status it may have had by the previous specification for it.

NOTE: "Defining" these indicators means specifying them as resulting indicators or field indicators in the input specifications, or as resulting indicators in the calculation specifications. This should not be confused with "using" these indicators. "Using" these indicators means specifying them in Indicators on the calculation form or in Output Indicators on the output form as many times as required. In the latter case they are merely tested to determine their status and are not reset by the test.

NOTE: Resulting indicators are not reset (turned on or off) until the next time a calculation is performed for which the program specifies the indicator as a resulting indicator. This means that one or more resulting indicators can be on at one time.

NOTE: An indicator specified in columns 58-59 (Zero or Blank) for ADD, Z-ADD, SUB, Z-SUB, MULT, DIV, MVR, MOVE, and MOVE1 is initialized on.

A resulting indicator used to test for a zero balance can be reset by one other condition. The output specification Blank After causes a numeric field to be set to zeros after execution of the output operation specified. If this field is also a Result Field being tested for zero, the resulting field indicator specified is turned on when the field is set to zeros by the Blank After specification. If the field is also a result field being tested for plus or minus, the resulting field indicators specified are not turned off when the field is set to zeros by the Blank After specification.

Table 2 illustrates the various conditions that cause resulting indicators to be turned on.

On the first line in Figure 74, DISCNT is subtracted from GROSS and the result is stored in NETAMT. If the answer is a minus number, Indicator 10 is set on. If the answer is zero, Indicator 15 is set on.

On the second line in Figure 74, the literal JANUARY is compared against the contents of DATE. If the result is equal, Indicator 24 is turned on.

COMMENTS (60-74)

Positions 60 through 74 of the form are not required by the program. Data placed in these positions will be printed as a separate field during the compilation of the

object program. An asterisk in column 7 is not required if this type of comment is in a line containing specifications.

USING THE CALCULATION SPECIFICATIONS SHEET

Figure 75 illustrates entries on the Calculation Specifications sheet that are used in part of a payroll application. The entries on the sheet are discussed by line number.

NOTE: The blank spaces signify that additional calculations have been specified, but in this example they have been omitted.

| Line Number | Explanation |
|-------------|--|
| 01 | The program branches to GRSPAY from some other detail calculation (not shown in this example). |
| 02 | The number of hours the employee worked is compared with the literal 40. If the employee worked more than 40 hours, Indicator 20 is turned on. If the employee worked less than 40 hours, Indicator 22 is turned on. |
| 03-05 | If Indicator 20 is on, three calculations are performed. The literal 40 is subtracted from the |

Table 2. How Resulting Indicators are Turned On

| | | Columns 54-55 | | Columns 56-57 | | Columns 58-59 | |
|---------------------------|-----------------------------|-----------------------|--------------------------------------|------------------------|-------------------------------------|----------------------|------------------------------|
| | | PLUS | HIGH | MINUS | LOW | ZERO OR BLANK | EQUAL |
| Arithmetic Operation → | If the Result Field has a : | Plus Value (Except 0) | - | Minus Value (Except 0) | - | Zero Value (0 And 0) | |
| Table Lookup → | If the table argument is : | - | Next higher than the search argument | - | Next lower than the search argument | - | Equal to the search argument |
| Compare Operation → | If factor 1 is : | - | Greater than factor 2 | - | Less than factor 2 | - | Equal to factor 2 |

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
 IBM System 360

Form X24-3351-1
 Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Control Level (0, U, L) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Half Adjust (H) | Resulting Indicators | | | Comments |
|------|-----------|-------------------------|------------|-----|-----------|----------|-----------|----------|--------------|--------------|-----------------------------------|----------------------|-------|---------------|----------|
| | | | And | And | Not | | | | | | | Plus | Minus | Zero or Blank | |
| | | | | | | | | | | | | | | | |
| | | | Not | Not | Not | | | | | | | | | | |
| 01 | C | | | | GROSS | SUB | DISCNT | NETAMT | 82H | | | 1015 | | | |
| 02 | C | | | | 'JANUARY' | COMP | DATE | | | | | 24 | | | |
| 03 | C | | | | | | | | | | | | | | |

Figure 74. Result Field Indicators

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
 IBM System/360

Form X24-3351-1
 Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Control Level (0, U, L) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Half Adjust (H) | Resulting Indicators | | | Comments |
|------|-----------|-------------------------|------------|-----|--------|---------------------------|-----------|----------|--------------|--------------|-----------------------------------|----------------------|-------|---------------|----------|
| | | | And | And | Not | | | | | | | Plus | Minus | Zero or Blank | |
| | | | | | | | | | | | | | | | |
| | | | Not | Not | Not | | | | | | | | | | |
| 01 | C | | | | GRSPAY | TAG | | | | | | | | | |
| 02 | C | | | | HOURS | COMP | 40 | | | | | 2022 | | | |
| 03 | C | | 20 | | HOURS | SUB | 40 | OVERHR | 31 | | | | | | |
| 04 | C | | 20 | | RATE | MULT | OVERHR | SAVE | 62H | | | | | | |
| 05 | C | | 20 | | SAVE | MULT | 1.5 | SAVE | H | | | | | | |
| 06 | C | | N22 | | RATE | MULT | 40 | GROSS | 62H | | | | | | |
| 07 | C | | N22 | | SAVE | ADD | GROSS | GROSS | | | | | | | |
| 08 | C | | N22 | | | GOTO | FICA | | | | | | | | |
| 09 | C | | | | | (ADDITIONAL CALCULATIONS) | | | | | | | | | |
| 10 | C | | | | FICA | TAG | | | | | | | | | |
| 11 | C | | | | GROSS | MULT | .03 | DFICA | 62H | | | | | | |
| 12 | C | | | | YDFICA | ADD | DFICA | HOLD | 62H | | | | | | |
| 13 | C | | | | HOLD | COMP | 144.00 | | | | | 2121 | | | |
| 14 | C | | 21 | | | GOTO | ADFICA | | | | | | | | |
| 15 | C | | | | 144.00 | SUB | YDFICA | DFICA | | | | | | | |
| 16 | C | | | | ADFICA | TAG | | | | | | | | | |
| 17 | C | | | | YDFICA | ADD | DFICA | YDFICA | | | | | | | |
| 18 | C | | | | | (ADDITIONAL CALCULATIONS) | | | | | | | | | |
| 19 | C | | | | WHTAX | TAG | | | | | | | | | |
| | C | | | | | (ADDITIONAL CALCULATIONS) | | | | | | | | | |

Figure 75. Using the Calculation Specifications Sheet

| <u>Line Number</u> | <u>Explanation</u> | <u>Line Number</u> | <u>Explanation</u> |
|--------------------|--|--------------------|---|
| | number of hours worked, and the overtime hours are placed in the field OVERHR, which is a three-position field with one decimal position. RATE is multiplied by OVERHR and the result is placed in the field SAVE, which is a six-position field with two decimal positions. SAVE is multiplied by the literal 1.5 (which is the overtime premium rate). The result is placed back in SAVE, and it is half-adjusted. | | the field DFICA, which is a six-position field with two decimal positions. The result is half-adjusted. |
| 06 | RATE is multiplied by 40, and the result is stored in GROSS. This operation is performed whether or not the employee worked any overtime. It is not performed if the employee has worked less than 40 hours. | 12 | DFICA is added to YDFICA and the result is placed in the field HOLD. |
| 07 | GROSS is added to SAVE if Indicator 22 is off. | 13 | The contents of HOLD are compared with the literal 144.00, and if HOLD is less than, or equal to, 144.00, Indicator 21 is turned on. |
| 08 | The program branches to the label FICA if Indicator 22 is off. | 14 | If Indicator 21 is on, the program branches to the label ADFICA. |
| 09 | Additional calculations. | 15 | YDFICA is subtracted from the literal 144.00 and, the result is placed in DFICA. |
| 10 | The operation code TAG provides the label FICA. The RPG program branches to this label. | 16 | The operation TAG provides the label ADFICA to which the program can branch (either from the specification on line 14 or sequentially from the specification on line 15). |
| 11 | GROSS is multiplied by the literal .03 and the result is placed in | 17 | DFICA is added to YDFICA and the result is stored in YDFICA. |
| | | 18 | Additional calculations. |
| | | 19 | The operation code TAG provides the label WHTAX to which the RPG program can branch. |

The file name may be alphameric, but it must not contain special characters or blanks.

When writing the output specifications, the file name need only be entered once. Enter it on the first line to define the file as shown in Figure 77.

TYPE H/D/T (15)

The entry in this column identifies the type of record being specified. The following three entries are used for this specification:

- H -- Heading Record
- D -- Detail Record
- T -- Total Record

NOTE: All heading records for a file must be entered first, followed by all detail records, and then by all total records for the same file (Figure 77).

Heading Records. These records usually contain constant information. However,

they may also contain information from input records, including the record present at the time the output record is produced.

Detail Records. These records have a direct relationship to the input record. Most data in a detail record comes from the input record or from calculations performed at detail time.

Total Records. Operations upon fields from the input record are preceded by the test for control-field changes, the performance of total-time calculations, and the formation of total records. Thus, data from an input record that causes a control-field change cannot contribute data to total records that result from that control change. But heading and detail records can contain data from the input record.

STACKER SELECT (16)

If punched output occurs in the object program, a stacker number is entered in this column. The cards fall in the stacker that is indicated by this column. The stacker pockets and their acceptable codes are listed in Figure 78.

SPACE (17-18)

This specification (and the next specification Skip, columns 19-22) is used to provide the proper spacing of printed reports.

NOTE: If the record is to be printed, at least one entry is required in columns 17-22.

Figure 77. Specifying Heading, Detail, and Total Lines

| Output Unit | Stacker Number | Stacker Select Code |
|----------------------|-----------------|---------------------|
| 1442 Card Read-Punch | 1 2 | 1 or Blank 2 |
| 2520 Card Read-Punch | 1 2 | 1 or Blank 2 |
| 2540 Card Read-Punch | P1 P2 RP3 | 1 2 3 |

Figure 78. Summary of Stacker Select Specifications (Output)

Space Before (Column 17)

Enter in this column the number of lines to be spaced before the line is printed. Specify zero, one, two, or three spaces before printing by placing the entry 0, 1, 2, or 3 in column 17. If this column is left blank, no spacing before printing will be provided.

Space After (Column 18)

Specify zero, one, two, or three spaces after printing by placing the entry 0, 1, 2, or 3 in column 18. A blank in this column will provide single spacing after printing a line.

SKIP (19-22)

This specification provides for the proper spacing of reports. It is directly related to the function of the printer carriage-control tape.

Skip Before (Columns 19-20)

The entries 01-12 cause the printer carriage to skip to channels 1 through 12 of the carriage tape before the line is printed. In Figure 77, before the heading line is written, the form skips to channel 1.

Skip After (Columns 21-22)

The entries 01-12 cause the printer carriage to skip to channels 1 through 12 of the carriage tape after the line is printed.

NOTE: The order in which spacing and skipping is performed is as follows:

Skip Before
Space Before
Skip After
Space After

Overflow Indicator

Carriage overflow or the line number associated with channel 12 in the Line Counter Specifications sheet cause the setting of the indicator as specified by the user in the File Description Specifications (columns 33-34).

Channel 12 is sensed as the corresponding line is printed. The overflow indicator is turned on after the next line is printed. Thus, one extra line is always printed after the overflow is detected and before the overflow functions can occur. In planning a printer operation, the carriage tape

overflow punch must coincide with the next-to-last line to be printed on the form.

Using the Line Counter specifications causes the overflow indicator to be turned on when the line counter reaches the specified line number.

The overflow indicator remains on for one complete processing cycle and is off after the heading and detail lines of the next record are printed. The overflow indicator can be used to control calculation specifications because it is on during calculations.

A test for the overflow status is made by the object module immediately before each line of the report is printed (but after any Space Before specifications are executed). Two conditions can occur at this time:

1. If the overflow indicator is on before a detail line is printed, the detail line, and any other detail lines whose output indicators are on, are printed. Any Space After specifications are executed, the next record is read, and a test is made to determine if a control break has occurred. If one has occurred, all total lines (caused by the control break) are printed, and then any overflow printing specified is performed.
2. If the overflow indicator is on before a total line is printed, all total lines, whose output indicators are on, are printed. This is followed by any specified overflow printing.

Automatic Skipping

If an overflow indicator is not specified as one of the first three indicators (columns 23-31 of output specifications) for any line of a print file, then the RPG compiler provides the object module with automatic skipping from channel 12 to channel 01 on that file.

Printing Lines Conditioned by Overflow

If an output line is conditioned by either an overflow or a control level indicator, the line will print whenever either of these indicators is on. If the overflow indicator and control level indicator occur at the same time during processing, the line prints twice.

This situation may be avoided by using the coding shown in Figure 79. The detail line 010 in Figure 79 specifies that if a level-1 control break does not occur on this cycle but has occurred on the previous cycle, indicator 25 is turned off. The total line 020 specifies that if a level-1 control break has occurred, indicator 25 is turned



INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
 IBM System/360

Form X24-3351
 Printed in U.S.A.

Date _____

Program _____
 Programmer _____

| | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | |
| | Punch | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Control Level (U, O, L, E) | Indicators | | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Decimal Positions Shift After (P) | Resulting Indicators | | | Comments |
|------|-----------|----------------------------|------------|-----|------|-------|----------|-----------|----------|--------------|--------------|-----------------------------------|----------------------|-------|--|----------|
| | | | And | And | Plus | Minus | | | | | | | Zero or Blank | | | |
| | | | Not | Not | Not | | | | | | | High | Low | Equal | | |
| | | | 1 | 2 | 3 | | | | | | | > 2 | < 2 | 1 = 2 | | |
| 01 | Ø | C | N | L | 1 | 25 | | | SET OF | | | | 25 | | | |
| 02 | Ø | C | L | | | | | | SET ON | | | | 25 | | | |
| 03 | | C | | | | | | | | | | | | | | |
| 04 | | C | | | | | | | | | | | | | | |



INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
 IBM System/360

Form X24-3352
 Printed in U.S.A.

Date _____

Program _____
 Programmer _____

| | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | |
| | Punch | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Filename | Space | Skip | Output Indicators | | | | Field Name | Zero Suppress (Z) Blank After (B) | End Position in Output Record | Packed Field (P) | Constant or Edit Word | Sterling Sign Position |
|------|-----------|----------|--------|-------|-------------------|-------|-----|-----|------------|-----------------------------------|-------------------------------|------------------|-----------------------|------------------------|
| | | | | | And | And | Not | Not | | | | | | |
| | | | Before | After | Before | After | Not | Not | | | | | | |
| | | | 1 | 2 | 1 | 2 | 1 | 2 | | | | | | |
| 01 | Ø | OUTPUT | H | Ø | 1 | | | | OFN | 25 | | | | |
| 02 | Ø | | OR | | | | | | L1 | | | | | |
| 03 | Ø | | | | | | | | | | 100 | 'ACCOUNT' | | |
| 04 | | | | | | | | | | | | | | |

Figure 79. Specifying Output Indicators for Overflow

on. Indicator 25 remains on for an extra total cycle because:

1. If overflow has occurred at detail time, it will be sensed at total time.
2. The overflow routine is executed after the control level-1 indicator (L1) has been turned off.

The output line is coded as shown in the lower half of Figure 79. By making the condition mutually exclusive, the line is not printed twice.

The order in which the object module prints lines conditioned by overflow follows:

1. When overflow occurs during the printing of a heading or detail line, the object module does not print lines conditioned by overflow. The program does, however, take note that the overflow condition has occurred.

2. Total lines not conditioned by overflow are printed.
3. The object module then tests to see if overflow has occurred.
4. Total lines conditioned by overflow are then printed.
5. Heading and detail lines conditioned by overflow are then printed.

NOTE: If an overflow condition has been specified, automatic skipping is not performed.

Multiple Printers

With this program it is possible to use a maximum of eight output printer files for each RPG object module. When producing records for more than one printer, the user must specify the printer on which the overflow condition occurs by assigning a unique overflow indicator for each printer.

In columns 33-34 of the File Description sheet, one of eight available overflow indicators is entered. The following eight indicator codes are used:

OA, OB, OC, OD, OE, OF, OG, and OV.

OUTPUT INDICATORS (23-31)

This specification may be used for either file identification or field description. Entries in this column may specify a maximum of three indicators. These indicators control:

1. When the line is to become output, or
2. When a particular field is to be written.

The following information applies only to its use in the file identification line.

If more than one indicator is specified on one line, all indicators are considered in an AND relationship. That is, all indicator conditions specified must be satisfied before the output condition can be executed.

If the object module requires that more than three indicators be in an AND relationship, the letters AND are entered in columns 14-16 of the following line, and the additional indicators are specified on that line.

If the output condition is executed in an OR relationship (one or the other of two sets of conditions) the letters OR are entered in columns 14-15 of the following specification line, and the OR indicators are specified on that line.

Additional specification lines can specify as many output indicators in either an AND or OR relationship as required by the object module. Each additional line must begin with AND or OR in column 14.

When an OR line is specified on a print file, the printer control functions (columns 17-22) may all be left blank, in which case those of the preceding line will be implemented. They may, however, differ from the preceding line if required. This implementation does not occur for Stacker Select entry since a blank is a valid stacker select code.

If a line is to be conditioned as an overflow line, the overflow indicator must not appear in a specification line having the letters AND in columns 14 through 16.

Examples of Output Indicators

The entries for this specification can be arranged into the categories listed below.

1. A resulting indicator code specifies the particular indicator code on which the output operation is to be performed. The operation cannot be performed on other record types.
2. A field indicator code controls the output operation by the status of an input field.
3. A resulting indicator code controls the output operation by conditions that have occurred during calculations.
4. A control level indicator (L1 through L9 and LR) causes the output operation to be performed only at the control level specified.
5. The MR indicator code causes the output operation to be performed only if

there is a matching record in a second input file.

6. The halt indicators H0 through H9 are normally used to suppress an output operation when an error has been detected in the input data or during a calculation.
7. Overflow indicators cause the output operation to take place only if a page overflow has occurred. (Overflow line.)
8. The LP (first page) indicator enables overflow headings to be printed on the first page of a report. (This indicator is turned on at the beginning of processing before any input records have been read.)

In addition, this specification may contain a combination of some of the eight categories listed. Also, the operation may be controlled by the fact that an indicator must not be on. This is accomplished by placing an N in columns 23, 26, and 29.

Figure 80 shows six examples of output indicators when they are used as part of the file identification specifications. The numbers to the right refer to the following discussion:

| Line | | Form Type | Filename | Type (R/D/T) | Space | Skip | Output Indicators | | | | | | Field Name | Zero Suppress |
|------|---|-----------|-----------|--------------|----------------|-------|-------------------|----------|-----|----|----|----|------------|---------------|
| | | | | | Stacker Select | | And | And | And | | | | | |
| | | | | | Before | After | 23 | 26 | 29 | 30 | 31 | 32 | 33 | 34 |
| 0 1 | 0 | | DAILYRPTH | | | 02 | | | | | | | | |
| 0 2 | 0 | | | OR | | | | 1P | | | | | | ① |
| 0 3 | 0 | | | | | | | | | | | | | |
| 0 4 | 0 | | | D | 1 | | | 14N26N28 | | | | | | ② |
| 0 5 | 0 | | | AND | | | | N30 | | | | | | |
| 0 6 | 0 | | | | | | | | | | | | | |
| 0 7 | 0 | | | D | 1 | | | 40 | | | | | | ③ |
| 0 8 | 0 | | | OR | | | | 46 | | | | | | |
| 0 9 | 0 | | | | | | | | | | | | | |
| 1 0 | 0 | | | T | | 02 | | L2NMRNH2 | | | | | | ④ |
| 1 1 | 0 | | | | | | | | | | | | | |
| 1 2 | 0 | | SUMREC | T | 1 | | | L2 MR | | | | | | ⑤ |
| 1 3 | 0 | | | | | | | | | | | | | |
| 1 4 | 0 | | SUMCARD | T2 | | | | L2 | | | | | | ⑥ |

Figure 80. Specifying Output Indicators

1. The control carriage is skipped to channel 02 before printing, and the heading line is printed only when an overflow condition occurs or when the 1P (first page) indicator is on.
2. The detail line is printed only if Indicator 14 is on, and Indicators 26, 28, and 30 are off. Indicators 26, 28, and 30 could be field indicators from the input specifications, or they could be resulting indicators from the calculation specifications.
3. The detail line is printed if Indicator 40 or 46 is on.
4. The total line is printed and the control carriage is skipped to channel 2 before printing only if the Level-2 indicator is on, the MR indicator is off, and H2 is off. The specification NH2 prevents the object module from printing a line if an error condition has occurred.
The program does not stop until after all processing for the record has been completed. This feature enables the programmer to prevent the printing of erroneous data.
5. The summary record is printed at the Level-2 control time, but the MR (matching record) indicator must also be on.
6. A summary card punched at the Level-2 control time is selected into stacker 2 of an IBM 1442 Card-Read Punch.

This concludes the description of the file identification specifications. The remaining descriptions of the output form are concerned with the field-description section of the output specifications.

FIELD DESCRIPTION

These entries include specifications about:

1. The control of the individual fields of a record, and
2. The output format of individual fields of a record. The fields of the record are written on the lines below their corresponding file entries. Each field is described on a separate line, and no entries are permitted in columns 7-22 of a field-description line.

OUTPUT INDICATORS (23-31)

The same types of indicators used for file identification can be used for field

description. The maximum number of indicators that can be considered in an AND relationship is three; all must be specified on one field-description line.

Figure 81 shows four sets of indicators used as output indicators for field-description lines. The numbers to the right of the figure correspond to the following list:

1. Four fields are printed from a detail record identified by Indicator 44: invoice, amount, customer, and salesman. The entry 11 causes the field, SALESM, to be printed only for the first detail record of each control group. (Remember that a control level indicator remains on during the following detail calculation and print cycle.) The salesman field is "group indicated".
2. The second example illustrates how to prevent the printing of just one field of a record. The field AMOUNT is printed only if Indicator 16 is off. This indicator is used to determine if the calculated field AMOUNT is zero.
3. This example selectively prints the field headings for an invoice form. This specification prints all the heading information on the first form, but if the information for one customer order continues on two or more forms, only the customer and invoice fields on succeeding forms are printed.
Printing of the entire line is controlled by Indicators 04 or OF. In the field description specifications, the OF indicator is also used to prevent printing of fields order number, date, and salesman when an overflow condition occurs.
4. The last example illustrates how a field can be controlled for printing by an AND relationship and an OR relationship.

The field DIVSON is controlled by three AND indicators: 16, NH2, and NL3.

The field AMOUNT is controlled by two OR conditions. In the field description line, the OR relationship is used by writing the field name twice and specifying each appropriate OR indicator.

The letters OR cannot be specified in columns 14-15 of a field description.

FIELD NAME (COLUMNS 32-37)

This specification identifies each field of the record to be written. The fields may

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
 IBM System 360

Form X24-3352-1
 Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | |
| | Punch | | | | | | |

Page

| | |
|---|---|
| 1 | 2 |
|---|---|

Program Identification

| | | | | | |
|----|----|----|----|----|----|
| 75 | 76 | 77 | 78 | 79 | 80 |
|----|----|----|----|----|----|

| Line | Form Type | Filename | Type (M/D/T) | Space | | Skip | | Output Indicators | | | Field Name | Zero Suppress (Z) Blank After (B) | End Position in Output Record | Packed Field (P) | Constant or Edit Word | Sterling Sign Position |
|------|-----------|----------|--------------|--------|-------|--------|-------|-------------------|-----|--------|------------|--------------------------------------|----------------------------------|------------------|-----------------------|------------------------|
| | | | | Before | After | Before | After | And | And | And | | | | | | |
| | | | | Not | Not | Not | Not | Not | Not | Not | | | | | | |
| 0 1 | 0 | PRINT | D | 1 | | | | 44 | | | | | | | | |
| 0 2 | 0 | | | | | | | | | INVOIC | | | | | | |
| 0 3 | 0 | | | | | | | | | AMOUNT | | | | | | |
| 0 4 | 0 | | | | | | | | | CUSTR | | | ① | | | |
| 0 5 | 0 | | | | | | | L1 | | SALESM | | | | | | |
| 0 6 | 0 | | | | | | | | | | | | | | | |
| 0 7 | 0 | PRINT | D | 1 | | | | 24 | | | | | | | | |
| 0 8 | 0 | | | | | | | | | NAME | | | | | | |
| 0 9 | 0 | | | | | | | | | DEPT | | | ② | | | |
| 1 0 | 0 | | | | | | | N16 | | AMOUNT | | | | | | |
| 1 1 | 0 | | | | | | | | | | | | | | | |
| 1 2 | 0 | PRINT | D | 1 | 03 | 04 | | | | | | | | | | |
| 1 3 | 0 | | OR | | | | | OF | | | | | | | | |
| 1 4 | 0 | | | | | | | NOF | | ORDER | | | | | | |
| 1 5 | 0 | | | | | | | NOF | | DATE | | | | | | |
| 1 6 | 0 | | | | | | | NOF | | SALESM | | | ③ | | | |
| 1 7 | 0 | | | | | | | | | CUST | | | | | | |
| 1 8 | 0 | | | | | | | | | INVOIC | | | | | | |

Card Electro Number _____

| Line | Form Type | Filename | Type (M/D/T) | Space | | Skip | | Output Indicators | | | Field Name | Zero Suppress (Z) Blank After (B) | End Position in Output Record | Packed Field (P) | Constant or Edit Word | Sterling Sign Position |
|------|-----------|----------|--------------|--------|-------|--------|-------|-------------------|--------|--------|------------|--------------------------------------|----------------------------------|------------------|-----------------------|------------------------|
| | | | | Before | After | Before | After | And | And | And | | | | | | |
| | | | | Not | Not | Not | Not | Not | Not | Not | | | | | | |
| 0 1 | 0 | TOTAL | 7 | 1 | | | | 12 | MR | | | | | | | |
| 0 2 | 0 | | | | | | | 16 | NH2NL3 | DIVSON | | | | | | |
| 0 3 | 0 | | | | | | | 26 | | AMOUNT | | | ④ | | | |
| 0 4 | 0 | | | | | | | 28 | | AMOUNT | | | | | | |
| 0 5 | 0 | | | | | | | | | | | | | | | |

Figure 81. Specifying Field Names

be listed on the form in any sequence. The order in which they appear in the output record is determined by the entry in columns 40-43.

Enter in columns 32-37 the name of the field that is defined on the line. The field name must have been previously defined on either the input or calculation sheet. (For an exception to this, see Page Numbering.) If a constant is to be written, it is specified in Constant or Edit Word (columns 45-70), and Field Name is left blank.

Examples of field names are given in Figure 81.

ZERO SUPPRESS (38)

An entry of Z in this specification causes zero-suppression of the field in the output record, which must be a numeric field. Zero suppression means that zeros to the left of the first (high-order) significant digit in the field are replaced by blanks in the output record.

This specification performs an additional function: zone bits in the unit position of a field are removed during zero suppression. This provides a means of "zone elimination" for amount fields. Normally, this specification is used for printing numeric fields, but not for punching them, because it is usually necessary to punch the sign codes and leading zeros.

If an edit control word is specified for the field, this specification must be blank.

BLANK AFTER (39)

An entry of B in this specification causes the output field to be reset to blanks or zeros after the field is placed in the output record. Alphameric fields are set to blanks; numeric fields are set to zeros.

This specification has an additional feature: if the output field being reset by the Blank After specification is also being tested for zero or blank on the input specifications or being tested for zero on the calculation specifications, the corresponding indicator is turned on after the output field is reset. However, associated plus or minus indicators are not turned off.

If a given field is specified for output at more than one position in the same record, the blank-after entry is made only on the last specification where the field is used.

NOTE: A Blank After specification also affects any constant in storage. Since each constant is stored only once for every RPG program — no matter how often it is used — a constant affected by a Blank After specification is not available for use in subsequent operations.

END POSITION IN OUTPUT RECORD (40-43)

This specification indicates the location of the field in the output record. In columns 40-43, enter only the position in the output record where the rightmost (low-order) character of the field is to be located.

Assume that a ten-position amount field is to be printed in print positions 21 through 30. The entry in columns 42 and 43 would be 30. Columns 40 through 41 are left blank, because the leading zeros may be omitted.

If an amount field is being edited, the program considers the entire edit word as the "field" to be placed in the output record. For example, in the previous example assume the letters CR are to be printed at the end of minus amount fields. The R would be printed in position 30, the C in position 29, and the 10-position amount field would then be printed in positions 19 through 28. Thus the amount field is always printed in positions 19-28, and the credit symbol (if applicable) is printed in positions 29 and 30.

Figure 82 illustrates various field description specifications. It shows the specifications for printing a detail line. The three quantity fields ORDQTY, SCRAPQ, and RECPTQ are zero-suppressed. The quantity fields SCRAPQ and RECPTQ are also reset to zeros by the Blank After specification.

Figure 83 illustrates an example of field selection specification. These entries punch a summary card at control level-1. In this example, a new balance (NEWBAL) and an old balance (OLDBAL) are used in the weekly reports; at the end of the month only the new balance is to be summarized for the following month's report.

Both fields are specified for the same punching positions in the output record. The actual field that is punched, however, is controlled by the setting of Indicator 26. The number of fields that can be specified for field selection is not limited.

Page Numbering for Multiple Printers

The individual printer files can initialize page numbering. Eight special PAGE entries (PAGE, PAGE1, PAGE2, PAGE3, PAGE4, PAGE5, PAGE6, and PAGE7) can be initialized to any count in the same manner as described for PAGE.

CONSTANT OR EDIT WORD (45-70)

This specification provides constants in the output records or permits editing of numeric fields. Constants and edit words must be left-justified.

Constants

This entry provides a convenient method of placing into the object module alphameric data (literals) that does not change from one processing of the program to the next. A literal is the actual data to be used in the output record rather than a name representing the location of data.

A literal of up to twenty-four alphameric characters may be placed in columns 45-70. The literal must begin in column 45, and it must be enclosed by a set of apostrophe symbols even if it contains only numeric characters. The literal beginning in column 45 will be placed on the output record as defined in End Position in Output Record, (columns 40-43).

Rules for Forming Alphameric Literals in an Output Record

1. Any character in the character set may be used in an alphameric literal. Blanks are treated as valid characters.
2. Alphameric literals must be enclosed in a set of apostrophe symbols. An apostrophe symbol may be contained within a literal by entering two consecutive apostrophe symbols within the literal. For example, the literal o'clock would be entered as 'o'clock' in columns 45-54 of the Output-Format Specifications sheet.
3. Field Name (columns 32-37) must be left blank when an alphameric literal is defined on the line.

Edit Words

An edit word provides for the punctuation of amount fields, including the printing of dollar signs, commas, periods, and sign status. Column 38 (Zero Suppression) and column 44 (Packed) must be left blank. An edit operation is shown in Figure 86.

When an amount field is to be edited, its edit word is placed in columns 45-70 of the line which specifies the field. An edit word consists of two parts (the body and the status) as shown in Figure 87.

The body of the edit word is the portion beginning with the leftmost character, and continuing to the right of the character that governs the transfer of the rightmost position of the data field.

The status of the edit word is the portion continuing to the right from the body including all characters preceding the closing apostrophe symbol. The CR (credit) or - (minus), if present, may appear anywhere within the status.

A character in the edit word that is replaced by a numeric character from the data field is referred to as a digit position. A blank in the edit word is a digit position as are the characters 0, *, and \$ under certain conditions.

Rules for Forming an Edit Word

1. An edit word must be enclosed in a set of apostrophe symbols.

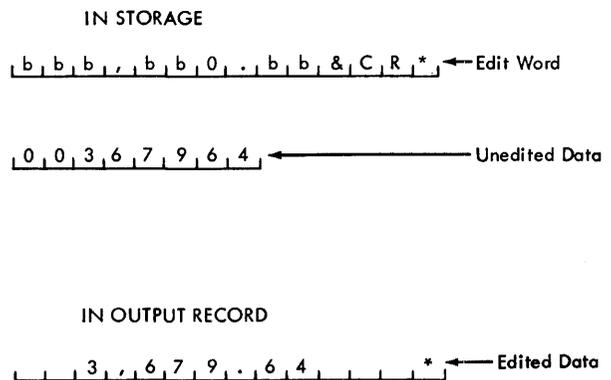


Figure 86. Edit Operation

2. A blank in the edit word is replaced with the digit from the corresponding position of the data field specified in Field Name.
3. An ampersand causes a space in the edited field. There is no way to obtain an ampersand in an edited field.
4. A zero is used to stop zero-suppression. It is put in the rightmost position where zero suppression is to take place. It is replaced with the character from the corresponding position of the data field, unless that character is a zero. At least one leading zero is suppressed.
5. An asterisk in the body of the control word is used for asterisk protection and zero suppression. It is put in the rightmost position where zero suppression is to take place. It is replaced with the character from the corresponding position of the data field unless that character is a zero and there is no significant digit to its left. Each zero that is suppressed is replaced by an asterisk.
6. A dollar sign in the body of the edit word written immediately to the left of the zero-suppression code (0) causes the insertion of a dollar sign in the position to the left of the first significant digit. This is called the floating dollar sign. If it is necessary for the dollar sign to appear when all digits in the data are significant, the edit word must start with an ampersand to allow a space in which it can print.

A dollar sign that is entered immediately after the initial apostrophe symbol will be fixed. That is, it will be printed in the same location each time. This is called the fixed dollar sign.

NOTE: In the edit word '\$0bb' the dollar sign is considered to be fixed and not floating.

7. The decimal and commas are printed in the edited output field in the same relative positions in which they were written in the edit word unless they are to the left of significant digits. In that case, they are blanked out or replaced by an asterisk. Any characters other than the fixed dollar sign which precede the first digit position will always be suppressed.
8. All other characters used in the body of the edit word are printed if they are to the right of significant digits in the data field. If they are to the left of high-order significant digits in the data word, they are blanked out, or if asterisk protection is used, they are replaced by an asterisk.
9. The letters CR or the minus symbol in the status portion of the edit word are undisturbed if the sign in the data field is minus. If the sign is plus, they are blanked out.
10. Asterisks to the right of the CR or minus symbol are undisturbed. They are normally used to indicate a specific class of total. An error results unless a zero (for zero-suppression) or an asterisk (for asterisk protection) are present in the body of the edit word.
11. If there are more digit positions in the edit word than there are digits in the field to be edited, leading zeros will be added to the field before editing.
12. An edit word can contain a maximum of 15 digit positions. An exception occurs for a sterling field where the maximum is 16 digit positions.

Figure 88 illustrates the use of constants and edit words. The numbers to the left correspond to the following list:

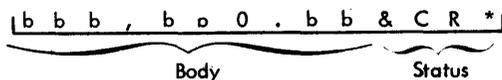


Figure 87. Two Parts of an Edit Word: Body and Status

1. The constant 26.75 is in the output record ending in position 96. The Field Name specification must be blank.
2. The constant DEPARTMENT TOTAL is contained in the output record ending in position 96. The Field Name specification must be blank.

| Field Name | Zero Suppress (Z) | Blank After (B) | End Position in Output Record | Packed Field (P) | Constant or Edit Word | VALUE IN | CONTAINED | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------|-------------------|-----------------|-------------------------------|------------------|-----------------------|------------|----------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|
| | | | | | | DATA FIELD | IN OUTPUT RECORD AS: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 7 |
| ① | | | 96 | | '26.75' | | 26.75 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ② | | | 96 | | 'DEPARTMENT TOTAL' | | DEPARTMENT TOTAL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ③ | AMOUNT | | 96 | | ' , 0. &CR*' | 000030 46- | 30.46 CR* | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ④ | AMOUNT | | 96 | | ' , \$0. &CR*' | 00030 46+ | \$30.46 * | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ⑤ | AMOUNT | | 96 | | ' , *. CR**' | 000030 46- | *****30.46CR** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 88. Using Constants and Edit Words

- This example illustrates zero-suppression to the left of significant digits. The letters CR are written because the amount field might be minus.
- In this example, the floating dollar-sign protection enters the \$ to the left of the first significant digit.
- Asterisk protection enters as many asterisks to the left of the first significant digit as required to fill out the number of positions specified in the edit word.

When using a printer with the UCS feature, data checks may occur during RPG compilation of a program that uses edit words. To suppress these unit checks, one of the following options must be used:

- Suppress data checks at compile time by omitting the SYSPRINT DD card parameter DCB=(,OPTCD=U). (Refer to IBM System/360 Operating System, Supervisor and Data Management Macro-Instructions.) Cataloged procedures do not

- include this parameter and can therefore be used to suppress data checks on a printer with UCS.
- Use the utility program, IEHUCSLD, (IBM System/360 Operating System: Utilities) to load X'20' and X'21' into the UCS buffer.

STERLING SIGN POSITION (71-74)

Enter in these columns the position in the record that will contain the sign of the sterling field. Leading zeros may be omitted. Enter an S in column 74 if the sign is in the normal position. For Print files the normal position must be used. If the Sterling specification is not required, leave this column blank. Additional information on Sterling is in Appendix D. Sterling Routines for the Report Program Generator.

The Line Counter Specifications sheet provides the important facility to store reports, which will ultimately be printed, on any intermediate device such as tape or disk. It also provides the convenience of an "internal" carriage control tape when using an actual printing device.

This sheet is necessary because any output lines for the printed report that depend upon the sensing of a channel-12 punch in the carriage control tape will not be produced for tape or a direct access storage device. Thus, the Line Counter Specifications sheet is used to relate the line of the printed page of the report to its corresponding punch in the carriage control tape.

A standard IBM System/360 write/control or independent carriage control character is added as the first byte of each record that is put out on the intermediate device. For example, a 132 character record becomes 133 characters long. All space before and skip before controls are put out as independent carriage control characters in a separate record that does not print. Space after and skip after controls are put out as modifier bits in conjunction with the write command for the line. A line that has space or skip before control, produces two records on the intermediate device, while a line with skip or space after control produces only one record. An auxiliary program must be used to retrieve these records for printing. This program can be written in RPG or another programming language.

NOTE: When automatic skipping is desired (overflow indicators are not specified) and the report is to be stored on some intermediate device, the Line Counter Specifications sheet must be used.

HOW TO USE LINE COUNTER SPECIFICATIONS

In Figure 89, three lines of the report are conditioned by the carriage control tape. Channel 1 is used to control printing of Monthly Sales Report.

Tape channel 6 is used to control amount, and channel 12 is used to condition total sales. On the Line Counter Specifications sheet (Figure 90) the name of the file is entered in columns 7-14. The entries, in this example, in columns 15-29 relate directly to the printer spacing chart. The numbers in the following list correspond to this discussion:

1. The first line controlled by the carriage control tape is line 6. In columns 15-17 of the Line Counter sheet, 006 is entered. The corresponding channel punch is channel 1. In columns 18-19, 01 is entered.
2. Line 13 of the report (amount) is controlled by a punch in channel 6. Columns 20-22 contain 013, and Columns 23-24 contain 06.
3. Line 17 (total sales) is controlled by the punch in channel 12 of the carriage control tape. Columns 25-27 of the Line Counter Specification sheet contain 017, and columns 28-29 contain 12.

Entries on the Output-Format Specifications are not changed. On the File Description sheet, the line that is used to define the output file must contain a V in File Format (column 19) and an L in the Extension Code Field (column 39). The L indicates that a line counter specification is required for the output file.

If the Line Counter Specifications sheet is used, an entry must be made for channel 1 and channel 12.

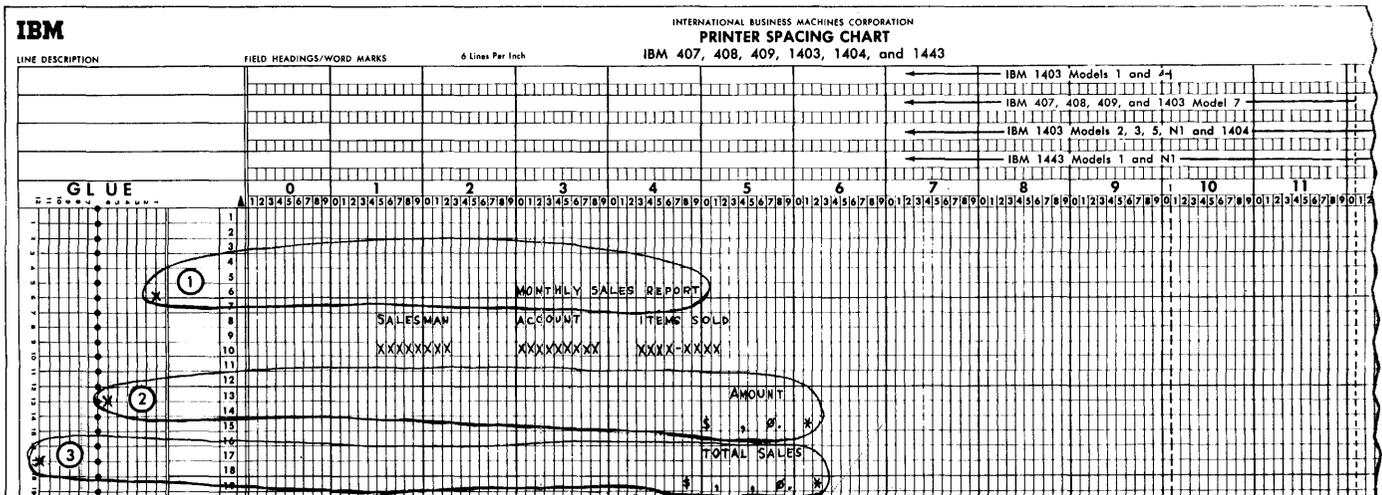


Figure 89. Using the Carriage Control Tape

FILE DESCRIPTION SPECIFICATIONS SHEET

This sheet is used to provide information to RPG about:

1. The input files, defined on the Input Specifications sheet, from which the object module will obtain data records.
2. The input files used by the object program, such as record-address files, table files, and chaining files.
3. The output files, defined on the Output-Format Specifications sheet on which the object module will write data records. Each file used by the object program must be defined.

Each file used by the object program must be defined. Each line of the File Description sheet is used to define one file.

This form also identifies each file used in the program with the input or output unit with which it is associated. Each line of the form is used to specify one file.

Maximum Number of Files Available

The maximum number of files that can be used in the program is 10. The list below defines the maximum number of files for the various types of files that can be used. Any combination of these, up to 10, is permitted.

| <u>Type of File</u> | <u>Maximum Number</u> |
|---------------------|-----------------------|
| Input | - |
| Primary | 1* |
| Secondary | 8 |
| RAF | 1 |
| Chained | 9 |
| Table | 8 |
| Output | 9** |
| Update | - |
| Primary | 1 |
| Secondary | 8 |
| Chained | 9 |
| Combined | - |
| Primary | 1 |
| Secondary | 1 |

*Each program must have one (and only one) primary file.

**Up to a maximum of eight printers can be used in a program.

FILENAME (7-14)

Each file used in the program is identified by writing the name of the file in columns

7-14. The file name must be left-justified (that is, it must start in column 7) and it must begin with an alphabetic character. The remaining characters of the name may be alphanumeric, but must not contain special characters or embedded blanks. The file name may be eight characters or fewer. (Embedded blanks are blank positions falling between other characters of the name.) The file name entered in these columns must also have been entered on the Input Specifications sheet (for input data files), on the Output-Format Specifications sheet (for output data files) or on the File Extension Sheet (for Table, RAF, or chaining files).

FILE TYPE (15)

An entry in this column specifies the type of file defined on the line of the sheet. The following four entries are allowed in this column:

I Input File

Identifies the file as an input file (it may be a record-address file, table file, or a file containing input data records).

O Output File

Identifies the file as an output file (it may be an updated table file to be written out).

U Update File

Identifies the file as an update file (Direct Access Storage Device only). An update file is both an input and an output file.

The file is an update file if the object program alters the data in one or more fields of each record contained in the file. The overall length of the record cannot be changed even in a variable length file.

A chained file may be updated at detail time or at total time. All other DASD files can be updated at detail time only.

C Combined File

Identifies the file as a combined file (card file only). A combined file is a file containing cards read into the system

and cards used for punching. It must reside on an IBM 1442 Card Read-Punch.

Reading and then punching into the same file is accomplished in two different ways.

1. Punching into the same card that is read.
2. Punching into a blank trailer card in the same file.

Figure 91 illustrates two examples of File Name and File Type specifications.

1. Detail cards are read into the system in one file, summary cards are punched into a separate file (which, in this example, might be the punch feed of an IBM 2540 and might contain blank cards only), and the printed report is in a third file.
2. There are two input files in this example. The second file (INPUTSEC) also contains cards that will be used for punching output data; therefore it is specified with file-type C.

FILE DESIGNATION (16)

This specification is used to designate the type of input file being defined on this line of the File Description sheet. The five entries permitted in this column are listed here. Detailed explanations of chained files, table files, record-address files, primary files, and secondary files may be found in the sections Using Tables

in the Object Program and Processing Multiple Input Files at the back of this publication.

If the file is an output file, leave this column blank. An entry must be made if the file is an Input, Update or Combined file. Enter in column 16 the following codes:

| Entry | Explanation |
|-------|---|
| P | The file defined is a primary file. Only one primary file may be defined. |
| S | The file defined is a secondary file. |
| R | The file defined is a record-address file which relates to a direct access storage file. Only one RAF may be defined. |
| C | The file defined is a chained file. |
| T | The file defined is a table file. |

END OF FILE (17)

Enter an E in column 17 if the input file is a sequential file and it is to be checked for an end-of-file condition.

For one input file, the E entry will cause the LR (Last Record) indicator to turn on when the last record of the file has been processed. For multiple input files that utilize chaining or matching records, the end-of-job condition (LR) will occur when all the input files (defined on this sheet with an E in column 17) have reached the end-of-file.

If this column is left blank for all input files, the end-of-job condition (LR) occurs when all input files have been processed.

NOTE: An E should not be entered in Column 17 if there are multiple input files to be processed in sequence (no chaining or matching records specified), if the file is processed randomly, or if the file is either an output or table file.

SEQUENCE (18)

This entry is normally made if there is more than one input file and the matching-fields specification (columns 61-62 of the Input Specifications Sheet) is used. An entry in this column indicates whether the matching fields are in ascending or descending sequence.

Enter an A in column 18 if the matching fields are in ascending order, or a D if the matching fields are in descending order.

NOTE: If this column is left blank when matching fields are used, ascending order is assumed.

The image shows a portion of an IBM File Description Sheet. At the top left is the IBM logo, and at the top right is 'REPORT P'. Below these are fields for Date, Program, and Programmer. The main table has columns for Line, Form Type, Filename, File Type, File Designation, End of File, Sequence, and File Format (Block Length and Record Length). Two rows are highlighted with circled numbers 1 and 2.

| Line | Form Type | Filename | File Type | File Designation | End of File | Sequence | File Format | |
|------|-----------|----------|-----------|------------------|-------------|----------|--------------|---------------|
| | | | | | | | Block Length | Record Length |
| 0 1 | F | INPUT | IFE | | | | | |
| 0 2 | F | SUMCARD | O | | | | | |
| 0 3 | F | PRINT | O | | | | | |
| 0 4 | F | INPUTPRI | IFEA | | | | | |
| 0 5 | F | INPUTSEC | CSEA | | | | | |
| 0 6 | F | PRINT | O | | | | | |
| 0 7 | F | | | | | | | |

Figure 91. Specifying File Name and File Type

Sequence-Checking of Input Files

If there is only one input file or a chaining file in the program, this specification may be used to sequence-check fields of the file to ensure that the file is in sequence. By entering the codes M1, M2, or M3 in columns 61-62 of the Input Specifications sheet, sequence-checking with respect to these fields occurs. In this specification (column 18), either A or D must be entered to specify whether the file is ascending or descending.

The Halt Indicator, H0, is turned on when a record of an input file is found to be out of sequence. Unless the H0 indicator is turned off by a SETOF operation in the calculation specifications, the program will terminate before the next input record is read.

FILE FORMAT (19)

This column is used to indicate the format of the input or the output records. Enter an F in this column if the records are fixed-length. Enter a V in this column if the records are variable in length.

NOTE: Enter a V in this column if the output file is used in conjunction with the Line Counter Specifications Sheet.

BLOCK LENGTH (COLUMNS 20-23)

This specification is used to indicate the block length of the input or output records. It is used in conjunction with the next specification, Record Length.

The RPG program provides four techniques for handling records:

1. Fixed-length. All records have the same number of bytes of data.
2. Variable-length. Each record can have a different number of bytes of data.
3. Unblocked. Only one logical record in each physical record.
4. Blocked. Two or more logical records in one physical record. Blocked records are not supported by the Direct Access Method (DAM).

Fixed-length, Unblocked (Figure 92). Each logical record is the same length as the physical record.

Fixed-length, Blocked (Figure 93). Blocked records are usually considered to be two or more logical records within one physical record.

Variable-length, Unblocked (Figure 94). Each logical record is the same as a physical record, and the logical records can vary in length. Each record contains both a block-length field (BL) and a record-length field (RL). These two fields are used by and created by the RPG program. The block-length and record-length fields are illustrated here only to show how the program utilizes and controls the records. Those fields are ignored by the user when he establishes his data records and when he specifies record-length and block-length specifications.

Variable-length, Blocked (Figure 95). One or more logical records (of variable-length) are contained within each physical record.

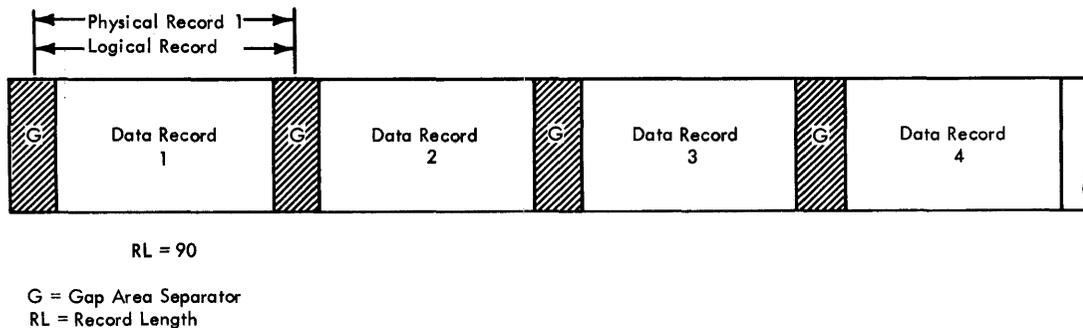


Figure 92. Fixed-Length, Unblocked Record Format

Specifications for Block Length

Unblocked

If the records are unblocked, the entry for this specification is the length of a record. If variable-length records are used, it is the length of the longest record. This means that the entry for this specification for unblocked records is always the same as the entry for the Record Length specification.

Blocked

If the records are blocked, the entry for the specification is the length of the largest block. For example, if three fixed-length 90-character records are contained in each block, the specification for Block Length is 270.

If there are variable-length records used in the file — for example Figure 95 — the Block Length specification depends upon not only the number of variable-length records in each block, but also upon the maximum size of each variable-length record.

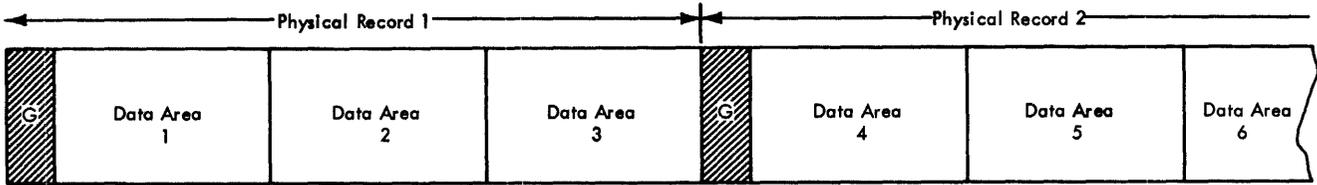


Figure 93. Fixed-Length, Blocked Record Format

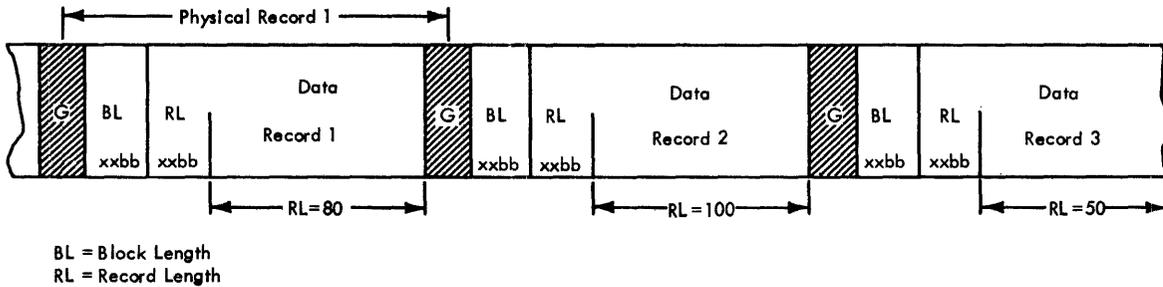


Figure 94. Variable-Length, Unblocked Record Format

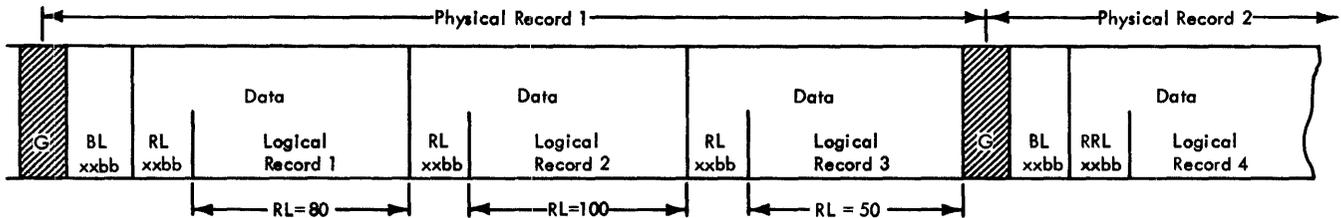


Figure 95. Variable-Length, Blocked, Record Format

If, in Figure 95, the three variable-length records (in physical record 1) will never exceed 80, 100, and 50 positions respectively, then the block length would be 230.

The entry must be right-justified, leading zeros may be omitted.

For blocked variable-length records, RPG allocates buffer space, depending on the type of I/O device specified, by either of the following formulas.

For printer or punch I/O devices,

$$\text{buffer size} = 4 + 5 \left(\frac{\text{block length}}{\text{record length}} + 3 \right) + \text{block length}$$

For all other I/O devices,

$$\text{buffer size} = 4 + 4 \left(\frac{\text{block length}}{\text{record length}} + 3 \right) + \text{block length}$$

RECORD LENGTH (24-27)

This specification is used to enter the length of the logical records contained in the file. If the file contains records that are variable in length, enter the length of the largest record. (The entry must be right-justified.)

NOTE 1: If the block length equals the record length, RPG considers the file to have unblocked records for both fixed-length and variable-length records.

NOTE 2: If the block length is greater than the record length, RPG considers the file to have blocked records for both fixed-length and variable-length records.

MODE OF PROCESSING (28) (Non-sequential DASD Only)

This specification is used to indicate the method or mode by which the file is processed. Acceptable entries are listed here.

Entry

- L Enter an L in this column if a segment of the file is to be processed. The upper and lower limits of the file will be supplied, in this case, by a Record Address File (RAF). The RAF will be supplied by the user. This entry applies to indexed-sequential files only.
- R Enter an R in this column if the user's records are to be processed randomly. In this case, the records to be processed will be obtained by a record-address file, or a chaining file. This entry applies to files with direct or indexed-sequential organization.

Blank If no entry is made in this column for the file, the entire file will be processed sequentially. This entry applies to indexed-sequential files only.

LENGTH OF RECORD ADDRESS FIELD (29-30) (Record Address File Only)

If the file being defined is a record-address file, enter the number of positions that each entry in the RAF occupies.

For example, if a six-position part number field is used in the RAF, the entry would be a 6.

RECORD ADDRESS TYPE (31) (Non-sequential DASD Only)

If the records from the file are to be retrieved by using record keys, enter a K in this column. The K indicates that the file defined on this line will be processed, using a record key.

If the records are to be retrieved by the record identification, enter an I in this column.

TYPE OF FILE ORGANIZATION (32) (Non-sequential DASD Only)

Enter an I if the file has indexed-sequential organization. Enter a D in this column if the file has direct organization.

RULES FOR DASD SPECIFICATIONS (Columns 28, 31, and 32)

The four rules listed below summarize the entries for the specifications Mode of Processing, Record Address Type, and Type of File Organization.

1. If a direct access storage device is not used in the system, columns 28, 31, and 32 are left blank.
2. If the type of file organization is Sequential, then columns 28, 31, and 32 are again left blank.
3. If the type of file organization is indexed-sequential (I in column 32), then column 31 must contain a K and column 28 can be either L, R, or blank.
4. If the type of file organization is direct (D in column 32), then column 31 can contain either a K or I, and column 28 can only contain an R.

Table 3 illustrates the code combinations possible for these three specifications.

OVERFLOW INDICATOR (33-34)

If the file defined on the line is a printer file or an output file with an associated Line Counter Specifications sheet and overflow indicators are used, enter the overflow indicator associated with the file. A maximum of eight overflow indicators is allowed. The following are permissible overflow indicators:

| | |
|----|----|
| OA | OE |
| OB | OF |
| OC | OG |
| OD | OV |

**KEY FIELD STARTING LOCATION (35-38)
(DASD Only)**

This specification indicates the location of the key field within the data record. This specification is provided so that the key field may be located anywhere within the data record.

The entry for this specification is the starting position of the key field. For example, if the key field is in positions 112 through 116, the entry would be 112.

The entry must be right-justified, leading zeros may be omitted.

This entry is required for indexed-sequential files only and is blank for directly organized files.

EXTENSION CODE (39)

This specification is used to indicate to the RPG processor that additional information about the file is coded on the File Extension Specifications sheet or Line Counter Specifications sheet.

Enter an E in this column if the file defined on the line is a:

- Chaining file
- Table file
- Record Address file (RAF)

These files always have additional specifications on the File Extension Specifications sheet.

Enter an L if the Line Counter Specifications sheet is used for the output file described on this line. If line counter is specified, TAPE or DISK11 must be specified under Device (40-46).

NOTE: It is possible to use the Line Counter Specifications sheet and still put out the data set directly to a printer. To accomplish this it is necessary to specify the Device as TAPE or DISK11 on the File Description Specifications sheet and to specify the printer as the output device via a DD card at execution time.

DEVICE (40-46)

This specification relates a file to a specific type of input or output unit during program compilation time.

If the output file is a printer, enter PRINTER in columns 40-46.

If the file is an input or output file and it is associated with a card reader or

Table 3. Processing Direct Access Storage Files

| Type of File Organization (Column 32) | Record Address Type (Column 31) | Mode of Processing (Column 28) |
|--|--|--|
| Sequential (blank) | Not applicable (blank) | The entire file (between limits) will be processed (blank) |
| Indexed-Sequential (I) | Record Key (K) | The entire file will be processed (blank) |
| | | A segment of the file will be processed (L) The limits to be processed are supplied by a Record Address File (RAF) |
| | | The records will be processed randomly (R) The addresses are supplied : (a) by an RAF, or (b) by the data contained in the chaining field of an input record. |
| Direct (D) | Track address with Record Key (K) or, Track Address with Record Identification (I) | The records will be processed randomly. (R) The addresses to be converted are supplied by an RAF, or by a chaining file. Conversion is required for RAF or chaining file. |

file. It is a chained file (C in column 16). It will be processed randomly (R in column 28). The record addresses that will be referenced by the chaining file are record keys (K in column 31), and the file is organized indexed-sequentially (I in column 32). The key field begins in position 46 of the record. This file is located on a direct access storage device and is given the Device code of DISK11.

5. The update file DISKUPDT (U in column 15) will be used for input, and it will be updated after processing of each record has been completed. It is an indexed-sequential file, and it will be processed randomly. The C in column 16 indicates that the file is a chained file. In this example, the record length is 200 with a block length of 400. The key field begins in position 1 of the record.

This file is located on a direct access storage device and is given the Device code of DISK11.

6. The file CARDREC is a combined file (C in column 15). Assume that the file will be used as input, and additional information will be punched in the input cards during processing. It is a secondary file (S in column 16). This combined file is read in and punched out on an IBM 1442 Card Read-Punch. The Device code is specified as READ42.
7. The file OUTPUT is a printed report in this example. It is variable in length, and the longest record is 132 characters. The overflow indicator for this file is OF (columns 33-34). This printed report has a Device code of PRINTER.

LABELS (53)

When label processing is used for a tape or disk, the program checks the labels on the input file to see if the correct file is on-line. The output files are checked, and if the label has expired, a new label is written.

If nonstandard labels are used for magnetic tape files, nonstandard label processing routines must be provided by the user and incorporated in the operating system at system generation time. (For

details refer to IBM System/360, Operating System, System Programmer's Guide.) The appropriate nonstandard label processing routine is selected and executed by the operating system.

Label Options in RPG

The specification Label (column 53) provides three options for label processing in the RPG program.

1. S Standard Labels. Label processing is provided by the RPG program. No additional programming is required by the programmer.
2. E Standard Labels Followed by User-Standard Labels. RPG provides processing for the standard labels and then provides an automatic exit to a user subroutine for the processing of the user-standard labels. (See next specification Name of Label Exit.) This option is not available for indexed-sequential files.
3. b No Labels. An entry of blank indicates no label processing is to be performed by the RPG program. This option is not available for DASD files.

NAME OF LABEL EXIT (54-59)

This specification must contain the name of the routine written by the user to process user standard labels (E in column 53). The name can be either alphabetic or numeric, but the first character must be alphabetic. If the entry is shorter than six characters, it must be left-justified.

Refer to Control Program Services (see Preface) for label exit register conventions.

EXTENT EXIT FOR DAM (60-65)

Not used.

COMMENTS (66-74)

Leave these columns blank unless comments are entered.

This concludes the description of the File Description Specifications sheet.

FILE EXTENSION SPECIFICATIONS SHEET

Entries made on the File Extension sheet provide information to RPG about such functions as chaining files and tables used in the object module, and information about record-address files. These functions are illustrated in Figure 97.

In the sections Using Tables in the Object Module and Processing Multiple Input Files detailed information and examples show how to use these functions.

The entries allowed on the File Extension sheet are discussed in the following section.

RECORD SEQUENCE OF THE CHAINING FILE (7-8)

This specification is used only for chaining files. The entry for this specification is the same entry that is made for the chaining file in Sequence (columns 15-16) on the Input Specifications sheet.

NUMBER OF THE CHAINING FIELD (9-10)

This specification is used only for chaining files. The entry for this specification is the identifying number of the chaining field (C1 through C9). This number

IBM

Date _____

Program _____

Programmer _____

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348-1
Printed in U. S. A.

| | | | | | | | | | |
|----------------------|---------------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic Punch | | | | | | | | |
|----------------------|---------------|--|--|--|--|--|--|--|--|

Page

Program Identification

| Line | Form Type | Record Sequence of the Chaining File | | Table Name | Number of Table Entries Per Record | | Table Name (Alternating Table) | Length of Table | | Comments | |
|------|-----------|--------------------------------------|-------------|------------|------------------------------------|-----------------------|-----------------------------------|-----------------|--|---|--|
| | | From Filename | To Filename | | Per Table | Length of Table Entry | | Entry | Packed (P) Decimal Pos. Sequence (A/D) | | Packed (P) Decimal Pos. Sequence (A/D) |
| | | | | | | | | | | | |
| 01 | E | | | | | | | | | | |
| 02 | E | ←→ | | | | | | | | Chaining Files Only | |
| 03 | E | | | | | | | | | | |
| 04 | E | | | | | | | | | | |
| 05 | E | | | | | | | | | | |
| 06 | E | | | | | | | | | | |
| 07 | E | ←→ | | | | | | | | Chaining Files RAF | |
| 08 | E | | | | | | | | | | |
| 09 | E | | | | | | | | | | |
| 10 | E | | | ←→ | | | | | | Label of Address Conversion Routine for RAF or Chaining Files | |
| 11 | E | | | | | | | | | | |
| 12 | E | | | | | | | | | | |
| 13 | E | | | | | | | | | | |
| 14 | E | ←→ | | | | | | | | Table Files | |
| 15 | E | | | | | | | | | | |
| 16 | E | | | ←→ | | | | | | 1st Table 2nd Table | |

Figure 97. Use of File Extension Specifications

was entered in Chaining Field (column 61-62) of the Input Specifications sheet.

FROM FILENAME (11-18)

This specification is used in conjunction with the next specification To Filename (19-26). The purpose of these two specifications is to identify—for the RPG program—the relationship between two files. For example, they provide the name of a chaining file and the name of the file that is chained to it. Both the From Filename and To Filename are taken from Filename (columns 7-14) of the related entry from the file description sheet.

Figure 98 illustrates the entries for those two specifications.

TO FILENAME (19-26)

The entries for this specification are described above and in Figure 98.

TABLE NAME (27-32)

This specification and the remaining specifications on the form (columns 33-57) are used to describe table files. Table files are processed in the same order in which they appear on the File Extension Specifications sheet. This specification is also used to specify the name of the address conversion routine for an RAF or chaining file.

A table file is composed of a table of arguments and a table of functions. If

both the arguments and functions are entered on one input unit, the table file is known as an "alternating" table file. That is, the input record contains an argument field followed by a function field, followed by the next argument field, etc. In this case, the argument table is described in columns 27 through 45, and the function table is described in columns 46 through 57 on the same specification line.

If only an argument table is used, it is still described in columns 27-45, and columns 46-57 are left blank.

It is possible to have the arguments contained in one input file and the functions in another input file. In this case, each file is described on a separate specification line in columns 27 through 45, and columns 46-57 are left blank.

NOTE: It is not a requirement of the program that the arguments must be specified first. The function entries may be listed first, however for the following specification descriptions the manual assumes the argument entries are specified first.

Specifications for Table Name

This specification contains the name of the argument table. The name must be in the form TABnnn. The entries nnn may be any alphanumeric characters.

When a file that has direct organization is processed under control of a record-address file or a chaining file, the entry for these columns is the label of the user's conversion routine.

| Type of File | * From Filename (11-18) | * To Filename (19-26) |
|---------------------|--|--|
| Chaining Files | <u>Chaining Filename</u> . This is the file that has the data record containing the chaining field. The name of the file is taken from columns 7-14 of the File Description Sheet. | <u>Chained Filename</u> . This is the file from which the data record is obtained. The name of the file is taken from columns 7-14 of the File Description specification for the Chained File. |
| Record Address File | The name of the record-address file is entered in this specification. | The name of the file that contains the data record to be processed is entered in this specification. |
| Table Files | If a Table file is being defined (columns 27 through 57) enter the name of the file that contains the table. | If the table file being defined, will be put out after it is updated, enter the name that was assigned to it on the Output-Format sheet. If it is not being put out, leave blank. |

* All entries must be left-justified

Figure 98. From and To Filenames

Figure 99 illustrates the File Extension specifications for a Record Address file (RAF) and the master customer file it is used with.

The concepts of chaining and record-address files have not been discussed at this point in the publication. For a complete discussion of chaining, address conversion, and record-address files, see Processing Multiple Input Files.

NUMBER OF TABLE ENTRIES PER RECORD (33-35)

Enter in columns 33-35 the maximum number of table entries (that is, arguments or functions) that are contained in each input record. The entry must be right-justified.

NUMBER OF TABLE ENTRIES PER TABLE (36-39)

In these columns, enter the exact number of table entries (arguments or functions) contained in the table. The entry must be right-justified.

NOTE: The above two entries refer to tables, not to files. Thus, in alternating table files the entries are the total number of arguments or functions, not the sum of the two.

LENGTH OF TABLE ENTRY (40-42)

Enter in columns 40-42 the length of each table entry. The maximum size of a numeric entry is 15 characters, of an alphameric entry 256 characters. The entry must be right-justified.

PACKED (43)

If the data for the table is in the packed-decimal format, enter P in this column. Otherwise, leave this column blank.

DECIMAL POSITIONS (44)

If the data contained in the table is numeric, enter the number of decimal positions (1-9). Enter a zero if there are no decimal positions. If the data is alphameric, leave this column blank.

SEQUENCE (45)

If the data contained in the table is in ascending sequence, enter an A in this column. If the data contained in the table is in descending sequence, enter a D in this column. Leave this column blank if the data contained in the table is not in ascending or descending sequence or if this specification is not required.

NOTE: The next four specifications (columns 46-57) are used only if alternating arguments and functions are read in on one input unit. The entries for these specifications are written on the same specification line as the entries in columns 33-45.

TABLE NAME (46-51)

If alternating arguments and function tables are used, enter the second table name in these columns. It must be of the form TABnnn. The entry must be left-justified.

LENGTH OF TABLE ENTRY (52-54)

Enter in these columns the length of each table entry. The maximum size of a numeric entry is 15 characters; of an alphameric entry 256 characters. The entry must be right-justified.

PACKED (55)

Enter a P if the data for the table is in the packed decimal format. Otherwise leave this column blank.

DECIMAL POSITIONS (56)

If the data contained in the table is numeric, enter the number of decimal positions

IBM INTERNATIONAL BUSINESS
REPORT PROGRAM GENERATOR IBM System/370
 Date _____
 Program _____
 Programmer _____

| Line | Form Type | Record Sequence of the Chaining File | | Table Name | Number of Table Entries | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------|-----------|--------------------------------------|---|------------|---------------------------|-----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| | | Number of the Chaining Field | | | Number of Table Per Table | Length of Entry | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 80 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 | 90 | 91 | 92 | 93 | 94 | 95 | 96 | 97 | 98 | 99 | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 |

(1-9). Enter a zero if there are no decimal positions. If the data is alphameric, leave this column blank.

SEQUENCE (57)

If the data contained in the table is in ascending sequence, enter an A in this column. If the data contained in the table is in descending sequence, enter a D in this column. Leave this column blank if the data contained in the table is not in ascending or descending sequence or if this specification is not required.

COMMENTS (58-74)

Leave columns 58-74 blank, unless comments are entered in these columns.

ENTRIES ON THE FILE EXTENSION SHEET

Figure 100 shows several examples of entries that can be made on the File Extension sheet. The numbers to the right of the entries correspond to these explanations:

1. In this example, INPUT is a card file containing the record key that will be used to process records in the DASD file MASTCUST. The file INPUT is the chaining file. That is, it is the file that links or chains to another file

(in this case MASTCUST). The field contained in INPUT, which is used to link the two files, is the chaining field.

The record sequence of the input file is taken from the Input Specifications sheet. C1 is the number of the chaining field. Thus, INPUT is chained to MASTCUST by using a field defined on the input specifications that contains C1 in columns 61-62. A complete discussion of chaining may be found in Chaining at the back of this publication.

2. In this example, RAFFILE is a record-address file that supplied the addresses of the records to be processed in the file DISKUPDT.
3. TABFIL is the name of a table file that contains both a table of arguments and a table of functions.

The arguments in the file are identified by the table name of TABARG. The argument table is described in columns 33-45. There are 10 arguments in each record. The number of arguments in the table is 150 and each argument is 10 characters long. The arguments are numeric and there are no decimal positions, thus the entry is 0 (column 44).

The functions in the file are identified by the table name TABFUN. The function table is described in columns 52-57. Each function is 10 characters long and each function is organized in

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS
 IBM System/360

Form X24-3348-1
 Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

75 76 77 78 79 80
 Program Identification

| Line | Form Type | Record Sequence of the Chaining File | | | | | | | | | | | | | | | | Number of Table Entries Per Record | | | | | | | | | | Table Name | | | | | | | | | | Length of Table | | | | | | | | | | Comments |
|------|-----------|--------------------------------------|-------------|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|------------------------------------|-----------------------|--------------------------------|----|--------|----|----|----|----|----|------------|----|-------|-----------------|----|----|----|----|----|----|-----------------|----|----|----|----|----|----|----|----|----|----------|
| | | Number of the Chaining Field | | | | | | | | | | | | | | | | Per Table | Length of Table Entry | Table Name (Alternating Table) | | | | | | | | | | Entry | Length of Table | | | | | | | | | | | | | | | | | |
| | | From Filename | To Filename | Table Name | 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 | 58 | 59 | 60 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 | 70 | 71 | 72 | |
| 01 | E | AAC1 INPUT MASTCUST | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ① | | | | | | | |
| 02 | E | RAFFILE DISKUPDT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | ② | | | | | | | |
| 03 | E | TABFIL TABARG | | | | | | | | | | | | | | | | 10 | 150 | 10 | | TABFUN | | | | | | | | | | 10 | | | | | | | | | | | | ③ | | | | |
| 04 | E | TABFIL NEWTAB | | | | | | | | | | | | | | | | 10 | 150 | 10 | | | | | | | | | | | | | | | | | | | | | | | | ④ | | | | |
| 05 | E | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 100. Entries for the File Extension Sheet

the form: argument-function. Therefore, TABARG was specified first.

4. This example shows the specifications for a table file that contains only arguments. After the table of arguments is updated the table is to be written on an output unit.

TABFIL is the name of the table file.

NEWTAB is the name given to the table file when it is being written

on the output unit (output operations).

The arguments in the file are identified by the table name of TABREC. The argument table is described in columns 33-45. Ten table entries are in each record. The number of table entries in the table is 150, and each table entry is 10 characters long. The data is numeric, but there are no decimal positions, thus the entry is 0. Columns 46 through 57 are left blank.

USING TABLES AND EXIT ROUTINES IN THE OBJECT MODULE

This section of the publication contains information on:

1. How to create and use tables, and
2. How to transfer control from the RPG program to a subroutine coded by the user, and how to return to the RPG program.

USING TABLES IN THE OBJECT MODULE

RPG enables the programmer to use tables in the object module. A table is nothing more than a systematic arrangement of data which is used by the object program in much the same manner that a shipping clerk would use a rate table for obtaining freight rates. The clerk might scan the table for the desired city and then select the corresponding rate.

A table may consist of two parts: an argument and a function. In Figure 101 the table consists of part numbers (arguments) and prices (functions). If the price of part number 10 is wanted, the table is searched until part number 10 is found. The corresponding function of 10 in the table is 155. (The 155 represents \$1.55, in this example.) The number used to search the table is called the search argument. The card file in Figure 101 contains part numbers that have been placed in the table in a predefined sequence. The cards do not contain the prices of the parts. The part

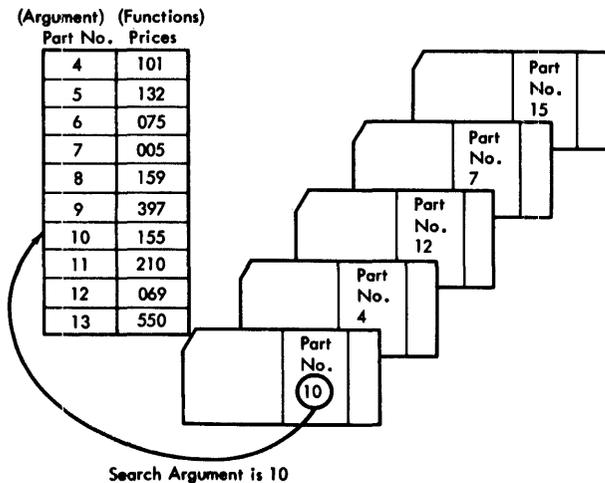


Figure 101. Using a Table

number is selected from the card by the RPG program, the table is searched, and the price is retrieved and made available for additional processing. Tables are loaded into storage by the RPG object module before any files are processed.

All entries in a table will be:

1. Arguments,
2. Functions,
3. Alternating arguments and functions, or
4. Alternating functions and arguments.

Figure 102 shows these four possibilities.

Rules for Forming Tables

1. Each unit of table data is called a table entry. That is, each argument is a table entry, and each function is a table entry.
2. The collection of all argument entries is assigned a name. The collection of all function entries is assigned a name.

These table names must be unique, and must contain the letters TABnnn (nnn may be any alphameric entry). In

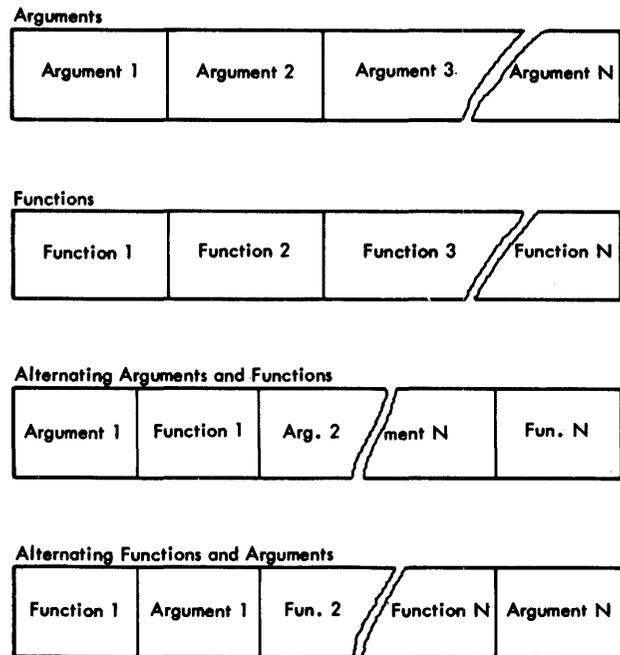


Figure 102. Four Types of Tables

Figure 106 the alternating table file called RATETABL is split into the collection of arguments (TABNUM) and the functions (TABRAT). RATETABL is the name of the file containing these two tables.

3. All tables may be loaded from the same device in the order they are specified on the File Extension sheet. The tables will be loaded into storage before the object module is processed, and each line entry on the File Extension sheet must have:
 - a. A file name (columns 11-18). Multiple file extension specifications for any table file are allowable.
 - b. Entries in columns 27-45 if the table is only arguments or functions.
 - c. Entries in columns 27-45 and columns 46-57 if the table consists of alternating arguments and functions.

Rules for Creating Records Containing Table Data

1. Each record must begin with the first table entry of that record in position 1.
2. All records must have the same number of table entries, except the last. In Figure 104, the first card in the table file has seven table entries. All subsequent card records must have seven table entries. For example, the second card could not contain six; the third could not contain eight.
3. All entries must be adjacent in every record. In Figure 103, the first entry begins in Position 1 and the second entry begins in position 4. No blanks can be contained between the table entries.
4. All entries belonging to a table must have the same length. In Figure 103, each argument is three positions long, and each function is six positions long.

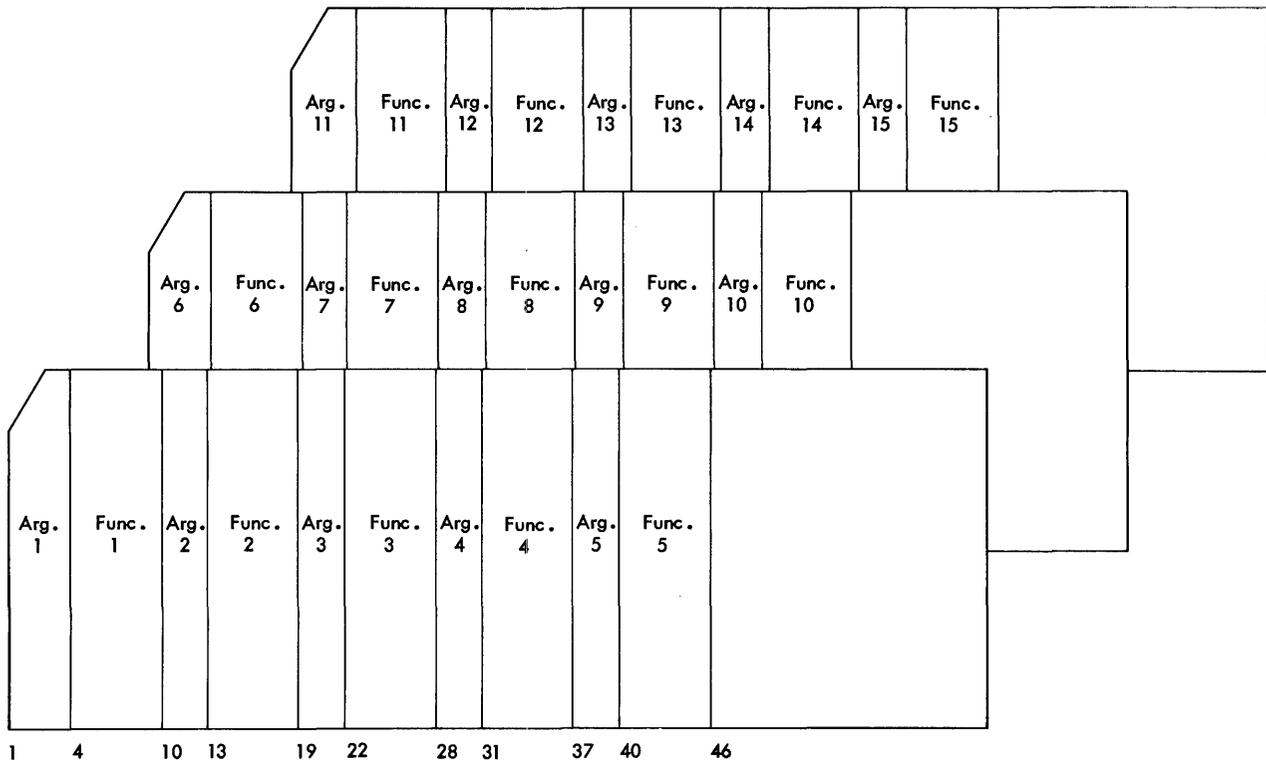


Figure 103. Table File Containing Arguments and Functions

If the updated table is to be put out on a printer, no automatic skip to a new page will be initiated by the RPG program.

The table written out has the same format as specified for the input table in columns 33-45 or 52-57 on the File Extension Specifications sheet.

METHODS OF PROCESSING TABLES

The operation code LOKUP entered on the Calculation Specifications sheet causes a table lookup operation to be performed.

Factor 1 contains the search argument. The search argument may be a literal or a field name.

NOTE: The length of the data in the argument table (table argument) must be equal to the length of the search argument. The length includes the decimal positions.

Factor 2 contains the name of the table which contains the arguments.

The Result Field contains the name of the table from which an associated function is to be located.

The Result Field may be left blank if the user wants to determine if an argument is present in the table, but does not require the corresponding function.

Resulting Indicators (columns 54-59) must always have an entry when the table lookup operation is performed. The presence of indicators in this specification indicates the type of lookup to be performed. The indicators are turned on whenever the condition is satisfied.

The program may search for the table argument next-higher than the search argument, or it may search for the table argu-

ment next-lower than the search argument, or it may search for the table argument equal to the search argument. An entry must be made in columns 54-59. Combinations of high-equal or low-equal searches may be specified by placing indicators in the appropriate two of the three fields (columns 54-59).

Performance of LOKUP

The lookup operation is performed in this way:

1. The object module takes the field name or literal in Factor 1 and searches the table indicated by Factor 2. The kind of lookup is determined by the entries in Resulting Indicators.
2. After the proper entry from the argument table has been found, the corresponding function from the function table indicated by the entry in Result Field is located and placed in the special holding area of the function table. If the proper table argument is not found, the indicators in columns 54-59 are not turned on.

Using LOKUP Data Obtained

Other operations may be performed using the data just found by the table lookup operation. This data is stored in the special holding area within the function table and can be retrieved by merely using the name of the function table in either Factor 1 or Factor 2 of an operation.

Figure 105 illustrates several ways the data found by the operation may be used. The numbers on the figure correspond to this discussion.

EXAMPLE OF USING TABLES

Figures 106 and 107 illustrate an input data file, the way a table might appear, and the entries necessary on the RPG specification sheets.

In this example, a card-input file contains the number of hours worked by each employee (columns 42-44) and the employee's number (columns 1-5). The RPG program takes the employee number and uses it as the search argument to find the salary rate for the employee. After the salary rate has been found, it is multiplied by the hours worked by the employee. The result of this operation is the amount earned for each employee.

In this example, the table consists of alternating arguments and functions. The way the table data might appear is shown in Figure 106. The name of the file that contains the arguments and functions is RATETABL. The collection of arguments is called TABNUM (table number), and the collection of functions is called TABRAT (table rate).

Entries on the specification sheets follow.

File Description Specifications Sheet

The two files are defined on the File Description sheet. The file containing the input card records is called TIMECARD. It is an input file (I in column 15); it is the primary file (P in column 16); and when the file is depleted, processing is terminated (E in column 17). The records in the file are in ascending order (A in column 18); they are fixed-length records (F in column 19). Each record has a block length of 80 (80 in columns 22-23), and each record is 80 characters long (80 in columns 26-27). This file is read in on the IBM 2501 Card Reader, so the device code is READ01.

The table file is defined on the line below the card-input file. The name of the

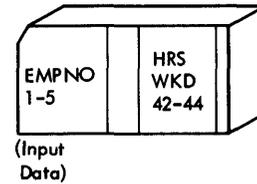
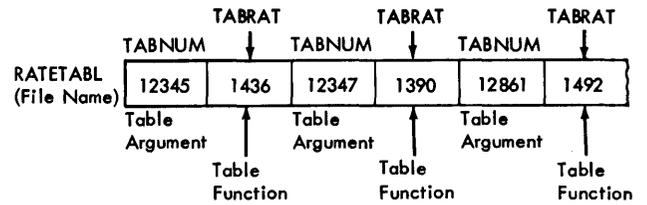


Figure 106. Using Alternating Arguments and Functions

file (RATETABL) is entered in Filename (columns 7-14). It is an input file (I in column 15), and the records in the file are fixed-length (F in column 19). The file has a block length of 80, and each record is 80 characters long. The E in column 39 indicates that additional information about the file is coded on the File Extension sheet. This file is read in on the IBM 1442 Card Read-punch, and therefore it is assigned the Device code of READ42.

File Extension Specifications Sheet

On the File Extension sheet, the table file is further defined. The name of the file is entered in From Filename (columns 11-18). The collection of arguments (TABNUM) is entered in the first Table Name (columns 27-32). There are 8 arguments per record (columns 34-35), and there are 1500 arguments in the table (columns 36-39). Each table entry is five positions long (5 in column 42), and there are no decimal positions (0 in column 44). The table is ascending (A in column 45).

| Line | | Form Type | Filename | File Type | File Designation | Sequence | Block Length | Record Length | L/R | Mode of Processing | Record Address Type | Type of File Organization | Overflow Indicator | Key Field Starting Location | Extension Code E/L | Device | Symbolic Device | Name of Label Exit | Extent Exit for DAM | Comments |
|------|---|-----------|----------|-----------|------------------|----------|--------------|---------------|-----|--------------------|---------------------|---------------------------|--------------------|-----------------------------|--------------------|--------|-----------------|--------------------|---------------------|----------|
| 01 | F | | TIMECARD | IPEAF | | | 80 | 80 | | | | | | | READ01 | | | | | |
| 02 | F | | RATETABL | IT | | | 80 | 80 | | | | | | | READ42 | | | | | |

Figure 107. Coding Sheets for Table Example (Part 1 of 2)

The collection of functions is described in columns 46-57. The name of the functions (TABRAT) is entered in the second Table Name (columns 46-51). Each entry in the table is four positions long (4 in column 54), and there are three decimal positions specified (3 in column 56).

as specified by the entries in Field Location (columns 47 and 51), and the employee number is given the field name EMPNUM. The number of hours worked by the employee is found in columns 42-44 of the input record, as specified by the entries in Field Location. The name HRSWKD is assigned to the number of hours worked by each employee.

Input Specifications Sheet

The input file (TIMECARD) is described on the Input Specifications sheet. The name of the file is entered in columns 7-14. The file is assigned a sequence of AA (columns 15-16), and resulting indicator 01 is turned on whenever an input record is present for processing. No record identification codes are specified because every record will be processed in the same way.

Lines 020 and 030 are used to describe the locations of the two input fields used by the program. The employee number is located in columns 1-5 of the input record,

Calculation Specifications Sheet

Three calculation specifications are shown. On line 010, EMPNUM (employee number) is used as Factor 1. The employee number is the search argument. The operation code LOKUP which is coded in Operation (columns 28-32) causes the lookup operation to be performed. Factor 2 contains the name of the collection of arguments (TABNUM) which is searched. The Result Field contains the name of the collection of functions (TABRAT). Thus, this operation causes the employee

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3348-1
REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS Printed in U.S.A.
 IBM System/360

Date _____
 Program _____
 Programmer _____

| | | | |
|----------------------|---------------|----------|--|
| Punching Instruction | Graphic Punch | Page 1 2 | Program Identification 75 76 77 78 79 80 |
|----------------------|---------------|----------|--|

| Line | Form Type | Record Sequence of the Chaining File | | Table Name | Number of Table Entries Per Record | | Table Name (Alternating Table) | Length of Table Entry | | Comments |
|------|-----------|--------------------------------------|-------------|------------|------------------------------------|-----------------------|--------------------------------|-----------------------------|---|----------|
| | | From Filename | To Filename | | Per Table | Length of Table Entry | | Decimal Pos. Sequence (A/D) | | |
| 01 | 0E | RATETABL | | TABNUM | 81500 | 5 | TABRAT | 4 | 3 | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3350-1
REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS Printed in U.S.A.
 IBM System 360

Date _____
 Program _____
 Programmer _____

| | | | |
|----------------------|---------------|----------|--|
| Punching Instruction | Graphic Punch | Page 1 2 | Program Identification 75 76 77 78 79 80 |
|----------------------|---------------|----------|--|

| Line | Form Type | Filename | Sequence Number (1-N) Option (O) | Resulting Indicator | Record Identification Codes | | | Field Location | | Field Name | Control Level (1,1,1P) Matching Fields, or Chaining Fields | Field Indicators | | Sterling Sign Position |
|------|-----------|------------|----------------------------------|---------------------|-----------------------------|-----------|----------|----------------|------|------------|--|------------------|------|------------------------|
| | | | | | Position | Character | Position | Character | From | | | To | Plus | |
| 01 | 0I | TIMECARDAA | 01 | | | | | | | | | | | |
| 02 | 0I | | | | | | | 1 | 50 | EMPNUM | | | | |
| 03 | 0I | | | | | | | 42 | 44 | HRSWKD | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3351-1
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS Printed in U.S.A.
 IBM System 360

Date _____
 Program _____
 Programmer _____

| | | | |
|----------------------|---------------|----------|--|
| Punching Instruction | Graphic Punch | Page 1 2 | Program Identification 75 76 77 78 79 80 |
|----------------------|---------------|----------|--|

| Line | Form Type | Control Level (0,1,9, LR) | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Resulting Indicators | | | Comments |
|------|-----------|---------------------------|------------|-----|-----|----------|-----------|----------|--------------|--------------|----------------------|-------|---------------|----------|
| | | | And | And | And | | | | | | Plus | Minus | Zero or Blank | |
| 01 | 0C | | | | | EMPNUM | LOKUP | TABNUM | TABRAT | 43 | | | | 03 |
| 02 | 0C | | | | | TABRAT | MULT | HRSWKD | EARN\$ | 52H | | | | |
| 03 | 0C | | | | | N03 | MOVE | +000.00 | EARN\$ | | | | | |

Figure 107. Coding Sheets for Table Example (Part 2 of 2)

number (EMPNUM) to be used as the search argument for the data contained in TABNUM. The result field is four positions long with three decimal positions. The 03 entered in columns 58-59 indicates that indicator 03 will be turned on when the search argument finds an entry in the argument table that is equal to the search argument.

The specifications on line 020 are performed when indicator 03 is on. The rate for the employee (TABRAT) which has just been located is multiplied by the number of hours worked (HRSWKD), and the result is stored in EARN\$ which is five positions long and has two decimal positions. The answer is half-adjusted.

If the search argument does not find an equal entry in the argument table (indicator 03 is not on), the specifications on line 030 are performed. Columns 9-11 contain the specification N03.

The literal +000.00 is then moved to the field EARN\$, specifying that the employee does not have an entry in the table.

EXIT TO A USER'S SUBROUTINE

GENERAL INFORMATION

By use of the EXIT operation code on the Calculation Specifications sheet, RPG provides the facility to transfer control from the RPG object program to some subroutine that has been coded independently. A subroutine might be a standard routine such as a state withholding tax.

The subroutine, written in the assembler language, is coded by the user, and entries made on the calculation sheet enable the programmer to:

1. Exit from the RPG program to the subroutine,
2. Execute the subroutine,
3. Reference fields and indicators defined in the RPG program (RLABL usage),
4. Reference fields defined in the user's subroutine (ULABL usage), and
5. Return to the main program after the subroutine has been performed.

HOW TO CODE EXIT

On the Calculation Specifications sheet, the EXIT operation can be a conditional operation. When entries are placed in columns 7-8, 9-11, 12-14, or 15-17, the EXIT will occur when the designated conditions are satisfied. If no indicators are used, the

EXIT will occur everytime the detail calculations are performed. Columns 28-31 must contain the operation code EXIT and Factor 2 must contain the label of the user's subroutine. The subroutine name may be from 1-6 alphameric characters with the first character being alphabetic. Factor 1 is not used.

POSITION OF EXIT IN THE CALCULATION SPECIFICATIONS

The following results will be obtained depending on the location of the EXIT code on the Calculation Specifications sheet.

| <u>Calculation Entry</u> | <u>When the Exit Will Occur</u> |
|--------------------------|--|
| 1. First Detail | At the end of the data routine (after the data is extracted from the input record). |
| 2. Last Detail | Immediately before heading records are written. |
| 3. First Total | At the end of the input routine (after the record-type has been determined and the control-field break has been tested). |
| 4. Last Total | Immediately before the total records are written. |

GENERAL RULES FOR USING EXIT

RPG provides the facility for the subroutine to test indicators and use tables and fields that have been defined in the RPG program. RPG also provides the facility for the RPG program to use fields that have been defined in the subroutine. These two facilities are provided by using the two operation codes RLABL and ULABL. RLABL and ULABL operations can be specified anywhere within the calculation specifications.

RLABL

If the user has defined a field or table in the RPG program and it is to be used in the subroutine to which the EXIT will occur, he must code:

1. RLABL in Operation.
2. The name of the field or table in Result Field.
3. The length of the field in Field Length.
4. The decimal indication in Decimal Positions.

The user may need to reference, in the subroutine, indicators that are used in the RPG portion of this program. To do this, the user must code:

1. RLABL in Operation.
2. INnn in Result Field. The nn represents the specific indicator that the user wants to test in the subroutine. Therefore, if MR was to be tested in the subroutine, he would code INMR in Result Field.

ULABL

If the user has defined a field in the subroutine, and this field is to be used in this RPG program he must code:

1. ULABL in Operation.
2. The name of the field in Result Field.
3. The length of the field in Field Length.
4. The decimal indication in Decimal Positions.

When executing the subroutine, the user may have to use an indicator in the subroutine, and later reference that indicator in the RPG program. This can only be accomplished by first defining the indicator in the RPG program and then defining it in a RLABL operation.

USE OF REGISTERS

The way in which registers are used by the programmer is strictly defined. These rules must be followed:

1. The using register that contains the entry address of the called subroutine is register 15.
2. When control of the program passes from the RPG program to the subroutine, the address of the RPG instruction to which the subroutine must return is stored in register 14.
3. The RPG instruction to which the subroutine returns is the instruction that follows the EXIT operation.
4. If registers are used within the subroutine the contents of the registers the programmer intends to use must be stored before the subroutine is executed.
5. Before the subroutine transfers back to the RPG program, the registers must be restored to their original contents.

USING INDICATORS, FIELDS, AND TABLES IN THE EXIT ROUTINE

Indicators

If, in the exit subroutine, the user sets on, sets off, or tests indicators, he must observe the following rules:

1. To set on an indicator, set the data located at INnn to hexadecimal F0.
2. To set off an indicator, set the data located at INnn to hexadecimal 00. (Indicator L0 and 00 must never be set off.)
3. To test indicators:
 - a. If on, the data at INnn will be hexadecimal F0.
 - b. If off, the data at INnn will be the hexadecimal 00.

Fields

If numeric data from the RPG object program is used in the subroutine, it will be in the packed-decimal format. If numeric data from the subroutine is supplied to the RPG object program, it must be in the packed-decimal format.

Tables

The subroutine may refer to a table which is defined in the RPG program. As each table is created in the program, a "Table Linkage Field" is created for it. This field is a control field utilized by table operations. The format of this field is illustrated in Figure 108. Significant subfields are described below.

1. This subfield (1 byte) contains switches used by RPG.
2. This subfield (1 byte) contains the length minus 1 of each table entry. For numeric fields this subfield contains a number one less than the unpacked length although the actual table entry is in packed format.
3. This subfield (2 bytes) contains the number of entries in the table.
4. This subfield (4 bytes) contains the address of the beginning of the table.
5. This subfield (4 bytes) contains the address of the byte following the end of the table.
6. This subfield (4 bytes) is used by the RPG object program as a pointer to the selected table entry. For example, as

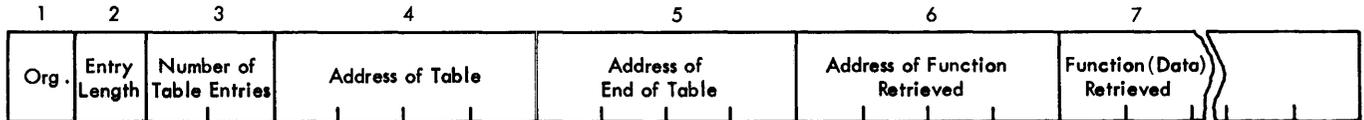


Figure 108. Format of Table Linkage Field

- the result of a LOKUP operation, this subfield contains the address of the corresponding function retrieved from the table when an equal is found in the argument table.
7. This subfield is used by the object module as a transient work area. For example, as the result of a LOKUP operation this subfield contains the data from the function retrieved from the table when an equal is found in the argument table. (This subfield is word-aligned and its length is equal to the length of a table entry.) This area is also called the "special holding area".
- The subroutine can use the data retrieved from a preceding LOKUP operation by simply referring to TABnnn. (Assuming that TABnnn has been defined by an RLABL operation.) The effective address of any reference to TABnnn is the first byte of subfield 7. To access the table itself, the address contained in subfield 4 of the Table Linkage Field for TABnnn must be used.
- EXAMPLE OF EXIT TO A SUBROUTINE**
- Figure 109 shows the coding steps necessary to implement the EXIT routine.
1. An input file of the name INPUT turns on resulting indicator 01 if an X is in position 80.
 2. If the field AMOUNT is zero or blank, field indicator 02 is turned on.
 3. The operation SETOF defines (and sets off) indicator 14 for the RPG program so that it can be subsequently defined for use in the subroutine. (This is performed only if indicator 01 is on.)
 4. Whenever indicator 01 is on, the calculation specifications entry EXIT causes the program to exit to the user's subroutine called TAXRTE.
 5. Within the subroutine, the user wants three things: the AMOUNT field, and the indicators 02 and 14. The RLABL operations enable the subroutine to reference the AMOUNT field, to test indicator 02, and to utilize indicator 14 in the subroutine.
 6. In the subroutine, assume there is a field (TAXAMT) that the user wants to use on the output-format specifications. If the field TAXAMT in the subroutine is blank, the subroutine turns on indicator 14. TAXAMT is referenced in the output specifications, but it will not be printed if indicator 14 is on. If the user chooses, he can use it later in the calculation specifications. The entry ULABL enables the field TAXAMT to be referenced by the RPG program.
 7. On the output sheet, TAXAMT is treated as a field.
- This concludes the description of tables and EXIT subroutines.

RPG enables the programmer to branch from detail calculations to total calculations causing the generated program to repeat certain specifications. This repetition is called a detail-to-total loop. Essentially, this gives the programmer the facility to put out one or more total lines for each input record. Each total line may vary in content according to program specifications and may be put out during a detail-to-total loop, depending on its output indicators.

Certain restrictions are imposed on the use of the detail-to-total branch by the flow of logic used by RPG. These restrictions should be understood by the programmer before using the detail-to-total branch. A detailed example is provided by Figures 109.1, 109.2, and 109.3.

DATA MOVEMENT FROM INPUT AREA TO FIELD LOCATIONS

Each time detail-to-total loop is executed, data items are moved from the input area to their respective field locations. If an input field is to be used as a result field and changed by calculations during a detail-to-total loop, the changes in the result field will be overlaid when the input fields are reinitialized to their input area values. Therefore, a new field should be defined and the input item moved to this new field, which can be then used as a result field. These results can thus be protected.

An indicator can be used to allow the move from an input field to a new field only once, usually during the normal flow of logic before a detail-to-total branch. This indicator can be used to prevent the new field from being overlaid with the reinitialized input field.

SUPPRESSION OF NORMAL ACTIVITY

Before performing a detail-to-total branch, total calculations, total lines, detail calculations, and detail lines may have to be suppressed to get the desired results. This may be done by conditioning lines and calculations with indicators. One such suppression might be to allow data to be moved from an input field to a protected field only once.

BRANCHING

The branch from detail-to-total calculations is accomplished by a GOTO specification in

detail calculations and a TAG in total calculations. Branching is controlled by conditioning the GOTO specification on an indicator. This indicator is set on during detail calculations to initiate the detail-to-total branch. The results of a total calculation performed during the detail-to-total loop may be used to set off this indicator and thus discontinue the loop.

The indicators used to control a detail-to-total branch must be specified with care. These indicators must be specified so that they initiate the detail-to-total branch; but when the desired results of the detail-to-total loop have been obtained, these indicators must inhibit the branch from further initiation. The generated program must resume normal logic flow and read the next input record. Failure to control the detail-to-total loop in this manner may result in an infinite repetition of the detail-to-total branch.

OUTPUT

The sequence of output lines produced by the generated program is altered by the use of the detail-to-total branch:

1. Total lines produced during a detail-to-total loop are put out first.
2. Detail lines conditioned for output at detail time follow.
3. Finally, all normal total lines (not conditioned as part of the detail-to-total loop) are put out according to their indicators.
4. If the overflow indicator is turned on while printing total lines in a detail-to-total loop, all lines conditioned on the overflow indicator will be put out before that indicator is turned off.

EXAMPLE

In the following example (Figures 109.1, 109.2, and 109.3) seven fields are read on input. They are: NAME, ADDRESS, LOCATE, MONTH, INITIL, NUMBER, and INCRMT. The desired output is a set of mailing labels, addressed NAME, ADDRESS, and LOCATE. These labels are dated. The first label bears the date MONTH, INITIL, where INITIL is the day of the month. The day of the month on subsequent labels is incremented by INCRMT. The field NUMBER determines how many labels



INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
 IBM System/360

Form X34-318-1 U/A025
 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

| | | | |
|---------------------|---------------|----------|--|
| Funding Instruction | Graphic PUNCH | Page 1 2 | Program Identification 75 76 77 78 79 80 |
|---------------------|---------------|----------|--|

| Line | Item Type | Control Level | Indicators | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Field Position | Resulting Indicators | | | | Comments |
|------|-----------|---------------|------------|-----|----------|-----------|----------|--------------|--------------|----------------|----------------------|-------|---------------|---------|----------|
| | | | And | And | | | | | | | Plus | Minus | Zero or Blank | Compare | |
| 01 | C | | N | 77 | | Z-ADD | INCRMT | INC | 2 | 0 | | | | | |
| 02 | C | | N | 77 | | Z-ADD | INITIL | DAY | 2 | 0 | | | | | |
| 03 | C | | N | 77 | DAY | SUB | INC | DAY | | | | | | | |
| 04 | C | | N | 77 | INC | MULT | NUMBER | TEMP | 3 | 0 | | | | | |
| 05 | C | | N | 77 | TEMP | ADD | DAY | TEMP | | | | | | | |
| 06 | C | | | | | SETON | | | | | | 77 | | | (6) |
| 07 | C | | | | DAY | COMP | TEMP | | | | 998899 | 77 | | | (7) (11) |
| 08 | C | | 88 | 99 | | GOTO | TOTAL | | | | | | | | (8) |
| 09 | C | | N | 88 | 99 | SETOP | | | | | | 77 | | | (12) |
| 10 | C | | L | 0 | TOTAL | TAG | | | | | | | | | (3) |
| 11 | C | | L | 0 | DAY | ADD | INC | DAY | | | | | | | (9) |



INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
 IBM System/360

Form X34-332-1 U/A 025
 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

| | | | |
|---------------------|---------------|----------|--|
| Funding Instruction | Graphic PUNCH | Page 1 2 | Program Identification 75 76 77 78 79 80 |
|---------------------|---------------|----------|--|

| Line | Item Type | Filename | Type (D/D/T) | Spaces | Skip | Output Indicators | | Field Name | Field Position | End Position in Output Record | Constant or Edit Word | Sterling Sign Position |
|------|-----------|----------|--------------|--------|------|-------------------|-----|------------|----------------|-------------------------------|----------------------------|------------------------|
| | | | | | | And | And | | | | | |
| 01 | O | OUTPUT | H | 12 | 01 | | IP | | | | | |
| 02 | O | | OR | 12 | 01 | | OF | | | | | (1) |
| 03 | O | | | | | | | | 43 | | 'PRINTING MAILING LABELS' | |
| 04 | O | | D | 22 | | | 01 | | | | | |
| 05 | O | | | | | | | NUMBERZ | 11 | | | |
| 06 | O | | | | | | | | 36 | | 'LABELS HAVE BEEN PRINTED' | |
| 07 | O | | | | | | | | 41 | | 'FOR' | |
| 08 | O | | D | 11 | | | 01 | | | | | (13) |
| 09 | O | | | | | | | NAME | 25 | | | |
| 10 | O | | D | 11 | | | 01 | | | | | |
| 11 | O | | | | | | | ADDRESS | 25 | | | |
| 12 | O | | D | 11 | | | 01 | | | | | |
| 13 | O | | | | | | | LOCATE | 26 | | | |
| 14 | O | | T | 21 | | LRN | 77 | | | | | |
| 15 | O | | | | | | | | 19 | | 'LABELS FOR' | |
| 16 | O | | | | | | | MONTH | 23 | | | |
| 17 | O | | | | | | | | 44 | | 'ISSUES BEGIN ON DAY' | |
| 18 | O | | | | | | | INITIALZ | 47 | | | |
| 19 | O | | | | | | | | 60 | | 'OF THE MONTH' | (14) |
| 20 | O | | T | 11 | | LRN | 77 | | | | | |
| 21 | O | | | | | | | | 34 | | 'AND HAVE AN INCREMENT OF' | |
| 22 | O | | | | | | | INCRMTZ | 37 | | | |
| 23 | O | | | | | | | | 43 | | 'DAYS.' | |
| 24 | O | | T | 31 | | 77 | | | | | | |
| 25 | O | | | | | | | NAME | 25 | | | |
| 26 | O | | | | | | | | 36 | | 'ISSUE' | |
| 27 | O | | | | | | | MONTH | 40 | | | |
| 28 | O | | | | | | | | 41 | | '.' | (15) |
| 29 | O | | | | | | | DAY | 44 | | | |
| 30 | O | | T | 11 | | 77 | | | | | | |
| 31 | O | | | | | | | ADDRESS | 25 | | | |
| 32 | O | | T | 11 | | 77 | | | | | | |
| 33 | O | | | | | | | LOCATE | 26 | | | |

Figure 109.1. Coding for Detail-to-Total Branch Example

PRINTING MAILING LABELS

① Heading Lines

JOHN DOE ISSUE MAR. 03
805 MAIN
ANYTOWN, USA

JOHN DOE ISSUE MAR. 10
805 MAIN
ANYTOWN, USA

JOHN DOE ISSUE MAR. 17
805 MAIN
ANYTOWN, USA

JOHN DOE ISSUE MAR. 24
805 MAIN
ANYTOWN, USA

JOHN DOE ISSUE MAR. 31
805 MAIN
ANYTOWN, USA

⑩ Detail-to-Total
loop output lines

5 LABELS HAVE BEEN PRINTED FOR

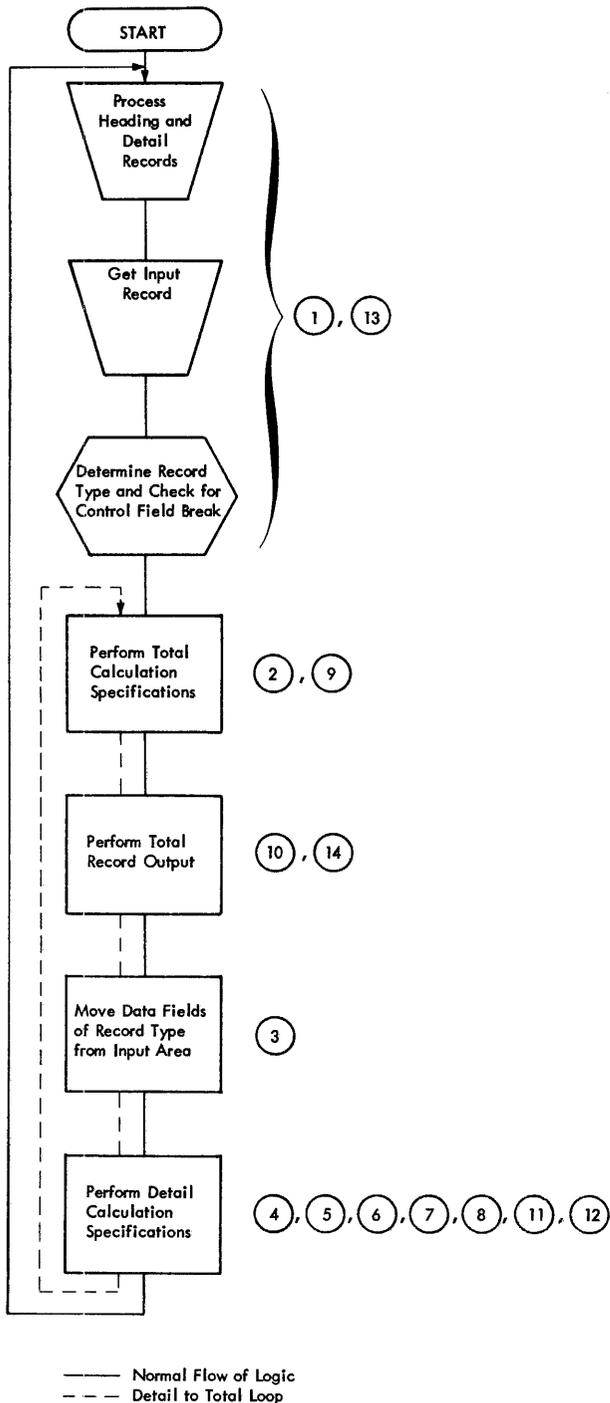
JOHN DOE
805 MAIN
ANYTOWN, USA

⑬ Detail output lines

LABELS FOR MAR. ISSUES BEGIN ON DAY 3 OF THE MONTH
AND HAVE AN INCREMENT OF 7 DAYS.

⑭ Normal total
output lines

Figure 109.2. Output from Detail-to-Total Branch Example



are produced. The following numbered steps refer to circled numbers on Figures 109.1, 109.2, and 109.3.

1. Heading lines are printed, the first input record is read and its type determined.
2. Total calculations are suppressed as they are conditioned on indicator 77, which is on only during a detail-to-total loop in this example.
3. Data is moved from the input area to input fields.
4. Input data is moved from the input fields to new fields and protected by indicator 77. (The move is performed only when indicator 77 is off, which is true only during normal detail time.)
5. Results are formed in the new fields and protected by indicator 77.
6. Indicator 77 is set on, conditioning the detail-to-total branch, and protecting the new fields.
7. A detail time test is made to decide whether to perform the detail-to-total branch or discontinue the loop.
8. When the result of the test is low, indicator 88 is turned on and a detail-to-total branch is performed.
9. Total time calculations are changing one of the fields used in the test performed in step 7. In this manner, the number of times the loop is performed is controlled.
10. Total lines are printed in conjunction with the detail-to-total loop.
11. Step 7 is repeated.
12. When the results for this detail record have been obtained, the settings for indicators 88 and 99 are reversed, indicator 77 is turned off and the detail-to-total loop is discontinued.
13. Detail time output lines are printed and the next input record is read.
14. Normal total lines are printed.

Figure 109.3. Logic Flow for Detail-to-Total Branch Example

This section of the manual is for readers not familiar with disk storage operations. It contains a general description of data organization and retrieval, and defines some of the terms you may encounter in related literature.

INTRODUCTION AND TERMINOLOGY

The distinction between two concepts is important: file organization and file processing.

File Organization is the method of arranging data records on a direct access storage device. A file is organized during the development stages of the application.

File Processing is the method of retrieving data records from the file.

To achieve the most efficient use of the System/360 components, carefully consider the relationship between how a file is organized and how to retrieve records from it. This is particularly important when designing data files for storage in a direct access storage device, such as the IBM 2311.

The method of organization best suited to a particular file of disk records depends upon many factors. These factors must be analyzed for each file in each particular application. Frequently, you can use more than one method of processing on the same file. For example, records within a file might be processed at random during an updating run, and sequentially during a billing run.

LOGICAL FILE VS PHYSICAL UNIT

It is important to distinguish between a logical file and the physical unit used to store the file. A logical file is a group of related data records, such as a payroll file.

A physical unit for storage of data records could be an IBM 2400-series Magnetic Tape Unit, an IBM 2311 Disk Storage Drive, or an IBM 2501 Card Reader.

A logical file may occupy part of a disk storage drive, an entire disk storage drive, or more than one disk storage drive. The location of the logical file in disk storage is defined by its lowest and highest

addresses. This area is the extent area. One logical file can occupy more than one extent area. The extent areas do not have to be adjoining.

DATA FILES

Data files are recorded on such media as: paper, cards, tape, or disk packs. Data files consist of a number of individual records that range from a few records up to thousands or millions of records.

RECORD

A record can be defined as a collection of information consisting of alphameric and/or nonalphameric characters related to a common identifier. The common identifier is known as a record's control field, or key. Usually, one of the fields within a record identifies the record. For example, man number could be the key or identifier for a payroll record.

The size or length of records varies from file to file, and can be from eighteen characters to 4,000 characters.

A single record usually includes one or more logical data fields. A data field is a sequence of one or more characters which is treated as a processing unit of information. An individual data field is normally identified by its location within a record.

The logical structure of records and of fields within records is important in high-speed recording media such as magnetic tape and disk. This logical structure is strongly affected by whether a record is of fixed- or variable-length.

FIXED-LENGTH RECORDS

In fixed-length record files, all records are allocated the same number of character storage positions. Identical data fields are present in every record, whether they are used or not. The control field (key) is usually the first field present in a record.

In many applications, fixed-length records would make inefficient use of file storage space. For example, a fixed record-length of 850 positions would waste storage and processing time, if the average record

length is 230 positions and the minimum length is only 100 positions.

Situations such as this require the development of space-saving techniques based on varying the number of storage positions allocated to data records.

VARIABLE-LENGTH RECORDS

Completely variable-length records are sometimes developed for more efficient use of storage. In this approach, the data portion of the record may be of any length, but the key (control field) size is constant. A record-length character-count field in each record shows the length of a variable-length record.

BLOCKING RECORDS

The length of individual data records varies with type of data and the application that requires such data. The format of a data record is significant to the efficient use of the various storage media available on the System/360. One important element in the design of data records involves blocking and deblocking. Input/output units (storage media) are relatively inefficient when used to record short blocks of information. To increase the efficiency of input/output units, data records are assembled into blocks of records whose sizes are convenient and efficient for processing.

Each physical record on either tape or disk requires inter-record gaps. These gaps are blank areas used to distinguish beginning and ending points of a record. If records are blocked before loading onto a tape or disk, many of these gaps can be eliminated. Variable-length blocks are permitted in System/360, Operating System RPG. The length of a variable-length block is indicated by a block-length character-counter field present in each block. (See Variable-Length Records.)

The operating system handles the blocking and deblocking of records so the user need only determine the most efficient blocking factor for his particular data file and equipment specifications. The system also creates and maintains the block-length and record-length count fields; no programming for these facilities is required by the RPG programmer.

In the operating system, only the input records for indexed-sequentially or sequentially organized files can be blocked.

FILE ORGANIZATION

Data records should be organized and stored to facilitate subsequent processing. The three types of file organization are: sequential, indexed-sequential, direct.

SEQUENTIAL ORGANIZATION

The logical sequence of records in this file depends upon a significant key (control field) appearing in the records. To establish a sequentially organized file, sort and then store the records in key sequence. This allows for records with successively higher or lower keys (control numbers) to have successively higher physical address numbers. Cards and tape files are always organized in this serial manner and usually are considered as one continuous "string" of records.

INDEXED-SEQUENTIAL ORGANIZATION

In this type of file organization also, the sequence of records depends upon a key (control) field. The records are stored sequentially in the file. This variation of file organization differs from sequential organization in two ways:

1. The records may be retrieved from the file sequentially or in a random sequence.
2. Only records with transaction activity need be retrieved.

These differences occur because indexed-sequential organization uses "index tables" which indicate to the program the general location of the records. Thus, the program does not have to "step through" the file, record after record, to locate a specific record.

The index tables (prepared and maintained by the operating system) are analogous to the index card file in a library.

For example, if the library user knows the name of the book or the author he can look in the index card file and find the location of the book in the book files. This might be an address (catalog number) of 426.25. He would then go to the book shelves, and (if it was his first time in the library) start at the first row of the book files and proceed through the rows until he found the shelf that contained 426.25. Usually, each row contains a sign

to indicate the beginning and ending numbers of all books in that particular row. Thus, as he proceeded through the rows, he would compare 426.25 with the numbers posted on each row. Assume that one row was labeled 300.88-550.00. He would then search that row for the shelf that contained the book. The shelves (like the rows) might also contain number ranges to indicate their contents. In this case, he would scan the shelf numbers until he found something like 342.00-440.96. Then he would look at individual book numbers on that shelf until he found 426.25.

The RPG program uses index tables in much the same way to locate records organized in an indexed-sequential file.

DIRECT ORGANIZATION

In direct file organization, the records are generally not stored in the sequence of their keys (control numbers). A randomizing formula converts the record key to a numerical address (physical address) of the storage device. The record is stored at the physical address developed by the randomizing formula. In effect, a file of records will be scattered throughout an entire disk file.

RPG does not provide for creating a file with direct organization. Creation of a file with direct organization must be accomplished by the programmer using the input/output macro facilities of the assembler programming system. During the processing of the object program the user has the ability to exit to a subroutine to perform a randomizing routine upon the input data records. RPG cannot process duplicate records. (Duplicate records occur when two different control fields convert to the same physical address.)

FILE PROCESSING

For the three methods of file organization (sequential, indexed-sequential, and direct) there are three methods of file processing:

1. Sequential processing of sequentially organized files.
2. Sequential processing of indexed-sequentially organized files.
3. Random processing of indexed-sequentially organized files and directly organized files.

SEQUENTIAL PROCESSING (Sequential files)

In sequential processing of a sequentially organized file, every record in a file is examined, and each successive record in the physical file is processed in order. For example, in a card file, the card records are processed in the order that the cards are fed into the system. The 14th card in the file could not be processed until after the 13th card had been processed.

SEQUENTIAL PROCESSING (Indexed-Sequential files)

Sequential processing of an indexed-sequentially organized file has two variations.

1. An entire logical file is processed. For example, the physical unit consists of payroll records in cylinders 0 to 42 and inventory records in cylinders 43 to 99. Only the logical (payroll) file might be processed.
2. Only a segment of a file is processed. For example, a payroll file is to be updated with new pay increases. The payroll file is in sequence by department, and each week the pay raises for various departments become effective. Therefore, on each week's processing, only segments of the payroll file are updated. The updating is accomplished by reading in a card file that contains the limits of the file to be processed. One such card record might indicate that the records for departments 26-41, are to be updated, another the records for departments 76-80, etc.

RANDOM PROCESSING

In random processing, the sequence of processing has no relationship to the sequence in which the data is stored in the file. The data file could be organized in either a direct or an indexed-sequential order. This processing is sometimes called direct.

Indexed-Sequential Files

To find a random record in an Indexed-Sequential file, an index or series of indexes is first scanned to localize the area of search by determining the track that

contains the record. The index is a sequential list of the key records (of the data) with corresponding track addresses. The entire track is then scanned to find the individual record to be processed. This kind of processing is referred to as processing in a random sequence with record keys.

This type of processing is analogous to directing someone to a house location. "The Martin family lives on Harrison Street" (a track address) "and their house number is 4216" (a key).

Direct Files

To find a random record in a direct file, compute the track address by the same randomizing formula used to load the file of records. You can make direct access to the record. Index tables are not required. This kind of processing is called processing in a random sequence and it can be done using keys or record identification (ID). The record identification indicates only the location of the record on the track. For example, the 2nd, 12th, 18th, etc. record on the track. The program makes no comparison of key (control field) data when a record number is provided.

This type of processing can also be compared to directing someone to a house location. "The Martin family lives on Harrison Street", (a track address), "and their house is the 5th house from the beginning of the street." (The 5th is the record identification.)

If random processing is performed with key field only, the user supplies track address and record key field. Starting with this address the program searches the track for the record with the corresponding record key.

FILE PROCESSING IN RPG

The preceding information in this section provided a general introduction to disk storage concepts for the System/360, Operating System. The material that follows is a summary of file processing methods for the RPG program.

RPG generated programs process the following input file organizations:

1. Sequential
2. Indexed-Sequential
3. Direct

RPG will only create sequential output files.

SEQUENTIAL ORGANIZATION

The records on the file will be made available for processing in the same order in which they are located on the medium. The file might be contained in cards, on magnetic tape, or on a DASD. The entire file is processed, beginning with the first record and continuing until the file is depleted.

The end-of-file condition is determined as the last card of the file is read. In the case of DASD, the extent of the file is obtained from the operating system.

The records may be fixed-or variable-length and blocked or unblocked.

INDEXED-SEQUENTIAL ORGANIZATION

An indexed-sequential file can only be on a direct-access-storage device (such as disk).

RPG processes indexed-sequential files in three ways:

1. Sequentially, by processing the entire file.
2. Sequentially, by processing a segment of the file between the given limits.
3. Randomly, by processing records on the file in a random order.

The records may be fixed-length and blocked or unblocked.

Processing the Entire File Sequentially

The generated program obtains the limits of the file from the operating system, and the entire file is processed sequentially by record key in ascending or descending record-key sequence.

Processing Part of the File Sequentially

If only a part of the entire file is to be processed, the generated program must be supplied with both the low and high keys that describe which part of the file will be processed.

An auxiliary file is used to supply these limits. This file is called a Record

Address File (RAF). The RAF does not contain data to be processed. It contains the record keys (in this case the limits) of the data records which will be processed.

The object module obtains the limits to be processed from a record contained in a RAF and then processes all the data between those limits. The object module then reads another record in the RAF, and the procedure is continued until the RAF is depleted.

Processing the File Randomly

An indexed-sequential file may be processed randomly by supplying an RAF or a chaining file. Instead of supplying the limits (as in the case of sequential processing), the RAF or chaining file contains the record key of each record of the file to be processed.

DIRECT ORGANIZATION (RANDOM)

A file with direct organization must reside on a direct access storage device. A

direct file is always processed randomly. Records are retrieved from this file by using a relative track address and either a record key or record identification. The records must be fixed-length and unblocked.

A direct file is processed randomly by supplying a record-address file or a chaining file.

The RPG program or an external subroutine must provide the necessary steps to convert the data fields contained in the RAF or chaining file to the relative track address and record key or record ID to be retrieved.

The format of the relative track address created by the subroutine must be in the form TTR. (Refer to IBM System/360, Operating System, Supervisor and Data Management Services.)

The next two major sections of the manual are Processing Single Input Files and Processing Multiple Input Files. These sections describe disk storage processing in the RPG program. In these sections, processing methods are described in detail including specific examples of methods and of coding specifications.

PROCESSING SINGLE INPUT FILES

In this section, the methods of retrieving records from one input file are discussed. Three types of file organization can be processed: Sequential, Indexed-Sequential, and Direct.

If there is only one input file, it must be designated as the primary file by entering a P in column 16 of the File Description sheet.

SEQUENTIAL FILE

Figure 110 shows the coding on the File Description sheet necessary to process a sequential file. The name of the file is MASTERIN. The records are fixed-length, and the block length is 200. Each record is 200 characters long. A blank in column 32 indicates that the file is a sequential file.

PROCESSING AN INDEXED-SEQUENTIAL FILE BETWEEN LIMITS

If only a part of an indexed-sequential file is to be processed, the object program must be supplied with both the low and high keys that describe which part of the file will be processed. Mode of Processing (column 28) of the File Description sheet must contain L.

The object program obtains the limits to be processed from a record contained in

an RAF and then processes all data between those limits. The object program then reads another record in the RAF, and the procedure is continued until the RAF is depleted.

In Figure 111, the first card of the record-address file shows that the object module is to process from Bell to Dennis. The second card shows that the program is to process from Dixon to Howard. The third card shows that the third part of the file to be processed ranges from Keith to Paige. Thus, the data records that the program processes are contained within these limits. In this example, the record keys are customer names.

The record-address file in this example is nothing more than a file which supplies limits of the file to be processed. Rules for forming record-address files are contained in Creating Record Address Files at the end of this section of the manual.

Figure 112 illustrates the coding required on the File Description and File Extension sheets when limits of an indexed-sequential file are to be processed. In Figure 112, on the File Description sheet DISKIN is the name of the input file. The records are fixed-length, and the block length is 150. Each record is 150 characters long. Limits of the file are to be processed as indicated by the L in column 28. The K in column 31 indicates that the record key will be used to obtain the records from the file.

The record-address file (RAFLIMIT) is also an input file. The R in column 16 indicates that it is an RAF. It is

IBM INTERNATIONAL BUSINESS MACHINES
REPORT PROGRAM GENERATOR
 Date _____
 Program _____
 Programmer _____
 Punching Instruction _____
 Grid _____
 Punched _____

| Line | Form Type | Filename | File Type | File Designation | End of File | Sequence | File Format | Block Length | Record Length | L/R | M/T | E/D/T | Key Field Starting Location |
|------|-----------|----------|-----------|------------------|-------------|----------|-------------|--------------|---------------|-----|-----|-------|-----------------------------|
| 0 1 | F | MASTERIN | P | | | | F | 200 | 200 | | | | |
| 0 2 | F | | | | | | | | | | | | |

Figure 110. Coding a Sequential File

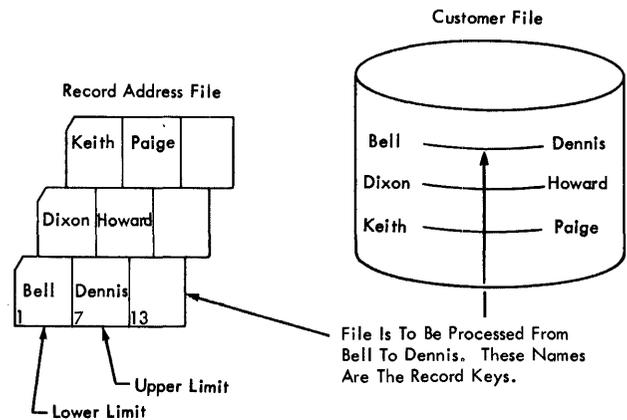


Figure 111. Contents of the Record Address File

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS
 IBM System 360

Form X24-3347-2
 Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Filename | File Type | | | | Mode of Processing | | | | Device | Symbolic Device | Name of Label Exit | Extent Exit for DAM | Comments | |
|------|-----------|------------|-----------|-----------|-----|-----|--------------------------------|---------------------|---------------------------|--------------------|--------|-----------------|--------------------|---------------------|----------|--|
| | | | I/O/U/C | P/S/C/R/T | A/D | F/V | Length of Record Address Field | Record Address Type | Type of File Organization | Overflow Indicator | | | | | | |
| 0 1 | F | DISKIN IP | | | | | | | | | | | | | | |
| 0 2 | F | RAFLIMITIR | | | | | | | | | | | | | | |
| 0 3 | F | | | | | | | | | | | | | | | |

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS
 IBM System 360

Form X24-3348 1
 Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Record Sequence of the Chaining File | | Table Name | Number of Table Entries Per Record | | Table Name (Alternating Table) | Length of Table Entry | | Comments |
|------|-----------|--------------------------------------|-------------|------------|------------------------------------|-----------------------|--------------------------------|-----------------------|-----------------------------|----------|
| | | From Filename | To Filename | | Per Table | Length of Table Entry | | Packed (P) | Decimal Pos. Sequence (A/D) | |
| 0 1 | E | RAFLIMIT | DISKIN | | | | | | | |
| 0 2 | E | | | | | | | | | |

Figure 112. Processing a Part of the Indexed-Sequential File

fixed-length; it has a block length of 80, and each record is 80 characters in length. The length of each record-address field is 6 characters.

Because the File Extension sheet is required to relate the RAF to the DASD file, an E is entered in Extension Code (column 39). On the File Extension sheet the coding illustrates that RAFLIMIT is to provide the addresses for DISKIN.

RANDOM PROCESSING OF AN INDEXED-SEQUENTIAL FILE

An indexed-sequential file may be processed randomly by supplying an RAF. Instead of supplying the limits (as in the case of sequential processing), the RAF contains the record key of each record of the file to be processed.

Direct organization is specified by a D in column 32 of the File Description sheet. The mode of processing is random, and an R in column 28 indicates random processing. If the record key is used to obtain the records, enter a K in column 31. If a record ID is used, enter an I in column 31.

When an RPG program processes an input file of this organization, an RAF must be supplied. The necessary steps must be supplied which convert the data fields contained in the RAF to the relative track address and the record key or record ID to be retrieved. The conversion routine depends, of course, on the way a particular data field can be converted to the track address of the record.

The conversion routine is unique, according to the needs of a particular installation. It may be nothing more than supplying the information without any calculations. To retrieve records from the file, some data fields must be converted to produce the relative track address of the record.

Supplying Data to be Converted

One entry in the RAF is supplied for each record to be retrieved. The data supplied by the RAF should be such that the relative track address and the record key or record identification can be derived. Because the RAF is not described on the input specifications, the entries in the RAF will be made available consecutively in a field called CONTD. This field is always alphameric, and it has the record length as specified in columns 29-30 of the File Description Specifications sheet.

NOTE: The field CONTD is always predefined by RPG as an RPG Label (RLABL)

Associating a Particular Conversion Routine with RAF

The File Extension specifications describing the RAF contain a field name in columns 27-32. This name is used as Factor 1 on the calculation specifications with the first specification of the conversion routine.

Methods for Specifying a Conversion Routine

Two methods of specifying a conversion routine to RPG are available:

1. The conversion routine is written on the Calculation Specifications sheet, or

2. The conversion routine can be written as an independent routine that must be combined with the generated object program.

The first specification of the conversion routine defines the type of conversion by means of the operation codes RPGCV or EXTCV. If the conversion routine is coded on the Calculation Specifications sheet, RPGCV is specified. If the conversion routine is external to the RPG language, EXTCV is specified.

External Conversion Routine

If the conversion routine is external to the RPG language, Factor 2 must contain the name (or label) of the user-supplied routine. This external conversion routine must follow the same conventions which are specified for the EXIT routine. See Use of Registers in this publication.

The Result Field must contain the name (or label) of the field, in the subroutine, which will contain the track address of the record to be retrieved. This is the result of the conversion.

Defining the Key or ID Field

Regardless of whether an RPG or an external conversion routine is specified, if record key retrieval is used (rather than record ID) the next calculation specification entry must define the field that will contain the record key.

The operation code is KEYCV, (a K in column 31 of the File Description sheet). The Result Field must contain the name (or label) of the field which will contain the actual record key used to locate the record.

If record ID retrieval is used, an operation code entry is not required. However, the File Description Sheet must contain an I in column 31.

In either key or ID retrieval, the result field of EXTCV or RPGCV must be a relative track address in the form TTR.

Conversion Operation Codes

The operation codes which follow are used in conjunction with conversion routines. If there are several conversion routines in the program, these codes are repeated.

RPGCV. This operation code indicates that the conversion routine is coded on the RPG calculation sheet.

ERPGC. This entry terminates the RPG conversion step entries that have been coded on the calculation sheet.

EXTCV. This entry indicates that the conversion routine is supplied by the user in a separate subroutine which is external to the RPG language.

KEYCV. This entry declares that the field specified in Result Field will contain the record key to be used with the relative track address. This entry must follow the RPGCV or EXTCV entry.

In the example shown in Figures 115 and 116 the data supplied in the RAF is both the record key and the data to be converted. The conversion routine shows how this field is then separated into two elements.

The field CONTD contains the 14-character field from the RAF. The first 9 characters contain the customer name which is used as the key. The remaining 5 characters contain a code for calculating the track address of the customer's record. Line 04 of the Calculation Specifications sheet (Figure 116) moves the first part of CONTD to the key field (KEYFLD) and line 05 moves the remaining part to the work field (WORKFD). The alternating table on TABFIL is used to convert this 5-character code to the 3-character relative track address that is moved to the field TRKADR.

Figure 117 shows how the calculation specifications would be coded if the conversion routine were external to the RPG language.

CREATING RECORD ADDRESS FILES (RAF)

General Information

A record-address file is one of the ways by which the necessary information to retrieve records from nonsequential files is supplied to the RPG program. Two types of record-address files may be used:

1. For random processing of a file with indexed-sequential organization or of a file with direct organization.
2. For sequential processing between limits, of a file with indexed-sequential organization.

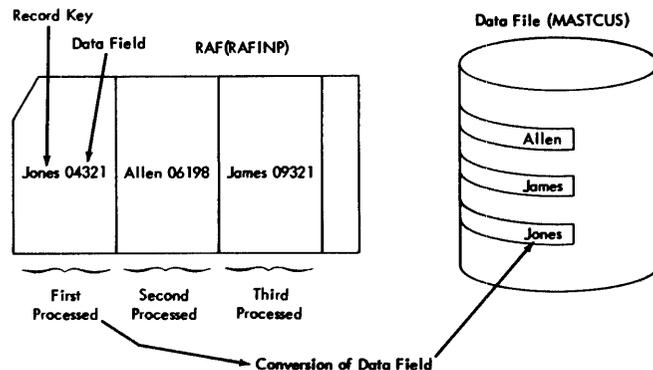


Figure 115. Conversion of a Record Address File

Only one RAF may be specified for an RPG program. An RAF is processed sequentially, and it must be on a file with sequential organization. An RAF is described on the File Description sheet and the File Extension sheet, but it is not described on the Input Specifications sheet.

Random Processing of Indexed-Sequential or Direct Organization

These rules must be followed when creating an RAF for random processing:

1. For indexed-sequential organization, the record-address field contains the record key.
For a direct organization, each entry in the RAF must consist of a field to be converted to the track address and to either the record key or the record ID.
2. The record addresses must begin in position 1 of the record and continue without blank spaces between the record-address fields.
3. The length of the field must be the same for all records. The numeric fields must always be unpacked.
4. The number of field entries in a record may vary. A blank field, which is equal in length to the record-address field, will cause the RPG program to read the next record in the RAF.

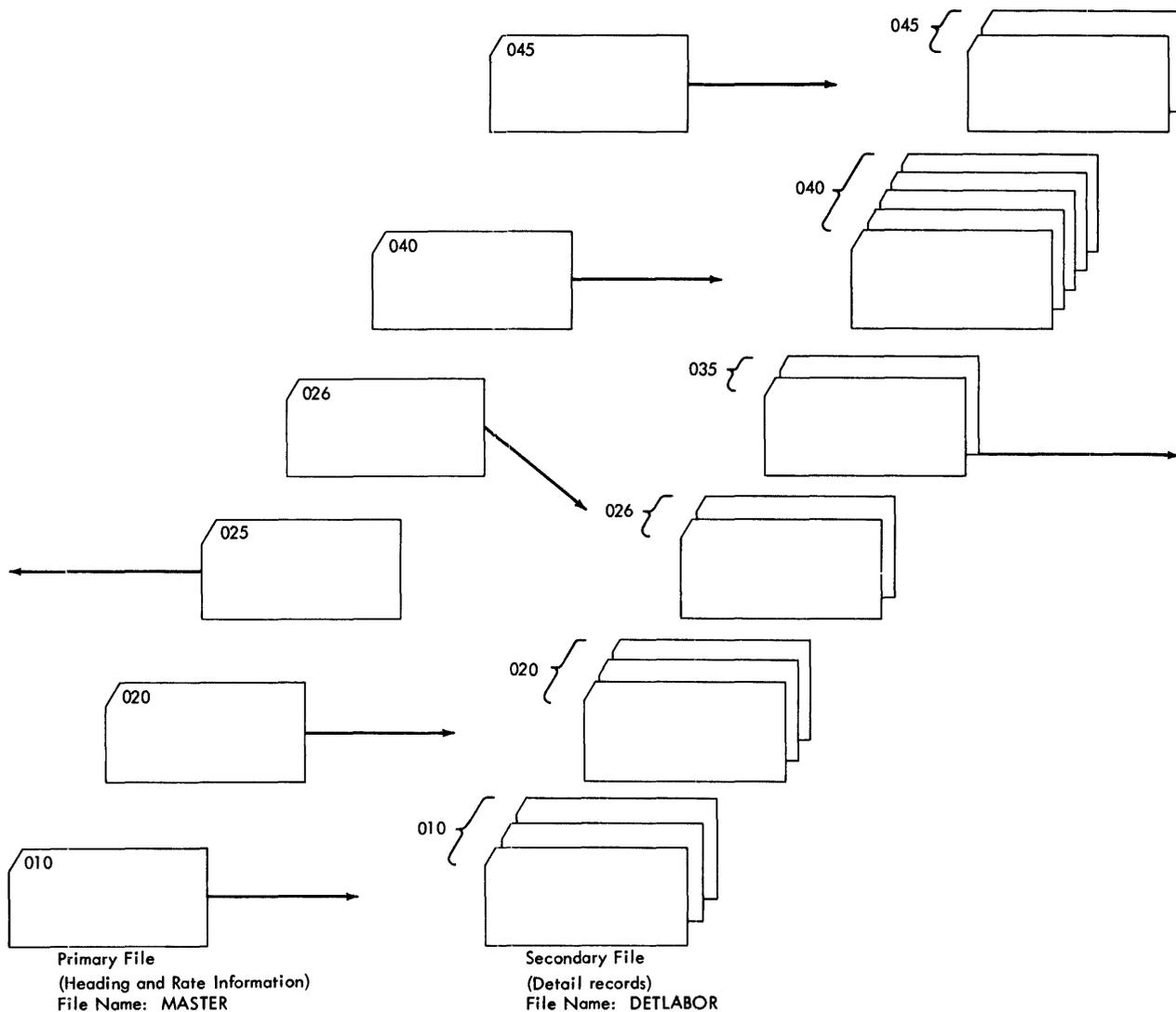


Figure 119. Two Input Files with Matching Records

information. The secondary file contains detailed information which supplements the primary file.

The input specifications required to match these records are shown in Figure 120.

The specifications M1 and M2 cause each detail record to be compared against the primary file's record that has just been read. The fields DIVSON and DETDIV in both files are identified by M2; the field department (DEPT and DETDEP) in both files is identified by M1.

Matching Record Indicator

The matching field entries of M1 and M2 (and also M3) have an associated internal indicator MR (Matching Record). This indicator, which is similar to a resulting indicator, is used to control functions specified on the Calculation and Output-Format Specifications sheets.

The MR Indicator is turned on when a record of a secondary file matches a record of the primary file. It remains on during

field is matched, the designations of M3, M2, and M1 must be assigned in the same sequence in which the fields are to be arranged for matching. M3 is assigned to the highest-order field, M2 to the next lower-order field, and M1 to the lowest-order field.

For example, in Figure 120, only two fields are matched. M1 is assigned to DEPT which is the low-order matching field, and M2 is assigned DIVSON, which is the high-order matching field.

The position in the record of the fields to be matched does not have to be the same in both files. For example, in Figure 120, the field DETDIV is located in positions 1-4 of the detail records, and the field DIVSON is located in positions 28-31 of the rate-header records.

Order of Processing Matched Records

Figure 121 illustrates a primary and a secondary file. The records in the two files will be processed according to four possibilities:

1. Whenever there is a matching primary record.
2. Whenever there is a matching secondary record.
3. Whenever there is an unmatched primary record.
4. Whenever there is an unmatched secondary record.

In Figure 121, indicator 01 is turned on whenever there is a primary record. Indicator 02 is turned on whenever there is a secondary record. Thus,

1. A matching primary record will be coded 01 and MR.
2. A matching secondary record will be coded 02 and MR.
3. An unmatched primary record will be coded 01 and NMR.
4. An unmatched secondary record will be coded 02 and NMR.

The order in which the primary file and the secondary file are processed is shown below. Sample Program Two uses the matching field specifications.

| <u>Primary File</u> | | <u>Secondary File</u> | |
|-------------------------------------|------------------|-------------------------------------|------------------|
| <u>Matching Field In The Record</u> | <u>Processed</u> | <u>Matching Field In The Record</u> | <u>Processed</u> |
| 001 | 1st | 001 | 3rd |
| 001 | 2nd | 001 | 4th |
| 002 | 5th | 002 | 6th |
| 004 | 9th | 003 | 7th |
| 004 | 10th | 003 | 8th |
| 006 | 12th | 005 | 11th |

If more than one secondary file is specified, the secondary files will be processed in the sequence they are specified in the Input Specifications sheet.

NOTE: Input Specifications for each file must be in the same sequence as specified on the File Description Specifications sheet.

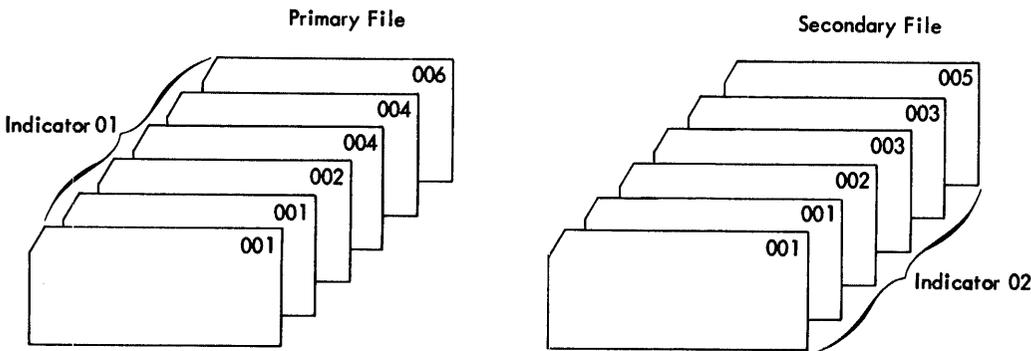


Figure 121. Order of Processing Matched Records

**RANDOMLY PROCESSING MULTIPLE INPUT FILES
(CHAINING)**

To understand chaining, assume that an input file, as shown in Figure 122, contains information about transactions made with several customers. The card file contains the customer's number, but it does not contain his name, address, or balance. Another file called MASTCUS (Master Customer File) contains information about each customer. The RPG object program has the ability to use the second file, when preparing a customer report.

The master customer file has indexed-sequential organization, and it is to be processed randomly. The record key in each disk record contains the customer's numbers. The field in the transaction file, which contains the customer's number, can be used to chain the files together. The object program takes the customer's number and locates the record with the same record key. The additional information, such as the customer's name, address, balance, etc., is associated with each record key, and it is immediately available for processing.

The field which links or chains a record of one file to a record in another file is

called a chaining field. The transaction file (CARDIN) is called the chaining file. The master customer file (MASTCUS) is the chained file because it is linked to the transaction file.

Up to nine chaining fields may be specified. The chaining fields can be located in one or more files. The chaining fields are designated by entering C1 through C9 in columns 61-62 of the Input Specifications sheet.

NOTE: There is no specific relationship between levels C1-C9 other than specifying the nine possibilities for chaining fields.

Chaining Example

Figure 123 illustrates coding used for chaining the transaction file CARDIN to the customer file MASTCUS.

File Description Specifications

On the File Description sheet, the card file is considered the primary file. When CARDIN is depleted, the program goes to the end-of-job routine (E in column 17).

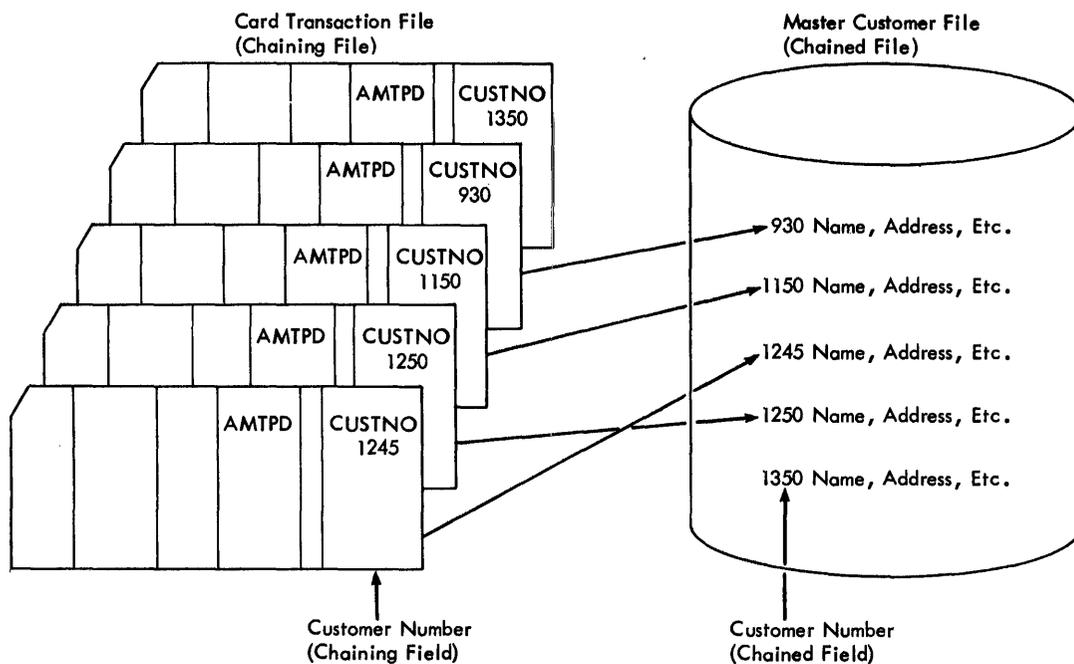


Figure 122. Using Chaining to Process an Indexed-Sequential File

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS
 IBM System/360 Form X24-3347-2 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

Punching Instruction: Graphic Punch

Page 1 2
 Program Identification: 75 76 77 78 79 80

| Line | Form Type | Filename | File Type | | Mode of Processing | | Device | Symbolic Device | Name of Label Exit | Extent Exit for DAM | Comments |
|------|-----------|----------|------------------|-------------|--------------------|---------------|---------|-----------------|--------------------|---------------------|----------|
| | | | File Designation | End of File | Length of Record | Address Field | | | | | |
| 0 1 | F | CARDIN | IC | F | 80 | 80 | EREAD40 | | | | |
| 0 2 | F | MASTCUS | IC | F | 1000 | 2000 | KI | 1 | DISK11 | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS
 IBM System/360 Form X24-3348-1 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

Punching Instruction: Graphic Punch

Page 1 2
 Program Identification: 75 76 77 78 79 80

| Line | Form Type | Record Sequence of the Chaining File | | Table Name | Number of Table Entries Per Record | | Table Name (Aligning Table) | Length of Table Entry | | Comments |
|------|-----------|--------------------------------------|-------------|------------|------------------------------------|-----------------------|-----------------------------|-----------------------|-------------|----------|
| | | From Filename | To Filename | | Per Table | Length of Table Entry | | Table Entry | Table Entry | |
| 0 1 | F | AAICARDIN | MASTCUS | | | | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS
 IBM System 360 Form X24-3350-1 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

Punching Instruction: Graphic Punch

Page 1 2
 Program Identification: 75 76 77 78 79 80

| Line | Form Type | Filename | Sequence Number (1-10) | Repeating Indicator | Record Identification Codes | | | Field Location | | Field Name | Control Level (1-17) | Matching Field, or Chaining Field | Field-Record Relation | Field Indicators | | | Sterling Sign Position |
|------|-----------|----------|------------------------|---------------------|-----------------------------|----------|----------|----------------|----|------------|----------------------|-----------------------------------|-----------------------|------------------|-------|---------------|------------------------|
| | | | | | Position | Position | Position | From | To | | | | | Plus | Minus | Zero or Blank | |
| 0 1 | F | CARDIN | AA | 01 | 80 | CX | | | | | | | | | | | |
| 0 2 | F | | | | | | | 75 | 80 | CUSTNO | C1 | | | | | | |
| 0 3 | F | | | | | | | | | | | | | | | | |
| 0 4 | F | | | | | | | | | | | | | | | | |
| 0 5 | F | MASTCUS | AA | 02 | 2000 | D1 | | | | | | | | | | | |
| 0 6 | F | | | | | | | 1 | 6 | CODE | | | | | | | |
| 0 7 | F | | | | | | | 7 | 20 | NAME2 | | | | | | | |
| 0 8 | F | | | | | | | 21 | 30 | BALANC | | | | | | | |
| 0 9 | F | | | | | | | 31 | 36 | CUSNO1 | | | | | | | |
| 1 0 | F | | | | | | | 37 | 50 | ADDRS | | | | | | | |
| 1 1 | F | | | | | | | 51 | 60 | CITY | | | | | | | |
| 1 2 | F | | | | | | | 61 | 65 | STATE | | | | | | | |
| 1 3 | F | | | | | | | 66 | 80 | SHTPER | | | | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS
 IBM System 360 Form X24-3351-1 Printed in U.S.A.

Date _____
 Program _____
 Programmer _____

Punching Instruction: Graphic Punch

Page 1 2
 Program Identification: 75 76 77 78 79 80

| Line | Form Type | Indicators | | | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Resulting Indicators | Comments |
|------|-----------|------------|-----|-----|----------|-----------|----------|--------------|--------------|----------------------|----------|
| | | For | And | Not | | | | | | | |
| 0 1 | C | 01 | 02 | | BALANC | SUB | AMTPD | BALANC | | | |
| 0 2 | C | 01 | N02 | | | SETON | | | | HI | |

Figure 123. Specifying Chaining

File Extension Specifications

The File Extension sheet contains the record sequence of the CARDIN file and the number of the chaining field.

Input Specifications

On the Input Specifications sheet, the two files are defined. CUSTNO is the field which chains the files.

Calculation Specifications

When chaining is used, the following situation may occur. A chaining record (for example, JONES) has no corresponding chained record. (JONES is not present in the chained file.) Such a situation may require special action by the program.

Indicators 01 and 02 are both on when a chaining record has a corresponding chained record. In this case 01 represents the chaining record, and 02 represents the chained record. When 01 and 02 are both on, AMPTD is subtracted from BALANC. However, if there is no corresponding record in the chained file (01N02), halt indicator H1 is turned on. Thus, the possibility of not having a corresponding record in the chained file is accounted for, and processing will terminate when this situation occurs.

NOTE: This example illustrates the only time that two resulting indicators (representing two different record types) can be on at the same time.

Additional Uses of Chaining

Figures 124 and 125 show two additional uses of chaining. In Figure 124, the card file contains two fields which are both used as chaining fields (salesman number and customer number). The field containing salesman number is chained to a file that is organized by salesman number. The field containing the customer's number is used to chain to the customer file.

In Figure 125, the card file is chained to the customer file. Within each customer record is a field which may be used to chain to another file. In this case, each record in the customer file contains an account number. The account number is used to chain to the account file.

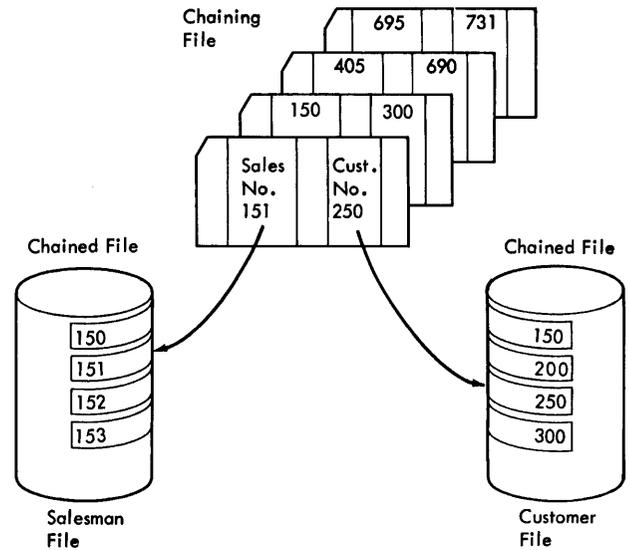


Figure 124. Chaining to Two Files

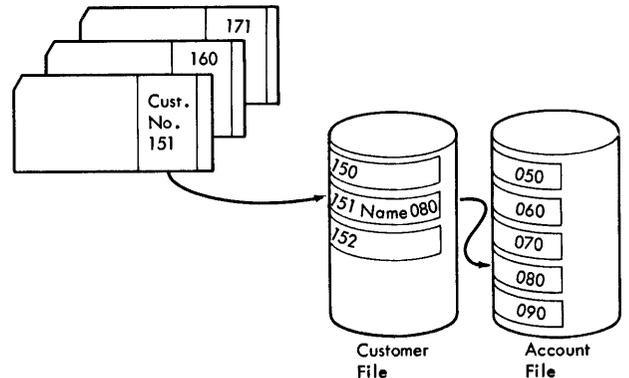


Figure 125. Using a Chained File as a Chaining File

Split Chaining Fields

Several fields that are not in adjoining positions in an input record can be specified as one chaining field. The fields are specified with the same chaining code (C1 for example) on the Input Specifications sheet, and the fields are then used as one chaining field. The fields are placed in the same sequence as they are defined on the Input Specifications sheet.

The first field defined on the input sheet is placed in the leftmost position, and the last field is placed in the rightmost position.

CONVERSION OF CHAINING FIELDS FOR DIRECT FILE ORGANIZATION

The data to be converted will be located in the fields designated as the chaining fields (Figure 126). The data field must be such that the record key or record ID can be derived from the conversion. The rules for conversion of a chaining field are the same as those for an RAF.

Figure 127 illustrates the conversion routine that converts the data contained in a chaining field (Figure 126) to the relative track address and record key of the record.

In this example, the conversion routine is coded on the Calculation Specifications sheet. On the File Description sheet, the two input files are described. TRANSREC is the primary file. MASTCUS is the DASD file that has direct organization and is

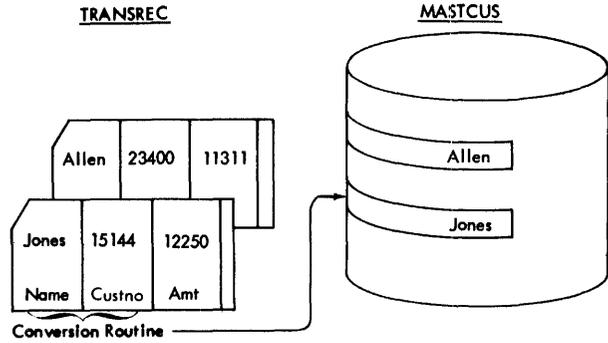


Figure 126. Specifying a Random Disk Address by Converting an Input Record (Chaining File)

processed randomly. Since in direct organization the key is separate from the data record, no entry is made in the Key Field Starting Location (columns 35-38). The conversion routine supplies the track address and the record key.

On the File Extension sheet, the record

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3347-2
REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS Printed in U. S. A.
 IBM System 360

Date _____
 Program _____
 Programmer _____

| Line | Form Type | Filename | File Type | | Mode of Processing | | Device | Symbolic Device | Name of Label Exit | Extent Exit for DAM | Comments |
|------|-----------|----------|------------------|-------------|--------------------------------|---------------------|---------|-----------------|--------------------|---------------------|----------|
| | | | File Designation | End of File | Length of Record Address Field | Record Address Type | | | | | |
| 0 1 | F | TRANSREC | P | E | 80 | 80 | EREAD01 | | | | |
| 0 2 | F | MASTCUS | IC | F | 150 | 150R | DISK11 | | S | | |
| 0 3 | F | | | | | | | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3348-1
REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS Printed in U. S. A.
 IBM System 360

Date _____
 Program _____
 Programmer _____

| Line | Form Type | Record Sequence of the Chaining File | | Table Name | Number of Table Entries Per Record | | Table Name (Alternating Table) | Length of Table Entry | Comments |
|------|-----------|--------------------------------------|-------------|------------|------------------------------------|----------------------------------|--------------------------------|-----------------------|----------|
| | | From Filename | To Filename | | Per Table | Length of Table Entry | | | |
| 0 1 | E | AAC1 | TRANSREC | MASTCUS | CONVER | REFERENCES THE CALCULATION SHEET | | | |
| 0 2 | E | | | | | | | | |
| 0 3 | E | | | | | | | | |

Figure 127. Conversion of a Chaining Field (Part 1 of 2)

SUMMARY OF MULTIPLE FILE PROCESSING

RPG processes multiple files two ways:

1. Sequentially — by using the matching record technique
2. Randomly — by using the chaining technique

Figure 128 shows the processing possibilities for files which have Sequential, Indexed-Sequential, or Direct Organization.

The numbers 1, 2, and 3 refer to the major subjects listed below. The letters A, B, and C refer to the subgroups.

Files with Sequential Organization

1. A file with sequential organization is processed sequentially and controls the processing of records in:
 - A. Another Sequential Organization. Both files are processed sequentially, using matching records to govern processing.
 - B. An Indexed-Sequential Organization. If the file is processed sequentially, the matching-record technique is used to control processing of the indexed-sequential file.

If the file is processed randomly, chaining fields in the sequential file specify which records in the indexed-sequential file are to be processed.
 - C. A Direct Organization. A direct organization is processed randomly, under control of the sequential file. The sequential file contains chaining fields which are converted to the relative track addresses of the records on the direct file.

Files with Indexed-Sequential Organization

2. A file with Indexed-Sequential Organization may be processed sequentially or randomly, and it controls processing of records in:
 - A. A Sequential Organization. The records in both files will be processed sequentially, using matching records to control processing.

- B. Another Indexed-Sequential Organization. If the file is processed sequentially, matching records are used to control processing. (Both files are processed sequentially.)

If the file is processed randomly, chaining fields are used to control processing.
- C. A Direct Organization. Chaining fields in the indexed file are converted to supply the record locations in the direct file which is processed. The direct file is processed randomly.

Files with Direct Organization

3. A file with direct organization is processed randomly and controls processing of records in:
 - B. An Indexed-Sequential Organization. The indexed-file is processed randomly, and chaining fields in the direct file control processing of the indexed file.

In both cases, the direct organization is processed randomly.
 - C. Another Direct Organization. Chaining fields contained within the direct file are converted to provide the location of the records in another direct organization. The records in both files are processed randomly.

Updating a DASD File

An RPG program may perform update processing of a DASD file. The file may be of sequential, index-sequential, or direct organization. The fields of records contained in the file may be changed, however, the size of the records may not be changed. It is not possible to add new records to a DASD file or to delete old records using RPG. Only records existing in the file may be processed. When an update file (U in Column 15 on the File Description sheet) is processed, only the fields to be updated must be entered on the Output-Format Specifications sheet. Although the entire record is to be retained, only the affected fields are entered on the output sheet.

| | | A | B | C |
|------|-----------------------------------|------------------------------------|--|------------------------------|
| From | To | Sequential Organization | Indexed-Sequential Organization (DASD) | Direct Organization (DASD) |
| 1 | Sequential Organization | Sequential Processing (Matching) | Sequential Processing (Matching) or Random Processing (Chaining) | Random Processing (Chaining) |
| 2 | Indexed-Sequential Organization * | Sequential Processing (Matching) □ | Sequential Processing (Matching) □ or Random Processing (Chaining) | Random Processing (Chaining) |
| 3 | Direct (DASD) Organization + | -- | Random Processing (Chaining) | Random Processing (Chaining) |

- * A Record Address File may be used to supply the limits (in the case of sequential processing) or the actual Record Keys (in the case of random processing).
- The From File must be specified as sequential.
- + A Record Address File is converted to supply the record locations on the DASD.

Figure 128. Processing Multiple Input Files

RPG JOB PROCESSING

The IBM System/360 Operating System (the operating system) consists of a control program and processing programs. The control program supervises execution of all processing programs, such as the RPG compiler, and all problem programs, such as an RPG program. Therefore, to execute an RPG program the programmer must first communicate with the operating system. The medium of communication between the programmer and the operating system is the job control language.

Job control language statements define two units of work to the operating system: the job and the job step. A job consists of executing one or more job steps. For example, three job steps are involved to compile, link edit and execute an RPG program.

1. Translate the source program into an object module by executing the RPG compiler component of the operating system.
2. Process the object module to produce a load module by executing the linkage editor component of the operating system.
3. Execute the compiled and link edited load module.

JOB CONTROL LANGUAGE

The RPG programmer uses the job control statements shown in Table 4 to compile, link edit, and execute programs.

These statements are discussed in this section as they are used to specify RPG job processing. A detailed explanation of each statement is given in IBM System/360, Operating System, Job Control Language.

Table 4. Job Control Statements

| Statement | Function |
|-----------|---|
| JOB | Indicates the beginning of a new job and describes that job |
| EXEC | Indicates a job step and describes that job step; indicates the cataloged procedure or load module to be executed |
| DD | Describes data sets, and controls device and volume assignment |
| delimiter | Separates data sets in the input stream from control statements; it appears after each data set in the input stream |

COMPILER PROCESSING

The names for DD statements (ddnames) relate I/O statements in the compiler with data sets used by the compiler. These ddnames must be used for the compiler. When the system is generated, names for I/O device classes are also established and must be used by the programmer.

Compiler Name

The program name for the RPG compiler is IESRPG. If the compiler is to be executed without using the supplied cataloged procedures in a job step (see Using Cataloged Procedures), the EXEC statement parameter, PGM = IESRPG, must be used.

Compiler ddnames

The compiler can use 7 data sets. To establish communication between the compiler and the programmer, each data set is assigned a specific ddname. Each data set has a specific function and device requirement. Table 5 lists the ddnames, functions, and device requirements for the data sets.

Table 5. Compiler ddnames

| ddname | FUNCTION | DEVICE REQUIREMENTS |
|----------|---|---|
| SYSIN | reading the source program | <ul style="list-style-type: none"> ● card reader ● intermediate storage |
| SYSPRINT | writing the storage map, listing, and messages | <ul style="list-style-type: none"> ● printer ● intermediate storage |
| SYSPUNCH | output data set for the object module deck | <ul style="list-style-type: none"> ● card punch ● intermediate storage |
| SYSUT1 | work data set needed by the compiler during compilation | <ul style="list-style-type: none"> ● direct-access ● magnetic tape |
| SYSUT2 | work data set needed by the compiler during compilation | <ul style="list-style-type: none"> ● direct-access ● magnetic tape |
| SYSUT3 | work data set needed by the compiler during compilation | <ul style="list-style-type: none"> ● direct-access ● magnetic tape |
| SYSGO | output data set for the object module used as input to the linkage editor | <ul style="list-style-type: none"> ● direct-access ● magnetic tape |

To compile an RPG source program, five of these data sets are necessary: SYSIN, SYSPRINT, SYSUT1, SYSUT2 and SYSUT3, along with the direct-access volume(s) that contains the operating system. With these five data sets, only a listing is generated by the compiler. If an object module is to be punched, a SYSPUNCH DD statement must be supplied. If an object module is to be passed to the linkage editor, a SYSGO DD statement must be supplied.

For the DD statement SYSIN or SYSPRINT or SYSPUNCH an intermediate storage device may be specified instead of the card reader or printer or card punch. The intermediate storage device can be magnetic tape or a direct access device.

If an intermediate device is specified for SYSIN, the compiler assumes that the source module deck was placed on intermediate storage by a previous job or job step. If an intermediate device is specified for SYSPRINT, the listing, and error/warning messages are written on that device; a new job or job step can print the contents of the data set. When the SYS-PRINT data set is written on an intermediate storage device, carriage control characters are placed in the records.

NOTE: SYSUT1 and SYSUT2 must not be assigned to a 2321 data cell. When using split cylinders, SYSUT1 and SYSUT2 must be assigned an equal number of track per cylinder.

Compiler Device Classes

Names for input/output device classes used for compilation are also specified by the operating system when the system is generated. The class names, functions, and types of devices are shown in Table 6.

The data sets used by the compiler must be assigned to the device classes listed in Table 7.

Table 6. Device Class Names

| CLASS NAME | CLASS FUNCTIONS | DEVICE TYPE |
|------------|---|------------------------------------|
| SYSSQ | writing, reading, backspacing (sequential) | ● magnetic tape ● direct-access |
| SYSDA | writing, reading, backspacing, updating records in place (direct) | ● direct-access |
| SYSCP | punching cards | ● card punch |
| A | SYSOUT output | ● printer ● magnetic tape |

Table 7. Correspondence Between Compiler ddnames and Device Classes

| ddname | Possible Device Classes |
|----------|--|
| SYSIN | SYSSQ, or the input stream device (specified by DD * or DD DATA) |
| SYSPRINT | A,SYSSQ |
| SYSPUNCH | SYSCP |
| SYSUT1 | SYSSQ,SYSDA |
| SYSUT2 | SYSSQ,SYSDA |
| SYSUT3 | SYSSQ,SYSDA |
| SYSGO | SYSSQ,SYSDA |

Compiler Options

Options are passed to the compiler through the PARM parameter in the EXEC statement.

PARM = { DECK } { LOAD } { LIST } { SEQN }
 { NODECK } { NOLOAD } { NOLIST } { NOSEQN }

The programmer specifies the options which are defined as,

DECK The object module is placed on the device specified in the SYSPUNCH DD statement, (usually the card punch).

LOAD The object module is placed on the device specified in the SYSGO DD statement, (usually intermediate storage).

LIST An output listing is written on the device specified in the SYSPRINT DD statement, (see Compiler Output).

SEQN The source deck is checked to ensure that the combined page/line field value in each specification is greater than the combined page/line field value of the previous specification.

The prefix NO is used with any of these options to specify that the option is not wanted. If any entry for any of the three options is omitted from the PARM parameter, the corresponding underlined entry (NODECK, LOAD, LIST, or NOSEQN) is assumed for that option. If contradictory options are entered (e.g., LIST, NOLIST) the entry without the underline (DECK, NOLOAD, NOLIST, or SEQN) is used for that option.

The DECK option specifies that the compiler output (i.e., the object module) is written on the data set specified by the SYSPUNCH DD statement. NODECK specifies that no object module is written. A description of the deck is given in Compiler Output.

The LOAD option indicates that the object module is written on the data set specified by the SYSGO DD statement. This option must be used if a cataloged procedure to compile, link edit, and execute is used.

The NOLOAD option indicates that the object module is not written on a sequential data set. This option must not be used if a cataloged procedure to compile, link edit, and execute is used. If NOLOAD and DECK are specified, the resulting object deck may be used as input to the linkage editor.

If the LOAD and DECK options are specified, the object module is written on the two data sets, indicated by the SYSGO and SYSPUNCH DD statements.

The cataloged procedures assume the options of NODECK, LOAD, LIST, NOSEQN. However, the programmer can override any or all of the assumed options as described in Overriding Cataloged Procedures.

Compiler Output

The RPG compiler can generate a listing of the source specifications, diagnostic messages and a memory map showing the names and addresses of object program routines. The compiler can also produce an object module card deck (Figure 129).

The output listing (see Sample Program One) provides:

1. Listing of each specification with related statement number and error notes.
2. Resulting Indicator Table showing:
 - a. Names of all RPG processor and user defined resulting indicators
 - b. Address (6 places) of each defined resulting indicator
3. Field Name Table containing:
 - a. Names of all fields
 - b. Address (6 places) of each field. ENTRY or EXTERN type field names are denoted by ENTRY or EXTRN
4. Literal Table containing:
 - a. All literals and edit words
 - b. Address (6 places) of each literal or edit word

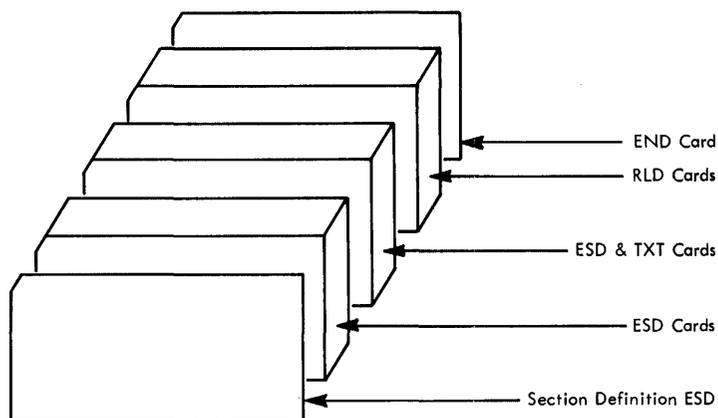


Figure 129. RPG Output Object Module Card Deck

5. Diagnostic listing of erroneous entries showing:
 - a. Statement number
 - b. Name of the erroneous field or resulting indicator
 - c. Appropriate error note
6. Further specification diagnostics
 - a. Statement number
 - b. Appropriate error note
7. Diagnostic messages. Refer to Appendix G.
8. Memory Map listing names and addresses of RPG object program routines.
9. The length (in hexadecimal) of the compiled program not including the IOCS modules or any user subroutines.
10. END OF COMPILATION message.

Compiler Return Codes

Table 8 shows the return codes issued by the RPG compiler for use with the COND= parameter of JOB or EXEC statements.

LINKAGE EDITOR PROCESSING

The linkage editor processes RPG object modules, resolves any references to subprograms, and constructs a load module.

Table 8. Compiler Return Codes

| Return Code | Explanation |
|-------------|--|
| 0 | no errors detected |
| 4 | minor errors detected; successful program execution is probable |
| 8 | errors detected; unsuccessful program execution is possible |
| 12 | serious errors detected; unsuccessful program execution is probable |
| 16 | critical errors detected; normal execution is impossible |
| 20 | unrecoverable I/O error occurred during compilation; compilation terminated |
| 24 | specified program interruption exit. For details refer to <u>IBM System/360, Operating System, Control Program Services.</u> |

Note: the COND= parameter is explained in the Job Control Language publication.

To communicate with the linkage editor, the programmer supplies an EXEC statement and DD statements that define all required data sets; he may also supply linkage editor control statements.

Linkage Editor Name

The program name for the linkage editor is IEWL. If the linkage editor is executed without using cataloged procedures in a job step, the EXEC statement parameter, PGM=IEWL, must be used.

Linkage Editor Input and Output

The modules that are processed by the linkage editor are contained in data sets as

1. Primary input data set (principal input)
2. Call library (for automatic library call)
3. Additional primary input or call library data sets

The primary input data set is required for all linkage editor job steps. The call library is defined only if the automatic

library call function is used. (Refer to IBM System/360, Operating System, Linkage Editor.) Additional sequential data sets or partitioned data sets are defined only as required.

The primary input is a sequential data set that contains object modules and linkage editor control statements. In an RPG compile, link and execute job the primary input data set contains the object modules produced by the job steps. The primary input can be a chain of sequential data sets. A library member can be specified on a DD statement to be processed as a sequential data set. For details refer to IBM System/360, Operating System, Job Control Language. The primary input data set must be specified by the ddname SYSLIN.

The linkage editor can accept input from other than the primary input source. Additional input sources such as user subroutines can be specified by the INCLUDE statement or automatic library call. For details of linkage editor control statements refer to IBM System/360, Operating System, Linkage Editor. Variations to incorporate user subroutines are discussed in Executing RPG — Input Stream Variations.

The output of the linkage editor is always placed in a PDS. Error messages and optional diagnostic messages are written on an intermediate storage device or a printer. In addition, a work data set is required by the linkage editor to do its processing.

Linkage Editor ddnames and Device Classes

The programmer communicates data set information to the linkage editor through DD statements identified by specific ddnames (similar to the ddnames used by the compiler). The ddnames, functions, and requirements for data sets are shown in Table 9.

Any data sets specified by SYSLIB or SYSLMOD must be partitioned data sets. (Additional inputs are partitioned data sets or sequential data sets.) The ddname for the DD statement that retrieves any additional libraries is written in INCLUDE and LIBRARY statements and is not fixed by the linkage editor.

The device classes used by the compiler (see Table 6) must also be used with the linkage editor. The data sets used by linkage editor may be assigned to the device classes listed in Table 10.

Table 9. Linkage Editor ddnames

| ddname | FUNCTION | DEVICE REQUIREMENTS |
|----------------|---|---|
| SYSLIN | Primary input data, normally the output of the compiler | <ul style="list-style-type: none"> ● direct access ● magnetic tape ● card reader |
| SYSLIB | automatic call library | <ul style="list-style-type: none"> ● direct access |
| SYSUTI | work data set | <ul style="list-style-type: none"> ● direct access |
| SYSPRINT | diagnostic messages | <ul style="list-style-type: none"> ● printer ● intermediate storage device |
| SYSLMOD | output data set for the load module | <ul style="list-style-type: none"> ● direct access |
| user-specified | additional libraries and object modules | <ul style="list-style-type: none"> ● direct access ● magnetic tape |

Table 10. Correspondence Between Linkage Editor ddnames and Device Classes

| ddname | Possible Device Classes |
|----------------|--|
| SYSLIN | SYSSQ, SYSDA, or the input stream device (specified by DD* or DD DATA) |
| SYSLIB | SYSDA |
| SYSUTI | SYSDA |
| SYSLMOD | SYSDA |
| SYSPRINT | A, SYSSQ |
| user-specified | SYSDA, SYSSQ |

LOAD MODULE EXECUTION

Execution ddnames

In the RPG source program the File Description specifications entries are used to identify the data sets (files). Data sets processed by the RPG load module must be defined by DD statements. The relationship is established between the data set defined in the source program and the DD statement by the ddname. The ddname must be the same

as the entry in Filename (columns 7-14) on the File Description Specifications sheet.

Execution Cancellation

During execution of the load module (object program), job processing is cancelled when a halt indicator (H0-H9) is detected on (see Halt Indicators).

Execution Return Codes

Table 11 shows the return codes issued by the RPG load module (object program) for use with the COND= parameter of the JOB or EXEC statements.

USING CATALOGED PROCEDURES

Because writing job control statements can become time-consuming work for the programmer, IBM supplies three cataloged procedures to aid in the compiling, link editing, and executing of RPG programs.

Compile

The cataloged procedure for compilation is RPGEN. It is invoked by specifying the name RPGEN as the first parameter in an EXEC statement (refer to Cataloged Procedures).

With the procedure RPGEN, a DD statement RPG.SYSIN indicating the location of the source program must be supplied.

The control statements that can be used to invoke the procedure are,

```

//jobname JOB
//stepname EXEC RPGEN
//RPG.SYSIN DD *
      [-----]
      |           |
      |   RPG Source Program   |
      |           |
      [-----]
/*

```

Table 11. Load Module Return Codes

| Return Code | Explanation |
|-------------|---|
| 0 | normal execution, no errors |
| 28 | H0 - initialized on or on due to programmer request |
| 32 | H0 - invalid chaining request |
| 36 | H0 - collating sequence error (matching records) |
| 40 | H0 - record sequence error (predetermined sequence) |
| 44 | H0 - undefined record type |
| 48 | H0 - BSAM I/O error (combined file) |
| 52 | H0 - DAM I/O error |
| 56 | H0 - ISAM I/O error (random processing) |
| 60 | H0 - ISAM I/O error (sequential processing) |
| 64 | H0 - QSAM I/O error |
| 68 | H1 - on |
| 72 | H2 - on |
| 76 | H3 - on |
| 80 | H4 - on |
| 84 | H5 - on |
| 88 | H6 - on |
| 92 | H7 - on |
| 96 | H8 - on |
| 100 | H9 - on |
| 132 | program interrupt (data): may occur when using non-numeric decimal data in a field specified as numeric data, or when using packed decimal data in a field specified as unpacked data |
| 144 | program interrupt (decimal divide): will occur if the value of FACTOR 2 of a DIV operation is zero |

Note: The COND=parameter is explained in the Job Control Language publication.

Link Edit and Execute

The cataloged procedure to link edit RPG object modules and execute the resulting load module is RPGECLG. It is invoked by specifying the name RPGECLG as the first parameter in an EXEC statement.

The cataloged procedure to link edit and execute consists of the control statements shown in Cataloged Procedures.

With the procedure RPGECLG, a DD statement LKED.SYSIN, which indicates the location of the object module, must be supplied.

The control statements that can be used to invoke the RPGECLG cataloged procedure are,

```
//jobname JOB
//stepname EXEC RPGECLG
//LKED.SYSIN DD *
```

```
-----
          RPG Object Module
-----
/*
```

When the RPG compiler is maintained in a private library, a JOBLIB DD statement is required in the input stream to process an RPG program. The JOBLIB DD statement is used to temporarily chain the private library with the system library. The format of the DD statement is,

```
//JOBLIB DD DSNAME= libname,           X
//                               DISP=(OLD,PASS), etc.
```

with libname being the fully qualified name of the RPG library. The JOBLIB DD statement must appear immediately before the first EXEC statement for the job. The concatenation is in effect only for the duration of the job.

Compile, Link Edit, and Execute

The cataloged procedure, RPGECLG, passes a source module through three procedure steps-compile, link edit, and execute. The cataloged procedure is invoked by specifying the name RPGECLG as the first parameter in an EXEC statement.

The statements that invoke the cataloged procedure RPGECLG are,

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
```

```
-----
          RPG Source Program
-----
/*
```

The SYSIN data set (source module) must be defined to the compiler as a separate DD statement not contained in the cataloged procedure.

CATALOGED PROCEDURES

This section contains figures showing the job control statements used in the RPG cataloged procedures and a brief description of each procedure.

If the user selects MVT (Option 4) of the control program he must be aware of several items; the direct access space requirements of SYSOUT data sets, that SPACE= and UNIT= keyword parameters are valid for any SYSOUT data set, the size of the region that is selected by default when the REGION= parameter is not specified for any job step. For additional information refer to the System Generation manual (see Preface).

The cataloged procedures make use of the OS/360 job control language symbolic parameter facility. Refer to Modifying Cataloged Procedures Symbolic Parameter Values in this section.

Compile

The cataloged procedure for compilation (RPGEC) is shown in Figure 130. The numbers to the right correspond to the numbered explanations that follow.

1. This statement assigns default values to the symbolic parameters defined in this procedure.
2. The system name IESRPG identifies the RPG compiler. The REGION= keyword parameter specifies the region size that is to be allocated to the job step, provided MVT in the operating system is specified. If MVT is not specified, the REGION= parameter is ignored. The size of the region required for any job step depends on the size of the program, the access methods used, the size and number of buffers requested, and the work areas required by the various system functions (e.g. OPEN, EOVS, CLOSE, etc.). Refer to the Control Program Options section of the Storage Estimates manual (see Preface).
The COND= parameter can be added to this statement by the EXEC statement that calls the procedure.
3. The destination for the compiler output listing is defined by this statement as the standard system output class, SYSOUT=A.
4. The destination for the object module (provided the option DECK has been specified) is defined as the standard system output class, SYSOUT=B. Usually the device assigned to this output class is a card punch.
5. The three compiler utility data sets are described by these statements. The SEP= subparameter in the last

statement and the SPACE= parameter in each statement are effective only if the device assigned is a direct access device. The number of specifications in the source program determines the space requirement. The procedure provides an initial allocation of 60,000 bytes and additional allocations (if required) of 12,000 bytes.

6. This statement describes the compiler output (object module) produced for input to the linkage editor. The compiler option LOAD causes the object module to be written on this data set. The third term in the DISP parameter causes the data set to be deleted if the job step terminates abnormally.

NOTE: DCB parameters must not be overridden for any data set used in the Compile step. To override the device assignments of SYSUT1 and/or SYSUT2 data sets, the user must assign both data sets to the same type of direct access device which must be a device other than the 2321 (e.g. 2311, 2314, 2301, etc.).

Link Edit and Execute

The cataloged procedure to link edit RPG object modules and execute the resulting load modules (RPGELG) is shown in Figure 131. The numbers to the right correspond to the numbered explanations that follow.

1. This statement assigns default values to the symbolic parameters defined in this procedure.
2. This statement initiates linkage editor execution. The options in the PARM= field cause the linkage editor to put out a cross-reference table, and a list of all control statements processed. The LET option causes the linkage editor to mark the load module as executable even though errors were encountered during processing. The REGION= parameter specifies the region size required for the execution of this job step (provided the user has selected MVT in the operating system control program). Refer to Control Program Options section of the Storage Estimates manual (see Preface).
3. This statement indicates that the input to the linkage editor is from the input stream.
4. The output from the linkage editor is specified as a member of a temporary data set, residing on a direct access device, and is passed to a succeeding job step. The third term in the DISP parameter causes the data set to be deleted if the job step terminates abnormally.

```

//DEFAULT PROC RGREGN=52K, RGPARM='MODECK,LOAD,LIST,NOSEQN', 1)
//          RGU3SPC='(600,(100,20))',RGU2SPC='(600,(100,20))', 2) (1)
//          RGU1SPC='(600,(100,20))',RGGOSPC='(80,(200,50))' 3)
//RPG      EXEC PGM=IESRPG,REGION=&RGREGN,PARM=(&RGPARM) 4) (2)
//SYSPRINT DD SYSOUT=A 5) (3)
//SYSPUNCH DD SYSOUT=B 6) (4)
//SYSUT3   DD UNIT=SYSSQ,SPACE=&RGU3SPC 7)
//SYSUT2   DD UNIT=SYSSQ,SPACE=&RGU2SPC 8) (5)
//SYSUT1   DD SPACE=&RGU1SPC,UNIT=(SYSSQ,SEP=(SYSUT2,SYSUT3)) 9)
//SYSGO    DD DSNAME=&&GO,UNIT=(SYSSQ,SEP=SYSPUNCH), 10) (6)
//          DISP=(MOD,PASS,DELETE),SPACE=&RGGOSPC

```

Figure 130. Compile Catalogued Procedure (RPGEC)

```

//DEFAULT PROC LKREGN=100K,LKPARM='XREF,LIST,LET',GOCOND='(5,LT,LKED)', 1) (1)
//          LKMDSPC='(1024,(50,20,1))',LKU1SPC='(1024,(50,20))' 2)
//LKED     EXEC PGM=IEWL,REGION=&LKREGN,PARM=(&LKPARM) 3) (2)
//SYSLIN   DD DDNAME=SYSIN 4) (3)
//SYSLMOD  DD DSNAME=&&GOSET(RPG),UNIT=SYSDA,DISP=(NEW,PASS,DELETE), 5) (4)
//          SPACE=&LKMDSPC
//SYSUT1   DD SPACE=&LKU1SPC, 6) (5)
//          UNIT=(SYSDA,SEP=(SYSLIN,SYSLMOD))
//SYSPRINT DD SYSOUT=A,DCB=(BLKSIZE=121) 7) (6)
//GO       EXEC PGM=*.LKED.SYSLMOD,COND=&GOCOND 8) (7)

```

Figure 131. Link Edit and Execute Catalogued Procedure (RPGELG)

5. The utility data set for the linkage editor is described by this statement.
6. The destination for the linkage editor output listing is defined by this statement as the standard output class, SYSOUT=A.
7. This statement initiates execution of the link edited load module. The notation *.LKED.SYSLMOD identifies the load module as being in the data set described in the job step LKED by the DD statement named SYSLMOD. The REGION= keyword parameter is not included in this job step. The effect of this omission is that the default region size is selected (providing the user has selected MVT in the operating system control program). If the default region size is not sufficient, refer to the section Overriding Parameters in the EXEC Statement.

Modifying Cataloged Procedures Symbolic Parameter Values

The RPG cataloged procedures make use of the job control language symbolic parameter facility. Refer to IBM System/360 Operating System, Job Control Language (C28-6539). A symbolic parameter is a symbol that appears in the operand of a cataloged procedure statement. Symbolic parameters are assigned values either on the EXEC statement that invokes the procedure or on a PROC statement in the procedure.

In the RPG procedures, symbolic parameters are used for REGION, COND, and PARM specifications on the EXEC statement, and for SPACE specifications on the DD statements. These values may be temporarily changed by coding the symbolic parameter and the desired value on the EXEC statement used to invoke the procedure.

Examples

In procedure RPGECLG (Figure 132), suppressing production of a listing might be desired. In this case, the EXEC statement in the input stream would be,

```
//stepname EXEC RPGECLG, RGPARM='NOLIST'
```

In the procedure RPGECLG (Figure 130), a punched object deck can be obtained, the source deck sequence checked, and SPACE= parameters of the data set SYSUT1 respecified by,

```
//stepname EXEC RPGECLG, X
// RGPARM='DECK,SEQN', X
// RGU1SPC='(200,(80,40))'
```

In the procedure RPGECLG (Figure 131), the link edit region (LKREGN) and the execution condition (GOCOND) specifications may be changed with,

```
//stepname EXEC RPGECLG, X
// GOCOND='(3,LT,LKED)', X
// LKREGN=90K
```

If it is desired to modify an EXEC or DD statement parameter not specified symbolically, it is necessary to override the desired statement with an appropriate statement in the input stream.

The symbolic parameter names are listed in Figure 132.1.

Overriding Statements in Cataloged Procedures

EXEC and DD statements appearing in cataloged procedures can be overridden. Such overriding of statements or fields is effective only for the duration of the job step in which the statements appear. The statements, as stored in the procedure library of the system, remain unchanged.

Overriding for the purposes of respecification, addition, or nullification is accomplished by including in the input stream statements containing the desired changes and identifying the statements to be overridden.

| Symbolic Name | Job Step | Description | Assigned Value |
|--|-----------|--|--|
| &RGREGN &RGPARM &RGU3SPC &RGU2SPC &RGU1SPC &RGGOSPC | Compiler | Region size PARM = field SYSUT3 space SYSUT2 space SYSUT1 space SYSGO space | 52K 'NODECK, LIST, LOAD, NOSEQN' '(600, (100,20))' '(600, (100,20))' '(600, (100,20))' '(80, (200,50))' |
| &LKREGN &LKMDSPC &LKU1SPC &LKPARM | Link Edit | Region size SYSMOD space SYSUT1 space PARM = field | 100K '(1024, (50,20,1))' '(1024, (50,20))' 'XREF, LIST, LET' |
| &GOCOND &GOCOND | Go | COND = field for RPGECLG COND = field for RPGECLG | '((9,LT,RPG), (5,LT,LKED))' '(5,LT,LKED)' |

Figure 132.1. Symbolic Parameter Name Assignments

Overriding Parameters in the EXEC Statement

It may be desirable to add the PARM=, COND=, or REGION= parameters to an RPG procedure EXEC statement which does not have the parameter. This can be done by including in the EXEC statement calling the procedure the notation PARM.stepname=,COND.stepname=, or REGION.stepname=, followed by the desired change. "Stepname" identifies the EXEC statement within the procedure to which the modification applies. Overriding the PGM= parameter is not possible.

Overriding and Adding DD Statements

A DD statement with name "procstep.ddname" is used to override parameters not containing symbolic parameters in DD statements in cataloged procedures, or to add DD statements to cataloged procedures. The procstep identifies the step in the cataloged procedure. If ddname is the name of a DD statement

1. present in the step, the parameters in the new DD statement override parameters in the DD statement in the procedure step.
2. not present in the step, the new DD statement is added to the step.

In any case, the modification is only effective for the current execution of the cataloged procedure.

When overriding, the original DD statement in the cataloged procedure is copied, and the parameters specified in it are replaced by the corresponding parameters in the new DD statement. Therefore, only parameters that must be changed are specified in the overriding DD statement.

If more than one DD statement is modified, the overriding DD statements must be in the same order as the DD statements appear in the cataloged procedure. Any DD

statements that are added to the procedure must follow overriding DD statements.

When the procedures RPGECLG and RPGECLG are used, a DD statement must be added to define the SYSIN data set to the compile step in the procedures (see Using Cataloged Procedures). When the procedure RPGECLG is used, a DD statement must be added to define the SYSLIN data set (see Using Cataloged Procedures).

Examples

In the procedure RPGECLG (Figure 130), the UNIT= parameter of the data set SYSUT1 can be respecified by including in the input stream:

```
//stepname EXEC RPGECLG
//RPG.SYSUT1 DD UNIT=(2311,SEP=(SYSUT2,SYSUT3))
```

In procedure RPGECLG (Figure 132) changing the COND= parameter to the EXEC statement which specifies execution of the linkage editor and specifying a region size of 60K in the GO step might be desired. In this case, the EXEC statement in the input stream would appear as follows:

```
//stepname EXEC RPGECLG, X
// COND.LKED=(9,LT,stepname.RPG),X
// REGION.GO=60K
```

Execution of the linkage editor job step //LKED is suppressed if the return code issued by the compiler (step RPG) was greater than 8 and the region size requested for the GO step was 60K.

The Job Control Language and Utilities publications provide additional description of overriding techniques.

RPG SOURCE PROGRAM DECK ARRANGEMENT

The deck prepared by the programmer is arranged as shown in Figure 133. The contents of the Operating System Job Control Statements are listed in the Job Control Language publication (see Preface).

The order in which the programmer places his control cards and source deck is as follows:

1. Operating System Job Control Statements
2. RPG Control Card (Processor Control Card)
3. File Description Specifications
4. File Extension Specifications
5. Line Counter Specifications
6. Input Specifications
7. Calculation Specifications
8. Output-Format Specifications

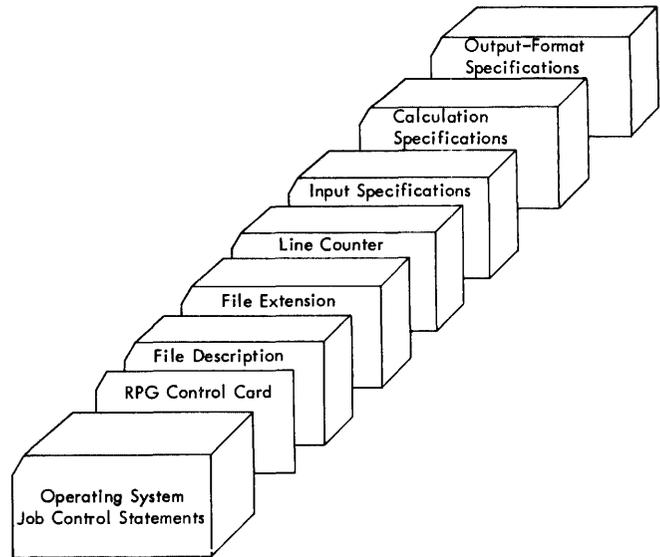


Figure 133. RPG Deck Arrangement in the Operating System Input Stream

RPG CONTROL CARD

The contents of the RPG control card follow.

| <u>Column</u> | <u>Contents</u> | <u>Explanation</u> | <u>Column</u> | <u>Contents</u> | <u>Explanation</u> |
|---------------|-------------------|--|---------------|-------------------|---|
| 1-5 | XXXXX | Page and Line numbers of the control card (can be 00000). | | | |
| 6 | H | Identifies the card as a header card. | | | |
| 7-16 | blank | Not used; may be left blank. | 20 | 0, 1, 2, or blank | for purposes of compatibility; it is treated the same as a blank. If the Sterling pence field on output is in the IBM format, punch 1. If it is in the BSI format, punch 2. Otherwise leave blank. |
| 17 | 1, 2, or blank | If the Sterling shillings field on input is in the IBM format, punch 1 in this column. If it is in the BSI format, punch 2. Otherwise leave blank. | 21 | I or blank | <u>Inverted Print</u> . If numeric literals and Edit words use the European conventions of punctuation (commas for decimal point and vice-versa), enter an I in this column. Otherwise leave blank. |
| 18 | 1, 2, or blank | If the Sterling pence field on input is in the IBM format, punch 1. If it is in the BSI format, punch 2. Otherwise leave blank. | 22-25 | blank | Not used; may be left blank. |
| 19 | 0, 1, 2, or blank | If the Sterling shillings field on output is in the IBM format, punch 1. If it is in the BSI format, punch 2. Otherwise leave blank or punch 0. The zero is allowed only | 26 | A, or blank | <u>Alternate Collating Sequence</u> . Enter an A in this column, if |

| <u>Column</u> | <u>Contents</u> | <u>Explanation</u> |
|---------------|-----------------|--|
| | | an external subroutine is used to translate the sequence of a matching field to the collating sequence of the System/360. If an external translating subroutine is not used, leave this column blank. |
| 27-74 | Blank | The name of the external subroutine is predefined by RPG to be ALTSEQ. Not used; may be left blank |
| 75-80 | XXXXXX | <u>Program Identification</u> . Enter in these columns the program identification. If column 75 is blank, the name RPGOBJ is used for program identification. The first four characters of the identification will be punched in Columns 73-76 of each card in the object program deck. Columns 77-80 of the object deck cards will contain a sequence number. |

EXECUTING RPG — INPUT STREAM VARIATIONS

The input stream varies considerably depending on the user's application. This section illustrates the input stream for various combinations of control statements source decks, user subroutines and data decks.

NOTE: When using a sequential scheduler, only one data set can be included in the input stream for each job step or procedure step. For additional information, refer to IBM System/360 Operating System, Job Control Language.

IBM supplies three cataloged procedures that are explained in Cataloged Procedures and illustrated in Using Cataloged Procedures. Additional examples are included in this section to illustrate inclusion of user subroutines for execution with the RPG load module.

The EXIT operand can be used to incorporate subroutines in assembler language. For convenience, frequently used subroutines are maintained in a library. The following illustrates three methods for including user subroutines with the RPG object module as input to the linkage editor. In these examples the user must supply DD statements corresponding to the File Description specifications. The LIBRARY statement is used to call the library in order to resolve the designated external references (SUBR1, SUBR2). The LKED.SUBLIB DD statement defines the private library containing the subroutines. These subroutines can be either load modules or object modules with control statements.

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
      [ RPG source program ]
/*
//LKED.SUBLIB DD DSNAME=libname,DISP=OLD
//LKED.SYSIN DD *
      LIBRARY SUBLIB(SUBR1,SUBR2)
/*
```

The INCLUDE statement is used to cause the linkage editor to process the subroutine modules (SUBR1,SUBR2). The LKED.SUBLIB DD statement identifies the private library that contains the subroutines. These subroutines can be either load modules or object modules with control statements.

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
      [ RPG source program ]
/*
//LKED.SUBLIB DD DSNAME=libname,DISP=OLD
//LKED.SYSIN DD *
      INCLUDE SUBLIB(SUBR1,SUBR2)
/*
```

The placement of the user subroutines that are included as object modules in the input stream of a compile and execute job is,

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
```

```
[RPG source program]
```

```
/*
//LKED.SYSIN DD *
```

```
[SUBR1 object module]
```

```
[SUBR2 object module]
```

```
[SUBR3 object module]
```

```
/*
```

Subroutines can be included both from a library and in the input stream. The job shown below includes subroutines SUBR1 and SUBR2 from the library and SUBR3 from the input stream.

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
```

```
[RPG source program]
```

```
/*
//LKED.SUBLIB DD DSNAME=libname,DISP=OLD
```

```
//LKED.SYSIN DD *
INCLUDE SUBLIB(SUBR1,SUBR2)
```

```
[SUBR3 object module]
```

```
/*
```

The operating system provides the capability to assemble user subroutines within the same job as the RPG compilation. The assembly can either precede or follow or both precede and follow the RPG compilation. The job setup for an assembly following the RPG compilation is,

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSGO DD DSNAME=&LOADSET
//RPG.SYSIN DD *
```

```
[RPG source program]
```

```
/*
//stepname EXEC ASMECLG
//ASM.SYSIN DD *
```

```
[source program for assembly]
```

```
/*
```

The first EXEC statement invokes the cataloged procedure for compilation (RPGECLG). The RPG.SYSGO DD statement overrides the cataloged procedure renaming the data set that contains the object module to be the same data set as used by the assembler. The object module that is output from the assembly is added to the data set that subsequently is input to the linkage editor. The subparameter MOD (in RPGECLG) specifies that the data set is added to and causes logical positioning after the last record in the data set.

The second EXEC statement invokes the cataloged procedure for assembly, link edit, and execute (ASMECLG). Refer to the Assembler (E) Programmer's Guide for a detailed description of cataloged procedures for assembly.

Execution must begin in the RPG produced object module. When an assembly precedes the RPG compilation, the RPG compiler is not the first program executed. Therefore, an ENTRY statement is required to specify the name of the RPG program. In order to link edit the assembly object module with the RPG object module, the SYSPUNCH DD statement in the ASMECLG catalog procedure must be overridden. The assembly object module must be on SYSSQ. The job setup for assembly of the user subroutine preceding the RPG compilation is,

```
//jobname JOB
//stepname EXEC ASMECLG
//ASM.SYSPUNCH DD DSNAME=&GO,UNIT=SYSDA, X
//          SPACE=(80,(200,50)),DISP= X
//          (MOD,PASS)
//ASM.SYSIN DD *
```

```
[source program for assembly]
```

```
/*
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
```

```
[RPG source program]
```

```
/*
//LKED.SYSIN DD *
ENTRY      source program identification
/*
```

The deck arrangement for jobs processing data files is,

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
    [RPG source program]
/*
//GO.ddname (parameters)
//GO.ddname (parameters)
.
.
.
.
//GO.ddname (parameters)
/*
```

Data sets (files) processed by an RPG program must be defined by DD statements (see Load Module Execution).

The deck arrangement for jobs processing data files and tables is,

```
//jobname JOB
//stepname EXEC RPGECLG
//RPG.SYSIN DD *
    [RPG source program]
/*
//GO.ddname (parameters)
//GO.ddname (parameters)
//GO.SYSIN DD *
    [tables]
/*
```

APPENDIX A. SAMPLE PROGRAMS

Three complete sample programs are included in this section.

The labels assigned to the fields into which the object program will place the information are as follows:

SAMPLE PROGRAM ONE

The input file in the first program consists of punched cards. Each card in the file contains a data record that includes from one to eighty characters of information. Each data record represents a purchase made from the reporting firm by a customer. The types of information and the card columns in which each appears are shown in Figure 134.

| <u>Field</u> | <u>Label</u> |
|----------------------|--------------|
| Customer Name | NAME |
| Invoice Data — Month | MONTH |
| Invoice Data — Day | DAY |
| Invoice Number | INVNO |
| Customer Number | CUSTNO |
| Customer Location | STATE |
| Customer Location | CITY |
| Invoice Amount | INVAMT |

Figure 135 is an output listing of the sample program.

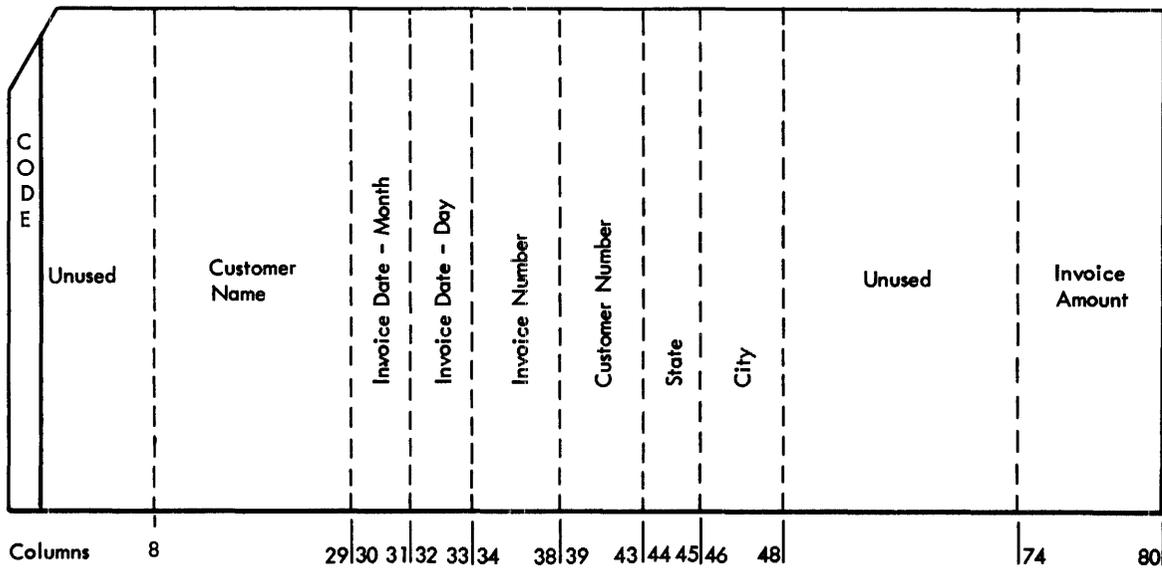


Figure 134. Input File-Card Format

```

OS/RPG(32K)*V1.L0          SAMPLI  RPG          09/20/66          PAGE 0001
00 000 H
001 01 010 FINPUT IPE F 80 80          READ40          SAMPLI
002 01 020 FOUTPUT O Y 132 132        DE PRINTER          SAMPLI
003 01 010 IINPUT AA 01 1 7-          SAMPLI
004 01 030 I          8 29 NAME          SAMPLI
005 01 030 I          30 310MONTH          SAMPLI
006 01 040 I          32 330DAY          SAMPLI
007 01 050 I          34 380INVNO          SAMPLI
008 01 060 I          39 430CUSTNO11          SAMPLI
009 01 070 I          44 450STATE          SAMPLI
010 01 080 I          46 480CITY          SAMPLI
011 01 090 I          74 802INVANT          SAMPLI
012 01 010 C 01 INVANT ADD TOTAL TOTAL 72          SAMPLI
013 01 020 C 01 INVANT ADD GRPTOT GRPTOT 72          SAMPLI
014 01 010 OOUTPUT H 201 1P          SAMPLI
015 01 020 O          DR OF          SAMPLI
016 01 030 O          53 * A C C O U N T S R *          SAMPLI
017 01 040 O          77 * E C C E I V A B L E R E *          SAMPLI
018 01 050 O          88 * G I S T E R *          SAMPLI
019 01 060 O          H 1 1P          SAMPLI
020 01 070 O          DR OF          SAMPLI
021 01 080 O          25 * C U S T O M E R *          SAMPLI
022 01 090 O          80 * L O C A T I O N I N V O I C E *          SAMPLI
023 01 100 O          109 * I N V O I C E D A T E I N V O I C E *          SAMPLI
024 01 110 O          H 2 1P          SAMPLI
025 01 120 O          DR OF          SAMPLI
026 01 130 O          42 * N U M B E R C U S T O M E R *          SAMPLI
027 01 140 O          46 * N A M E *          SAMPLI
028 01 150 O          79 * S T A T E C I T Y N U M B E R *          SAMPLI
029 01 160 O          108 * M O D A Y A M O U N T *          SAMPLI
030 02 010 O          D 2 01          SAMPLI
031 02 020 O          C U S T N O Z 23          SAMPLI
032 02 030 O          N A M E 53          SAMPLI
033 02 040 O          S T A T E Z 59          SAMPLI
034 02 050 O          C I T Y Z 67          SAMPLI
035 02 060 O          I N V N O Z 79          SAMPLI
036 02 070 O          M O N T H Z 90          SAMPLI
037 02 080 O          D A Y Z 96          SAMPLI
038 02 090 O          I N V A N T 109 * $ , 0 . *          SAMPLI
039 02 100 O          T 2 11          SAMPLI
040 02 110 O          G R P T O T 109 * $ , 0 . *          SAMPLI
041 02 120 O          110 * * *          SAMPLI
042 02 130 O          T 2 1R          SAMPLI
043 02 140 O          T O T A L 109 * $ , 0 . *          SAMPLI
044 02 150 O          111 * * *          SAMPLI

```

```

OS/RPG(32K)*V1.L0          SAMPLI  RPG          09/20/66          PAGE 0002
                                SYMBOL TABLES
RESULTING INDICATORS
ADDRESS RI          ADDRESS RI          ADDRESS RI          ADDRESS RI          ADDRESS RI          ADDRESS RI
000011 DF          000014 1P          000015 LR          000016 00          000017 01          00007A L0          00007B L1
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
000123 NAME          000139 MONTH          000138 DAY          00013D INVNO          00014G CUSTNO
000143 STATE          000145 CITY          000147 INVANT          00014B TGTAL          00014F GRPTOT

LITERALS
ADDRESS LITERAL          ADDRESS LITERAL          ADDRESS LITERAL
000153 A C C O U N T S R          00016B E C C E I V A B L E R E          000183 G I S T E R
00018E C U S T O M E R          000196 L O C A T I O N I N V O I C E          0001AC I N V O I C E D A T E I N V O I C E
0001C3 N U M B E R C U S T O M E R          00010B N A M E          0001DF S T A T E C I T Y N U M B E R
0001F7 M O D A Y A M O U N T          00020C --/--/--          000217 *
000218 **

```

```

                                MEMORY MAP
INPUT/OUTPUT INTERCEPT          000220
TABLE (INPUT AND OUTPUT)          00021C
DETERMINE RECORD TYPE          00044C
DATA SPECIFICATION          00024B
GET INPUT RECORD          00076C
DETAIL CALCULATIONS          00091C
TOTAL CALCULATIONS          000970
DETAIL LINES          000A8A
TOTAL LINES          000984
INPUT/OUTPUT REQUEST BLOCKS POINTER          0013D0
LOCATION OF DCB POINTERS          000D30
INPUT/OUTPUT INTERFACE ROUTINES          000F40
LINE COUNTER          000D80
WORK AREA POINTER          001840
OVERFLOW BYPASS          000A82
CONTROL LEVEL          00061C
TABLE(ASSEMBLE 4)          0008C4
TEST ZONE (BCD)          001414
OVERFLOW LINES          000A02
LINKAGE PROGRAM          00154C

```

```

OS/RPG(32K)*V1.L0          SAMPLI  RPG          09/20/66          PAGE 0003
PROGRAM LENGTH 001961
*END OF COMPILATION*

```

Figure 135. Output Listing

To produce an object module capable of writing the report shown in Figure 136, the programmer must prepare a source program as shown in Figure 137. The entries in the RPG specifications sheet are described here.

The output file OUTPUT is also defined on the File Description sheet. The format is variable; the block length is 132 and the records are 132 characters in length. The entry OF in columns 33-34 indicates that the output file defined on the line is to cause the overflow condition.

FILE DESCRIPTION SHEET

Two files (input and output) are described on this sheet. The input file INPUT (columns 7-11) is the primary file (column 16). It causes the end-of-job condition when it is depleted (column 17). The input records are fixed in length (column 19); the block length is 80 (columns 22 and 23); and the records are 80 characters in length (columns 26 and 27).

INPUT SPECIFICATIONS SHEET

The input file has a sequence of AA, and if column 1 contains the zone of a minus, Resulting Indicator 01 is turned on (as indicated by the entries in 19-20, 24 and 26-27). The locations of the fields which contain the input data are defined in columns 44-51. The names of the input fields are entered in columns 53-58. Whenever a

| ACCOUNTS RECEIVABLE REGISTER | | | | | | | |
|------------------------------|------------------|----------------|------|----------------|------------|-------------|----------------|
| CUSTOMER NUMBER | CUSTOMER NAME | LOCATION STATE | CITY | INVOICE NUMBER | INVOICE MO | INVOICE DAY | INVOICE AMOUNT |
| 10712 | AMALGAMATED CORP | 33 | 61 | 11603 | 11 | 19 | \$ 389.25 |
| | | | | | | | \$ 389.25* |
| 11315 | BROWN WHOLESALE | 39 | 231 | 12324 | 12 | 28 | \$ 802.08 |
| 11315 | BROWN WHOLESALE | 39 | 231 | 99588 | 12 | 14 | \$ 261.17 |
| | | | | | | | \$ 1,063.25* |
| 11897 | FARM IMPLEMENTS | 47 | 77 | 10901 | 10 | 18 | \$ 27.63 |
| | | | | | | | \$ 27.63* |
| 18930 | BLACK OIL | 16 | 67 | 11509 | 11 | 8 | \$ 592.95 |
| 18930 | BLACK OIL | 16 | 67 | 12292 | 12 | 23 | \$ 950.97 |
| | | | | | | | \$ 1,543.92* |
| 20716 | LEATHER BELT CO | 36 | 471 | 11511 | 11 | 8 | \$ 335.63 |
| 20716 | LEATHER BELT CO | 36 | 471 | 12263 | 12 | 17 | \$ 121.75 |
| | | | | | | | \$ 457.38* |
| 29017 | GENERAL MFG CO | 6 | 63 | 11615 | 11 | 14 | \$ 440.12 |
| 29017 | GENERAL MFG CO | 6 | 63 | 11676 | 11 | 23 | \$ 722.22 |
| | | | | | | | \$ 1,162.34* |
| 29054 | A-B-C DIST CO | 25 | 39 | 9689 | 9 | 11 | \$ 649.40 |
| 29054 | A-B-C DIST CO | 25 | 39 | 11605 | 11 | 13 | \$ 271.69 |
| 29054 | A-B-C DIST CO | 25 | 39 | 12234 | 12 | 14 | \$ 559.33 |
| | | | | | | | \$ 1,476.42* |
| | | | | | | | \$ 6,120.19** |

Figure 136. Printed Report

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
IBM System 360

Form X24-3352-1
Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Filename | Space | | | Skip | | | Output Indicators | | | Field Name | Zero Suppress (Z) Blank After (B) | End Position in Output Record | Packed Field (P) | Constant or Edit Word | Sterling Sign Position |
|------|-----------|----------|--------------|---------------|--------|-------|--------|-------|-------------------|---------|---------|------------|--------------------------------------|-------------------------------|------------------|-----------------------|------------------------|
| | | | Type (H/D/I) | Shrink Select | Before | After | Before | After | And Not | And Not | And Not | | | | | | |
| 0 1 | 0 | OUTPUT | H | | | | | | | | | | | | | | |
| 0 2 | 0 | | OR | | | | | | | | | | | | | | |
| 0 3 | 0 | | | | | | | | | | | | | 53 | | ACCOUNTS R | |
| 0 4 | 0 | | | | | | | | | | | | | 77 | | ECEIVABLE RE | |
| 0 5 | 0 | | | | | | | | | | | | | 88 | | G I S T E R | |
| 0 6 | 0 | | H | | | | | | | | | | | | | | |
| 0 7 | 0 | | OR | | | | | | | | | | | | | | |
| 0 8 | 0 | | | | | | | | | | | | | 25 | | CUSTOMER | |
| 0 9 | 0 | | | | | | | | | | | | | 80 | | LOCATION INVOICE | |
| 1 0 | 0 | | | | | | | | | | | | | 109 | | INVOICE DATE INVOICE | |
| 1 1 | 0 | | H | | | | | | | | | | | | | | |
| 1 2 | 0 | | OR | | | | | | | | | | | | | | |
| 1 3 | 0 | | | | | | | | | | | | | 42 | | NUMBER CUSTOMER | |
| 1 4 | 0 | | | | | | | | | | | | | 46 | | NAME | |
| 1 5 | 0 | | | | | | | | | | | | | 79 | | STATE CITY NUMBER | |
| 1 6 | 0 | | | | | | | | | | | | | 108 | | MO DAY AMOUNT | |

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION
REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS
IBM System 360

Form X24-3352-1
Printed in U. S. A.

Date _____

Program _____

Programmer _____

| | | | | | | | | | |
|----------------------|---------|--|--|--|--|--|--|--|--|
| Punching Instruction | Graphic | | | | | | | | |
| | Punch | | | | | | | | |

Page 1 2

Program Identification 75 76 77 78 79 80

| Line | Form Type | Filename | Space | | | Skip | | | Output Indicators | | | Field Name | Zero Suppress (Z) Blank After (B) | End Position in Output Record | Packed Field (P) | Constant or Edit Word | Sterling Sign Position |
|------|-----------|----------|--------------|---------------|--------|-------|--------|-------|-------------------|---------|---------|------------|--------------------------------------|-------------------------------|------------------|-----------------------|------------------------|
| | | | Type (H/D/I) | Shrink Select | Before | After | Before | After | And Not | And Not | And Not | | | | | | |
| 0 1 | 0 | | D | | | | | | | | | | | | | | |
| 0 2 | 0 | | | | | | | | | | | | | | | | |
| 0 3 | 0 | | | | | | | | | | | | | | | | |
| 0 4 | 0 | | | | | | | | | | | | | | | | |
| 0 5 | 0 | | | | | | | | | | | | | | | | |
| 0 6 | 0 | | | | | | | | | | | | | | | | |
| 0 7 | 0 | | | | | | | | | | | | | | | | |
| 0 8 | 0 | | | | | | | | | | | | | | | | |
| 0 9 | 0 | | | | | | | | | | | | | | | | |
| 1 0 | 0 | | T | | | | | | | | | | | | | | |
| 1 1 | 0 | | | | | | | | | | | | | | | | |
| 1 2 | 0 | | | | | | | | | | | | | | | | |
| 1 3 | 0 | | T | | | | | | | | | | | | | | |
| 1 4 | 0 | | | | | | | | | | | | | | | | |
| 1 5 | 0 | | | | | | | | | | | | | | | | |

Figure 137. RPG Specification Sheets — Program One (Part 2 of 2)

new customer number is read in, control level 1 is turned on (columns 59-60).

CALCULATION SPECIFICATIONS SHEET

The contents of the field TOTAL are added to the contents of the field INVAMT, and the result is stored in TOTAL. The result field has a length of seven positions, and two positions are reserved for the decimal portion. The field GRPTOT is added to INVAMT, and the result is stored in GRPTOT which is seven positions long, and has two decimal positions.

OUTPUT-FORMAT SPECIFICATIONS SHEET

OUTPUT, the name of the file to which the records defined on the line belong, is entered under Filename on the first line

of the sheet. In column 15 the output types, H, D and T are entered to designate the heading, detail, and total lines.

The first heading line, ACCOUNTS RECEIVABLE REGISTER, prints either on the first page (1P) or overflow conditions (OF). The OR entered in columns 14 and 15 of the second line allows for printing on the first page or on overflow. The other heading lines also print on these conditions.

When Output Indicator 01 is on, the field entered in Field Name will print in the positions indicated in columns 40-43. Zero suppression occurs on CUSTNO, STATE, CITY, INVNO, MONTH, and DAY.

The total lines are to print whenever control fields L1 or LR are on. The group total GRPTOT prints when L1 is ON, and after it is printed, the contents of GRPTOT are blanked out. The final total is printed when the LR (last record) indicator is on.

SAMPLE PROGRAM TWO

This is similar to the previous program. In this example, however, two input files are used. The Transaction File is a card file with fields as shown in Figure 134 of the previous program. Another input file (Master Customer File), which is on tape, contains information about the firm's customers (Figure 138). The fields contained in the two input files are illustrated in Figure 139.

The program is to process the master customer file, using records from the transaction file, to produce printed receipts. The master file is updated, by producing a new master customer file. The coding required for this program is shown in Figure 140.

FILE DESCRIPTION SHEET

The four files are defined on this sheet. The two input files TRANSIN and MASTERIN

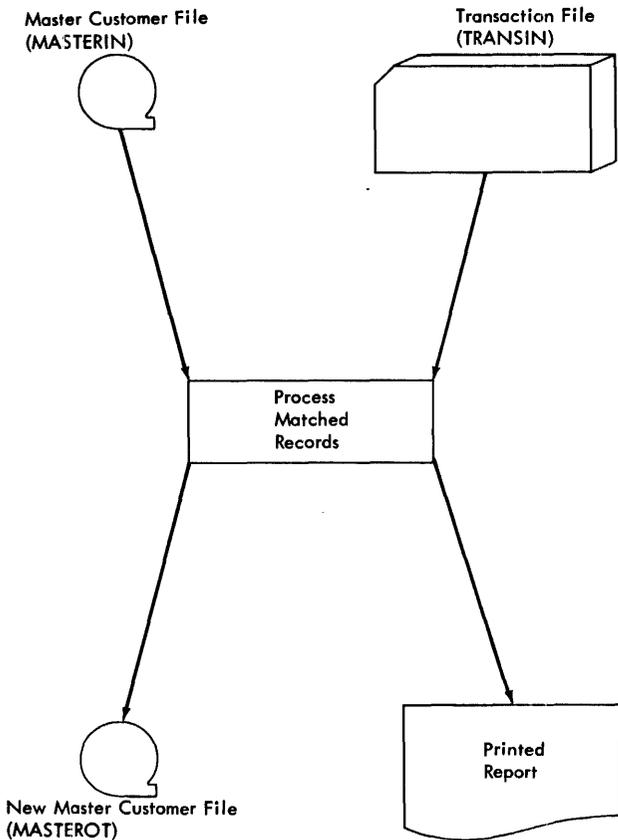


Figure 138. Sample Program Two

are defined, and the two output files MASTEROT, which is the updated master file, and MASTLIST, which is the printed report, are defined on the file-description sheet. TRANSIN is designated as the secondary file because it may not contain transactions

Transaction File

| Field | Label | Card Columns |
|-----------------------------------|---|--------------|
| Code | Minus (-) Zone, or Plus (+) Zone | 1 |
| Customer Name | NAME | 8-29 |
| Invoice Date | MONTH | 30-31 |
| Invoice Date | DAY | 32-33 |
| Invoice Number | INVNO | 34-38 |
| Customer Number | CUSTNO | 39-43 |
| State | STATE | 44-45 |
| City | CITY | 46-48 |
| Invoice / Amount Amount / Paid | AMT | 74-80 |

Master Customer File

| Field | Label | Location |
|-----------------------|---------|----------|
| Customer Number | CUSTNO | 1-5 |
| Customer Name | MASNAM | 6-27 |
| Street | MASTRT | 28-46 |
| City | MASCTY | 47-57 |
| State | MASTAT | 58-62 |
| Customer Balance | MASBAL | 64-70 |
| Date Of Last Payment | PAYDATE | 71-76 |
| Date Of Last Purchase | PAYPUR | 77-82 |

Figure 139. Input Fields - Sample Program Two

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3347-2
REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS Printed in U.S.A.
 IBM System 360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic Punch

Page 1 2

Program Identification: 75 76 77 78 79 80

| Line | Form Type | Filename | File Designation | | Mode of Processing | | Device | Symbolic Device | Name of Label Exit | Extent Exit for DAM | Comments |
|------|-----------|--------------|------------------|-------------|--------------------|---------------|---------|-----------------|--------------------|---------------------|----------|
| | | | Sequence | File Format | Length of Record | Address Field | | | | | |
| 0 1 | φ | TRANSIN IS | AF | 80 | 80 | READ40 | | | | | |
| 0 2 | φ | MASTERIN IPE | AF | 300 | 100 | TAPE | | S | | | |
| 0 3 | φ | MASTERTO | F | 300 | 100 | TAPE | | | | | |
| 0 4 | φ | MASTLISTO | V | 132 | 132 | OF | PRINTER | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3350-1
REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS Printed in U.S.A.
 IBM System 360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic Punch

Page 1 2

Program Identification: 75 76 77 78 79 80

| Line | Form Type | Filename | Sequence Number (N) | Option (O) | Record Identification Codes | | | Field Location | | Field Name | Field Indicators | | | Sterling Sign Position |
|------|-----------|------------|---------------------|------------|-----------------------------|-----------|----------|----------------|------|------------|------------------|------|-------|------------------------|
| | | | | | Position | Character | Position | Character | From | | To | Plus | Minus | |
| 0 1 | φ | TRANSIN AA | 02 | 1 | Z | | | | | | | | | |
| 0 2 | φ | OR | 03 | 1 | Z+ | | | | | | | | | |
| 0 3 | φ | | | | | | | 8 | 29 | NAME | | | | |
| 0 4 | φ | | | | | | | 30 | 31 | MONTH | | | | |
| 0 5 | φ | | | | | | | 32 | 33 | DAY | | | | |
| 0 6 | φ | | | | | | | 34 | 38 | INVNO | | | | |
| 0 7 | φ | | | | | | | 39 | 43 | CUSTNOL1M1 | | | | |
| 0 8 | φ | | | | | | | 44 | 45 | STATE | | | | |
| 0 9 | φ | | | | | | | 46 | 48 | CITY | | | | |
| 1 0 | φ | | | | | | | 74 | 80 | 2AMT | | | | |
| 1 2 | φ | BB | 04 | 1 | CD | | | | | | | | | |
| 1 3 | φ | | | | | | | 2 | 7 | DATE | | | 0505 | |
| 1 4 | φ | MASTERINAA | 01 | | | | | | | | | | | |
| 1 5 | φ | | | | | | | 1 | 100 | RECORD | | | | |
| 1 6 | φ | | | | | | | 1 | 50 | CUSTNOL1M1 | | | | |
| 1 7 | φ | | | | | | | 64 | 70 | MASBAL | | | | |
| 1 8 | φ | | | | | | | 71 | 76 | PAYDAT | | | | |
| 1 9 | φ | | | | | | | 77 | 82 | PAYPUR | | | | |

IBM INTERNATIONAL BUSINESS MACHINES CORPORATION Form X24-3351-1
REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS Printed in U.S.A.
 IBM System 360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic Punch

Page 1 2

Program Identification: 75 76 77 78 79 80

| Line | Form Type | Indicators | Factor 1 | Operation | Factor 2 | Result Field | Field Length | Resulting Indicators | | | Comments |
|------|-----------|------------|----------|-----------|----------|--------------|--------------|----------------------|-------|---------------|----------|
| | | | | | | | | Plus | Minus | Zero or Blank | |
| 0 1 | φ | MR 02 | MASBAL | ADD AMT | MASBAL | | | | | | |
| 0 2 | φ | MR 03 | MASBAL | SUB AMT | MASBAL | | | | | | |
| 0 3 | φ | MR 02 | | MOVE DATE | PAYDAT | | | | | | |
| 0 4 | φ | MR 03 | | MOVE DATE | PAYPUR | | | | | | |

Figure 140. Specifications for Sample Program Two (Part 1 of 2)

involving all the customers on the master customer file. TRANSIN is ascending in order. It has fixed-length records which are 80 characters long. The block length is 80.

MASTERIN is the primary file. If the transaction file does not have a corresponding customer number (specified by the Matching Fields specification on the input sheet) the master file is processed. Processing continues until all the records in the master customer file have been processed (indicated by the E in column 17). The input records contained in the master customer file are ascending in order, fixed in length, and have a block length of 300. Each record is 100 characters in length.

The file MASTLIST is the printed report. The format is variable. The length of the records can be 132. The overflow entry in columns 33-34 indicates that the overflow condition is to occur on the MASTLIST printer.

The output file MASTEROT is a tape file which contains the updated master customer records. The output records are fixed in length, and are 100 characters in length. The block length is 300.

NOTE: A blank in column 53 indicates that there are no labels in the output file.

INPUT SPECIFICATIONS SHEET

The two input files TRANSIN and MASTERIN are defined on the Input Specification sheet.

TRANSIN: The input records may be obtained from three types of cards.

Sequence AA has been assigned to two types. If card column 1 contains the zone of a minus, resulting indicator 02 is turned on. If card column 1 contains the zone of a plus, indicator 03 is turned on. The cards that have a minus in column 1 are selected into the 2 pocket (column 42). The locations of the input records and their labels are defined in columns 46-58 of the sheet.

The field CUSTNO (customer number) has entries in columns 59-60 (Control Level) and columns 61-62 (Matching Fields) of the Input Specifications sheet. Whenever a new customer number is read in, control level 1 (L1) is set on. This condition is tested on the Output-Format Specifications sheet to govern printing of total lines and to produce the updated customer file. The entry in matching fields specifies that customer number will be used to match another field (CUSTNO) in the MASTERIN file.

The first card in the transaction file is a date card. It is assigned sequence BB. Whenever column 1 contains a D, indicator

04 is turned on. The date is contained in columns 2-7 of the card. Indicator 05 is turned on whenever these columns are zero or minus.

MASTERIN: The tape input file that contains information about the firm's customers is assigned sequence AA. The first entry under field name defines the entire record. This entry (RECORD) is made to enable the entire record to be referenced on the Output-Format Specifications sheet. CUSTNO of the master file corresponds to CUSTNO in the transaction file. Whenever a new customer number is read in, L1 is set. The entry M1 indicates that the customer number in the master file will be matched with the customer number in the transaction file.

CALCULATION SPECIFICATIONS SHEET

Whenever the matching record indicator MR is on and indicator 02 is on, the contents of the field AMT are added to the MASBAL. The result is stored in MASBAL. The date is moved to the field PAYDAT.

Whenever the matching record indicator MR is on and indicator 03 is on, AMT is subtracted from MASBAL, and the result is stored in MASBAL. The date is moved to PAYPUR.

OUTPUT-FORMAT SPECIFICATIONS SHEET

The output file MASTEROT is the updated tape file. The entries in output indicators allow for the following. Whenever conditions 01 and NMR are satisfied (resulting indicator 01 is on and no matching record is present), the entire tape input record will be written out on tape. This condition results because there was no corresponding customer number in the transaction file for the master customer number.

To keep the master customer file complete, the old input record is written out on the updated tape file when no information is present in the transaction file.

If, however, L1 and MR are on, the input record is written out on tape. The entire record is written, but the fields MASBAL, PAYDAT, and PAYPUR contain the new entries based on the calculations. By coding the entries in this way, the new information for MASBAL, PAYDAT, and PAYPUR is entered on the master customer file, but the customer's name and address are retained.

The specifications for the printed report are listed under the name of the output file MASTLIST.

SAMPLE PROGRAM THREE

The third sample (Figures 141 and 142) illustrates some of the more complex capabilities of RPG.

1. Processing chained records on a

Direct Access Storage Device (DASD).

2. Updating records on a DASD.
3. Multiple input and output files.
4. Creating exception records.
5. Providing for processing when a record in the chained file is missing.

| CARDIN | | CARDOUT | | INVFIL | |
|-------------------------|-------------|---------------|-------------|---------------|----------|
| Field | Card Column | Field | Card Column | Field | Position |
| <u>Date Card</u> | | Code | 1 | Code | 1 |
| Code | 1 | Part Number | 2-9 | Part Number | 2-9 |
| Date | 2-7 | Description | 10-36 | Description | 10-36 |
| Not Used | 8-80 | Vendor Number | 37-42 | Vendor Number | 37-42 |
| <u>Transaction Card</u> | | Date | 43-48 | Not Specified | 43-49 |
| Code | 1 | Not Used | 49-80 | Minimum | 50-53 |
| Part Number | 2-9 | | | On Hand | 54-58 |
| Receipts | 10-13 | | | Not Specified | 59-125 |
| Issues | 14-17 | | | | |
| Returns | 18-21 | | | | |
| Not Used | 22-80 | | | | |

| INVENTORY LISTING | | 5/28/66 | | PAGE 1 | | | |
|-------------------|------------------------------|---------|---------|----------|--------|---------|-------------------|
| PART NUMBER | PART DESCRIPTION | MIN BAL | OLD BAL | RECEIPTS | ISSUES | RETURNS | NEW BAL |
| 00101238 | HEX NUT | 1,000 | 3,500 | 100 | 600 | | 3,000 |
| 00101239 | WASHER | 2,500 | 3,100 | 500 | 1,000 | | 2,600 |
| 00101240 | LKWSHR | 1,500 | 3,700 | | 3,500 | 100 | 300 BELOW MINIMUM |
| 00101241 | BOLT,6-IN | 500 | 650 | 50 | 100 | 50 | 650 |
| 00101242 | BOLT,8-IN | 500 | 645 | 100 | 245 | | 500 EXPEDITE |
| 00101243 | NO DISK RECORD FOR THIS PART | | | | | | |
| 00101244 | MACHINE SCREW,1-IN | 800 | 1,100 | 800 | 1,200 | 400 | 1,100 |

Figure 141. Input and Output Formats for Sample Program Three

| IBM | | INTERNATIONAL BUSINESS MACHINES CORPORATION | | REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS | | IBM System/360 | | Form X24-3347-2 U/M025 Printed in U.S.A. | | | | | | | | |
|---|-----------|---|------------------|--|----------|----------------|--------------------|---|-------|-----------------------------|--------|-----------------------------|--------------------|---------------------|----------|--------------------|
| Date _____ | | | | | | | | | | | | | | | | |
| Program _____ | | | | | | | | | | | | | | | | |
| Punching Instruction: Graphic <input type="checkbox"/> Punch <input type="checkbox"/> | | | | | | | | | | | | | | | | |
| Page 1 2 | | | | | | | | | | | | | | | | |
| Programmer _____ | | | | | | | | | | | | | | | | |
| Program Identification: 75 76 77 78 79 80 | | | | | | | | | | | | | | | | |
| Line | Form Type | Filename | File Type | | | | Mode of Processing | | | | Device | Symbolic Device | Name of Label Exit | Extent Exit for DAM | Comments | |
| | | | I/O/U/C | P/S/C/R/T | A/D | E/V | U/R | K/T | I/O/T | Key Field Starting Location | | | | | | Extension Code E/L |
| | | | File Designation | End of File | Sequence | File Format | Block Length | Record Length | U/R | K/T | I/O/T | Key Field Starting Location | Extension Code E/L | | | |
| 0 1 | P | INVFIL | UC | F | 125 | 125 | R | KD | | | | | | DISK11 | | S |
| 0 2 | P | CARDIN | IP | E | 80 | 80 | | | | | | | | EREAD40 | | |
| 0 3 | P | CARDOUT | O | | 80 | 80 | | | | | | | | READ40 | | |
| 0 4 | P | PRINTOUT | V | | 132 | 132 | | | | | OF | | | PRINTER | | |

| IBM | | INTERNATIONAL BUSINESS MACHINES CORPORATION | | REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS | | IBM System/360 | | Form X24-3348-1 Printed in U.S.A. | | | | |
|---|-----------|---|-------------|--|------------------------------------|----------------|-----------------|--------------------------------------|-----------------|------------|--------------|----------|
| Date _____ | | | | | | | | | | | | |
| Program _____ | | | | | | | | | | | | |
| Punching Instruction: Graphic <input type="checkbox"/> Punch <input type="checkbox"/> | | | | | | | | | | | | |
| Page 1 2 | | | | | | | | | | | | |
| Programmer _____ | | | | | | | | | | | | |
| Program Identification: 75 76 77 78 79 80 | | | | | | | | | | | | |
| Line | Form Type | Record Sequence of the Chaining File | | Table Name | Number of Table Entries Per Record | | | Table Name (Alternating Table) | Length of Table | | | Comments |
| | | From Filename | To Filename | | Per Table | Entry | Length of Table | | Entry | Packed (P) | Decimal Pos. | |
| 0 1 | E | CCC1 | CARDIN | INVFIL | CONRTN | | | | | | | |

Figure 142. Specifications for Sample Program Three (Part 1 of 5)

FILE DESCRIPTION SPECIFICATIONS

The primary input file, designated by a P in column 16, is on the card reader and is labeled CARDIN. This card file also acts as a transaction file for the updating of the inventory records contained on the file labeled INVFIL. The U in column 15 of the specification for INVFIL indicates that this DASD file is updated (used for both input and output). The file has direct organization indicated by the D in column 32, and is processed randomly as specified by the R in column 28.

The E in column 17 of the CARDIN file description entry directs the program to end execution when the last record of the CARDIN file has been processed. The E in

column 39 indicates that a file extension specification has been coded for the CARDIN file.

The output files are an exception card file labeled CARDOUT and the printed transaction report labeled PRINTOUT.

CARDIN and CARDOUT consist of 80-character records. INVFIL has a block length of 125 characters and a record length of 125 characters. Record length for PRINTOUT is variable, with a maximum of 132 characters.

FILE EXTENSION SPECIFICATIONS

The file extension sheet identifies CARDIN as the chaining file, indicating that record

APPENDIX B. INDICATOR CHART

| Indicators | Where Specified | Where Used | Turned On | Turned Off | Notes |
|--|---|--|--|--|---|
| Resulting Indicator 00 | Internal | Output Indicators on Output Specifications (Columns 24 - 25, 27 - 28, 30 - 31) | This indicator is always ON | Can never be turned OFF | |
| Resulting Indicators (01 - 99) | Input Specifications (Columns 19 - 20) | Indicators on Calculation Specifications (Columns 10-11, 13-14, 16-17) Output Indicators on Output Specifications (Columns 24-25, 27-28, 30-31) | When specified record type has been read and is ready for processing | Before the first record is read on the next processing cycle | Turning OFF and ON can also be accomplished by using SETON and SETOF operation codes |
| | Calculation Specifications (Columns 54-55, 56-57, 58-59) | Same as above | Whenever the specified field status condition is satisfied | The next time that this field status is to be tested | Same as above |
| Field Indicators (01 - 99) | Input Specifications (Columns 65-66, 67-68, 69-70) | Same as above | Same as above | Same as above | Same as above |
| Halt Indicators (H0 - H9) | Input Specifications (Columns 65-66, 67-68) Calculation Specifications (Columns 54-55, 56-57, 58-59) | Same as above | Same as above. H0 is automatically turned on for the conditions listed in Appendix H | Can only be turned OFF by SETOF operation code See Note | If these indicators remain ON, the object program will terminate before reading the next record |
| LR | Internal | Same as above and Calculation Specifications (Columns 7-8) | After processing the last record of the last file | | All Control Level Indicators (L1-L9) are also turned ON when the LR is turned ON |
| Control Level Indicators (L1-L9) | Input Specifications (Columns 59-60) | Same as above and Calculation Specifications (Columns 7-8) | When the value in a control field changes. All indicators of the lower levels are also turned ON | Before the first record is read on the next processing cycle | Turning OFF and ON can be accomplished by using SETON and SETOF operation codes |
| L0 | Internal | Indicators on Calculations Specifications (Columns 10-11, 13-14, 16-17) Output Indicators on Output Specifications (Cols. 24-25, 27-28, 30-31) | This indicator is always ON | Can never be turned OFF | |
| MR | Internal | Same as above | When multiple input files and the matching fields specification are used, this indicator is turned ON if a secondary file record matches the primary file record | Before the first record is read on the next processing cycle | |
| Overflow Indicators OA, OB, OC, OD, OE, OF, OG, OV | File Description (Columns 33-34) | Same as above | One line after channel 12 of the carriage control tape is sensed | After the detail and heading records are written | These indicators remain ON for one complete processing cycle |
| 1P | Internal | Output Indicators on Output Specifications (Columns 24-25, 27-28, 30-31) | This indicator is ON at the beginning of processing before any records are read | Before the first record is read | This indicator is used to govern printing of the first page of the report |

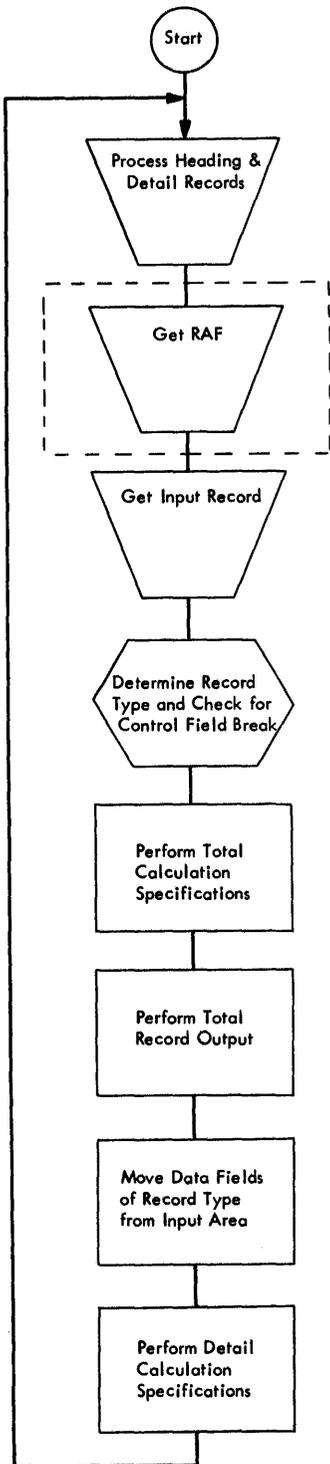


Chart 1. General Logic for RPG Object Module with RAF

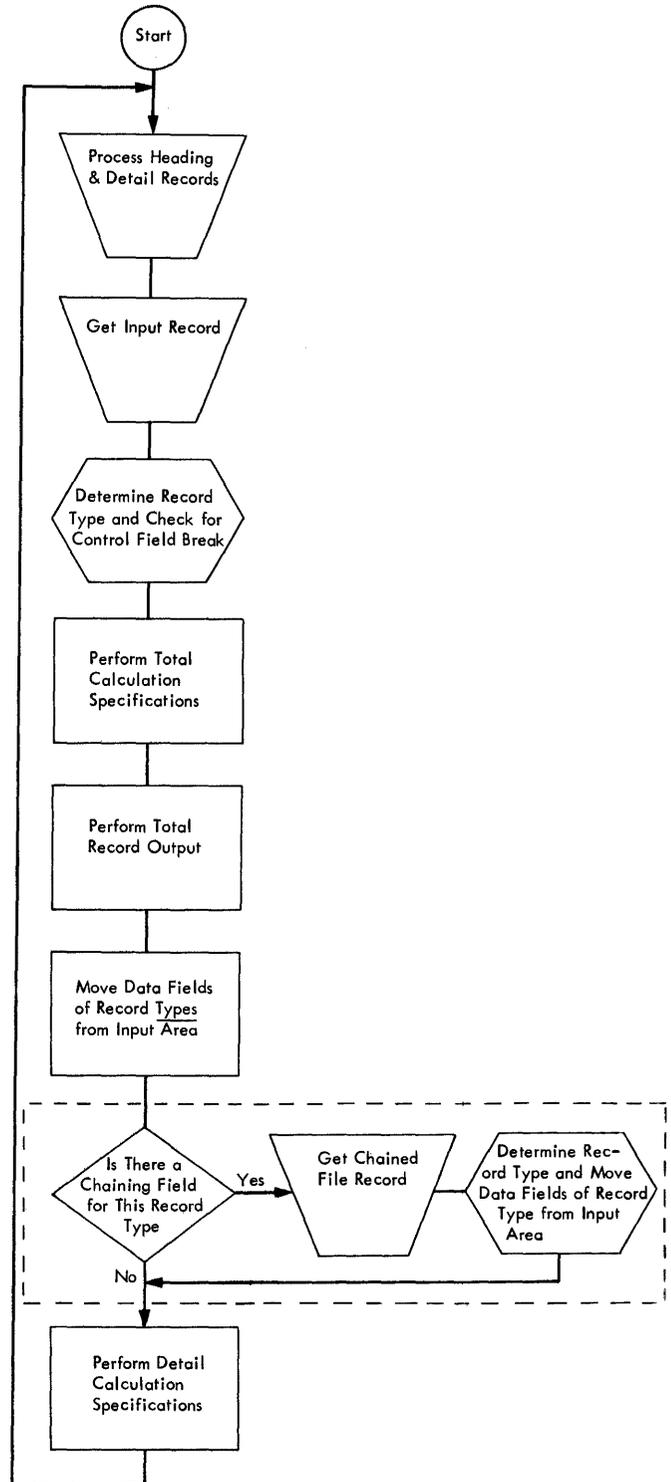


Chart 2. General Logic for RPG Object Module with a Chaining File

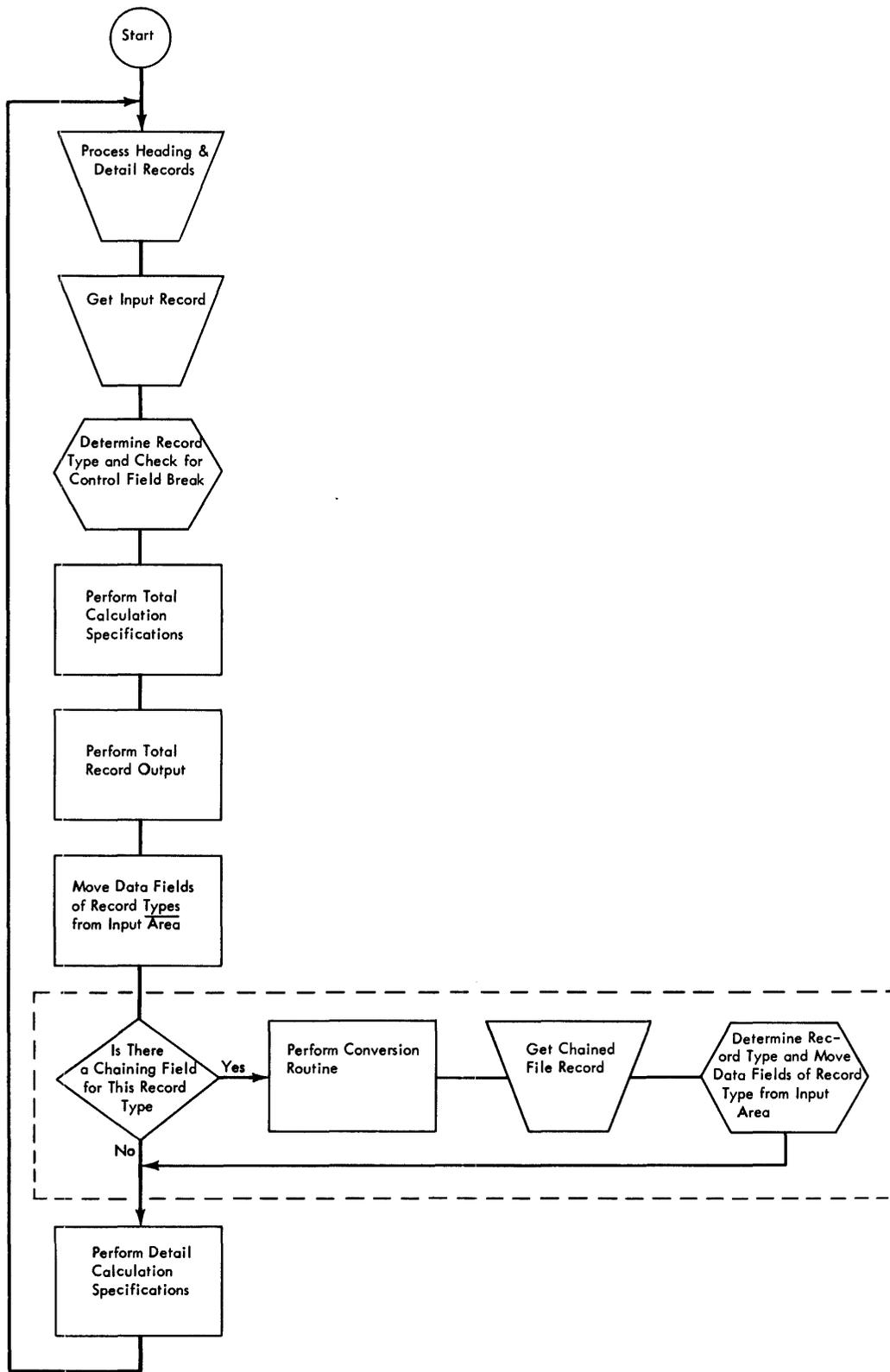


Chart 3. General Logic for RPG Object Module with Chaining File Which Requires Conversion

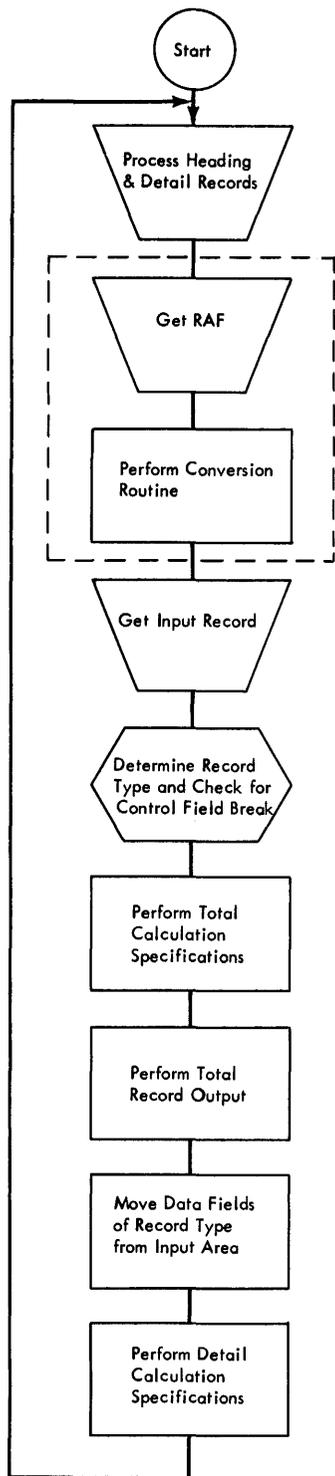


Chart 4. General Logic for RPG Object Module with RAF which Requires Conversion

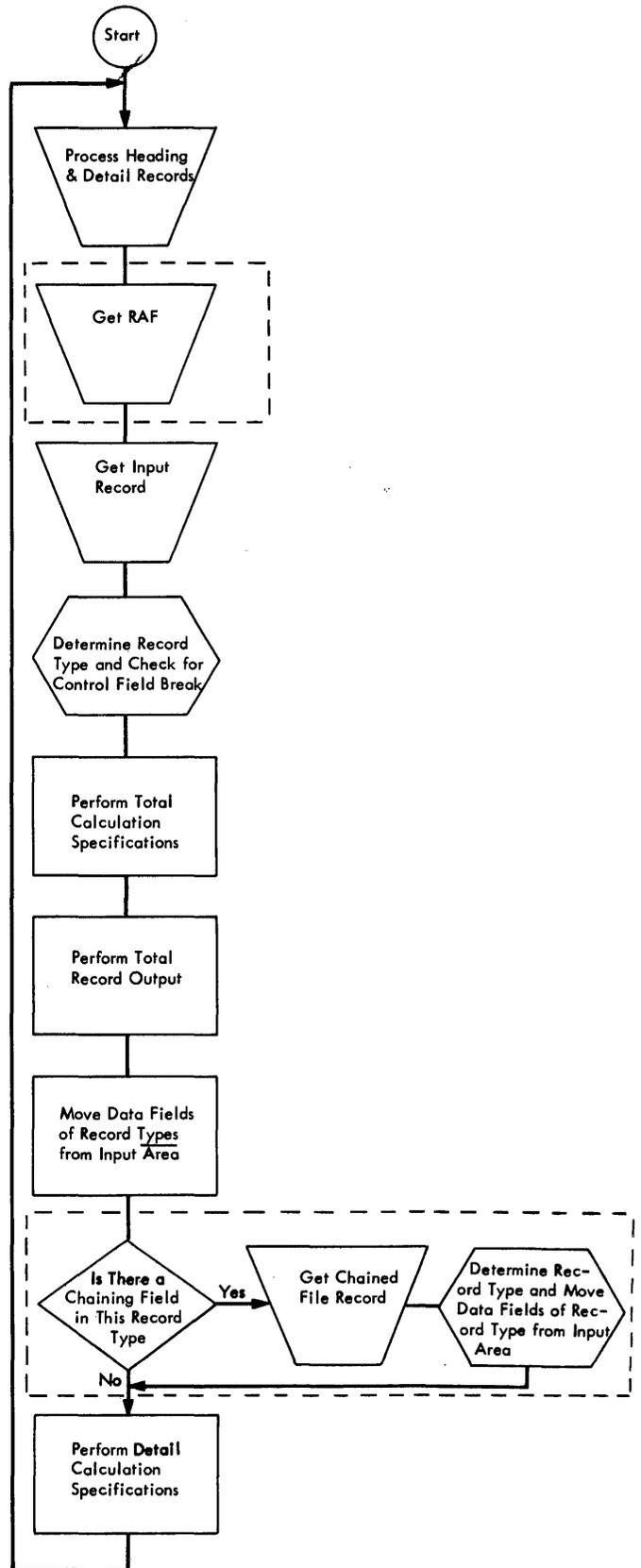


Chart 5. General Logic for RPG Object Module with RAF and Chaining Files

APPENDIX D. STERLING ROUTINES FOR THE REPORT PROGRAM GENERATOR

The RPG sterling routines furnish users with a convenient and time-saving means of handling sterling amounts. The presence of sterling fields is indicated to the RPG program by additional entries in the input and output specifications forms and in the control card. The file description, file extension, and calculation forms are not affected.

Sterling input information can be represented in two formats: IBM and BSI as described in the control card. The RPG sterling routines convert the input fields into a pence-format field. A pence-format field is a sterling amount represented in pence. If the output is to be printed, the fields are converted with shillings and pence printed in two positions each with zero suppression in effect in the tens position of each field. If the output is not printed, the output is converted to either BSI or IBM formats.

NOTE 1: On both input and output the pounds field must consist of at least one, and no more than nine positions.

NOTE 2: BSI or IBM input files of one program must use the same code combination throughout.

INPUT SPECIFICATIONS

The position of the sign must be specified in columns 71-74. Enter an S in column 74, if the sign is in the normal position. If the pence field has decimal positions, the normal position of the sign is in the right-most decimal position of the pence field. If the pence field has no decimal positions, the normal position of the sign is in the units position of the pounds field.

NOTE 1: One of the digits 0, 1, 2, or 3 must be entered in column 52, to indicate the number of required decimal positions.

NOTE 2: It is not permissible to use the same name for both a sterling field and a decimal field.

NOTE 3: The sign of the field must contain a numeric underpunch.

OUTPUT FORMAT SPECIFICATIONS

The position of the sign for sterling output fields must be specified in columns

71-74 in the same manner as for sterling input fields. The sterling sign will always appear on output whether the field is plus, minus, or zero.

Output Which is Not Printed. The field may be specified as any combination of IBM or BSI shillings and pence formats. The sign may appear anywhere within the record. When outside the field, the sign will be supplied with a zero underpunch.

Printed Output. The normal sign position must be used. Insert the letter S in column 74 of the Output Specifications Sheet.

NOTE 1: Shillings and pence are printed in two positions each. When no edit word is specified zero suppression is in effect in the tens positions of each field.

NOTE 2: The pounds field consists of at least one and no more than 9 positions. Zero suppression on the pounds field may be obtained by placing Z in column 38 of the specification sheet. The sign will not be suppressed since its position depends on the presence or absence of decimal pence.

NOTE 3: If a field is defined as a sterling field in the input but not in the output specification, the output will be in pence format.

NOTE 4: Editing is allowed only on printed output files. The rules governing the use of edit control words are the same as those for decimal fields. The features available are:

1. Zero suppression in the pounds field.
2. Zero suppression in the shillings field, if both pound and shilling values are zero.
3. Zero suppression in the pence field, if pound, shilling and pence values are zero.
4. Suppression of zeros preceding signs, and suppression of separation marks between pounds and shillings, shillings and pence, and pence and decimals.

CONTROL CARD

To select the required sterling routines, the RPG program needs information regarding the input and output formats. This information is entered in four columns of the

RPG processor control card. The entries are are: 1 for IBM Code, 2 for BSI Code.

CALCULATION SPECIFICATIONS

While no additional entries are required in this form, the user should keep in mind that all calculations are done in pence format. This must be considered when defining the length of result fields or when using Factors 1 and 2.

Lengths of Pence-Format Fields

If a pence-format result field is to be re-converted into a sterling output field, the highest amount it is permitted to contain is 239, 999, 999, 999.999. This converts to a field containing nine pounds positions which is the maximum allowed.

NOTE: In order to avoid the possible loss of the high order digit, fields that are read in as IBM or BSI sterling always contain one more position when put out as IBM or BSI sterling. For example, the five position sterling input field 9919+ (IBM format) converts to the five position pence field 23999. Since the RPG compiler in putting out this field must allow for a five position pence field containing up to 99999 pence (which converts to 416133, IBM format), the field on output will be six positions long.

Pound Sterling Formats

In addition to the printed output format, RPG will support, on the input and output fields, two standards for pence and shilling portions of sterling fields: IBM or BSI. Columns 17-20 of the RPG processor control card indicate either the IBM or BSI formats. The formats for IBM and BSI are listed here.

Column 17 (Sterling-Shilling Field On Input) IBM Format. Two positions are allowed for the shilling option in the input fields: 00-19 for 0 to 19 shillings.

BSI Format. The shilling option in the input fields is indicated as listed here:

0-9 Shillings by a 0-9 punch,
10 Shillings by a 12-punch,
11-19 Shillings by an A-I punch.

Column 18 (Sterling-Pence Field on Input) IBM Format. The pence option on the input field is as listed here:

0-9 Pence by a 0-9 punch,
10 Pence by an 11-punch,
11 Pence by a 12-punch.

BSI Format. The pence option on the input field is as listed here:

0-9 Pence by a 0-9 punch,
10 Pence by a 12-punch,
11 Pence by an 11-punch.

Columns 19 (Sterling-Shillings Field on Output) IBM Format. Two positions are allowed for the shilling option on the output field:

00-19 for 0-19 shillings

BSI Format. The shilling option on the output field is as listed here:

0-9 Shillings by a 0-9 punch,
10 Shillings by a 12-punch,
11-19 Shillings by an A-I punch.

Column 20 (Sterling-Pence Field on Output) IBM Format. The pence option on the output field is as listed here:

0-9 Pence by a 0-9 punch,
10 Pence by an 11-punch,
11 Pence by a 12-punch.

BSI Format. The pence option on the output field is as listed here:

0-9 Pence by a 0-9 punch,
10 Pence by a 12-punch,
11 Pence by an 11-punch.

APPENDIX E. CONVERSION ROUTINE OPERATION CODES

The following list shows the relationship between conversion routine operation codes and how they are specified.

| CODE | FACTOR 1 | FACTOR 2 | RESULT FIELD | SPECIFICATIONS THAT MAY FOLLOW THIS ENTRY |
|-------|----------------|--------------------------------------|--|--|
| RPGCV | Reference Name | No entry | Label of the field which will contain the track address of the record. | <u>KEYCV</u> |
| EXTCV | Reference Name | Name or Label of the user's routine. | Label of the field which will contain the track address of the record. | <u>KEYCV</u> |
| KEYCV | None | None | Label of the field which will contain the key of the record to be located. | If <u>RPGCV</u> has been used, the conversion steps follow <u>KEYCV</u> . If <u>EXTCV</u> has been used, any other RPG calculations follow. |
| ERPGC | None | None | None | Any other calculations in the RPG program. |

APPENDIX F. SUMMARY OF RPG SPECIFICATION FORMS

This summary contains a brief column-by-column description of each of the six RPG specification forms. The purpose of the summary is to provide the user with a concise reference guide.

The first five items are common to all six specification forms.

Page 1-2

Enter number of specification page. Assign ascending numbers to the pages of each program specification set to collate in the following sequence:

- File Description Specifications
- File Extension Specifications
- Line Counter Specifications
- Input Specifications
- Calculation Specifications
- Output-Format Specifications

Line 3-5

First two digits of line number are pre-printed. Use third position (column 5) to identify additional lines to be inserted between two preprinted lines.

Form Type 6

Contains a preprinted code (F, E, L, I, C, or O) which must be punched into all RPG specification cards.

Comments 7

Enter an asterisk (*) in each line to be used exclusively as a comments line.

Program Identification 75-80

Insert any information to identify certain cards or portions of an RPG source program.

FILE DESCRIPTION SPECIFICATIONS

Filename 7-14

Enter a name for each file used in the program. Names must be left-justified.

File Type 15

Enter one of the letters I, O, U, or C to identify Intput, Output, Uppdate, or Combined file.

File Designation 16

P for primary, S for secondary input or combined files. Enter P if only one input file or combined file is used. C for chained file, R for RAF or T for table file. Leave blank for output files.

End of File 17

Enter E for each input file or combined file that is to be checked to determine when the last record has been read and processed (LR indicator on). Leave blank if LR is to be turned on when the last record of all input and combined files has been read.

Sequence 18

Required when Matching Fields is used. A if the input file or combined file is in ascending sequence, D if it is in descending sequence. Leave blank for output files, or for input or combined files without Matching Fields.

File Format

F: Fixed-length records.

V: Variable-length records.

Block Length 20-23

Unblocked

If the records are unblocked, enter the length of a record. If variable-length records are used, enter the length of the longest record.

Blocked

If the records are blocked, enter the length of the largest block.

Record Length 24-27

Used to enter the length of the logical records contained in the file. If the file contains records that are variable in length, enter the length of the largest record.

Mode of Processing 28

Used to indicate the method or mode by which the file is processed. Enter an L in this column if a segment of the file is to be processed. Enter an R in this column if the records are to be processed randomly. If no entry is made in this column for the file, the entire file will be processed sequentially.

Length of Record Address Field 29-30

If the file is a record-address file, enter the number of positions that each entry in the RAF occupies.

Record Address Type 31

If the records from the file are retrieved by using record keys, enter a K in this column. If record ID is used to retrieve records, enter I in this column.

Type of File Organization 32

Leave blank if the file is organized sequentially. Enter an I if the file has indexed-sequential organization. Enter a D if the file has direct organization.

Overflow Indicator 33-34

If overflow indicators are used, enter the overflow indicator associated with the file. A maximum of eight overflow indicators is allowed: OA, OB, OC, OD, OE, OF, OG, and OV.

Key Field Starting Location 35-38

Indicates the location of the key field within the data record. Enter the starting position of the key field. This entry is required for indexed-sequential files.

Extension Code 39

Indicates that additional information about the file is coded on the File Extension Specifications sheet, or Line Counter Specifications sheet.

Enter an E if the file defined on the line is a:

Chaining file
Table file
Record Address file (RAF)

Device 40-46

Relates a file to a specific type of input or output unit.

If the output file is a printer, enter PRINTER.

If the file is an I/O file and it is associated with a card reader or card punch unit, enter:

| | |
|--------|--------------------------|
| READ01 | IBM 2501 Card Reader |
| READ20 | IBM 2520 Card Read-Punch |
| READ40 | IBM 2540 Card Read-Punch |
| READ42 | IBM 1442 Card Read-Punch |

If the file is an I/O file and it is associated with a tape unit enter TAPE.

If the file is associated with a 2311 Disk Storage Drive, enter DISK11.

Symbolic Device 47-52

Not used.

Labels 53

S Standard Labels.
E Standard Labels Followed by user-standard Labels.
b No Labels.

Name of Label Exit 54-59

Enter the name of the routine to process user-standard labels. If the entry is shorter than six characters, it must be left-justified.

Extent Exit for DAM 60-65

Not used.

Comments 66-74

Enter any desired comments. An asterisk in column 7 must not be used if this type

of comments is in a line containing specifications.

FILE EXTENSION SPECIFICATIONS

Record Sequence of the Chaining File 7-8

Used only for chaining files. Enter the same entry that is made for the chaining file in Sequence on the Input Specifications sheet.

Number of the Chaining Field 9-10

Used only for chaining files. Enter the identifying number of the chaining field. This number is entered in Chaining Field of the Input Specifications sheet.

From Filename 11-18

Used in conjunction with To Filename to identify -- for the RPG program -- the relationship between two files. For example, they provide the name of a chaining file and the name of the file that is chained to it.

To Filename 19-26

(See description above)

Table Name 27-32

Enter name of table:
for alternating input format,
enter name of table to which
the first entry in each input
record belongs;
for non-alternating input format,
enter name of single table.

For an RAF or chaining file, specify the label of the address conversion routine. Table name must consist of 6 characters, the first 3 must be TAB, the remaining 3 may be any alphabetic or numeric characters.

Number of Table Entries Per Record 33-35

Enter the maximum number of table entries (arguments or functions) that are contained in each input record. The entry must be right-justified.

Number of Table Entries Per Table 36-39

Enter the exact number of table entries (arguments or functions) contained in the table. The entry must be right-justified.

Length of Table Entry 40-42

Enter the length of each table entry. The maximum size of a numeric entry is 15 characters, of an alphameric entry 256 characters. The entry must be right-justified.

Packed 43

If the data for the table is in the packed-decimal format, enter P in this column. Otherwise leave this column blank.

Decimal Positions 44

If the data contained in the table is numeric, enter the number of decimal positions (1-9). Enter a zero if there are no decimal positions. If the field is alphameric, leave this column blank.

Sequence 45

If the data contained in the table is in ascending sequence, enter an A; in descending sequence, enter a D. Leave blank if the data is not in ascending or descending sequence or if this specification is not required.

Table Name 46-51

If alternating arguments and function tables are used, enter the second table name. It must be of the form TABnnn. The entry must be left-justified.

Length of Table Entry 52-54

Enter the length of each table entry. The maximum size of a numeric entry is 15 characters; of an alphameric entry 256 characters. The entry must be right-justified.

Packed 55

Enter a P if the data for the table is in the packed-decimal format. Otherwise leave this column blank.

Decimal Positions 56

If the data contained in the table is numeric, enter the number of decimal positions (1-9). Enter a zero if there are no decimal positions. If the field is alphameric, leave blank.

Sequence 57

If the data contained in the table is in ascending sequence, enter an A; in descending sequence, enter a D. Leave blank if the data is not in ascending or descending sequence or if this specification is not required.

Comments 58-74

Leave columns 58-74 blank, unless comments are entered in these columns.

LINE COUNTER SPECIFICATIONS

Filename 7-14

Enter the name of the output file.

Line Number (1) 15-17

Enter the number of the first line controlled by the carriage tape in columns 15-17.

The remaining specifications (items 2-12) perform the same function.

Channel Number (1) 18-19

Enter the channel number corresponding to the line number in columns 15-17. The remaining specifications (items 2-12) perform the same function.

INPUT SPECIFICATIONS

Filename 7-14

Enter a file name for each input or combined file, one entry per file. Must be left-justified; it may have up to 8 characters; first character must be alphabetic; remaining characters may be alphabetic or numeric; special characters or embedded blanks may not be used.

AND/OR Relationship 14-16

Enter AND to indicate that the AND relationship of Record Identification Codes in the preceding line is to be continued. Enter OR (columns 14 and 15) to indicate that the entries in Record Identification Codes of this line are to be in an OR relationship to the entries in the preceding line.

Sequence 15-16

Enter a number, beginning with 01 for each file and continuing in consecutive sequence to 99, to specify sequence-checking of card types. Enter leading zeros. Enter any two alphabetic characters to indicate that sequence-checking is not required. Lines with alphabetic entries in Sequence must precede lines with numeric entries.

Any numeric entry in Sequence requires an entry in Number.

Number 17

- 1 indicates that one and only one record of a specific record type should be present in each group.
- N indicates that one or more records of a specific record type may be present in each group. (Used only with numeric entry in columns 15-16.)

Option 18

- 0 indicates that a record of a specific control group used need not be present. Leave blank if a record must be present, or if records are non-sequential. (Used only with numeric entry in columns 15-16.)

Resulting Indicator 19-20

Enter any indicator from 01 through 99 to establish a two-digit code for the input-record type defined in Record Identification Codes. This sets special condition(s) in the object program each time the input record is read.

Record Identification Codes 21-41

This field is divided into three identical sub-fields: columns 21-27, 28-34, and 35-41. An AND relationship exists between these three sub-fields.

Position. Enter the number of input record column containing the identifying code. Must be right-justified.

Not. Enter N if the code described must not be present in the position specified. Otherwise leave blank.

C/Z/D. Enter D if only the digit portion of the specified position is to be checked. Enter Z if only the zone portion is to be checked. Enter C if both portions are to be checked.

Character. If C/Z/D contains C or D, enter any one of the 256 EBCDIC characters.

If C/Z/D contains Z, enter: &, A through I, or 0 to check for a 12-zone; -, J through R, or 0 to check for an 11-zone; S-Z for 0-zone; 0 through 9, or blank to check for the absence of zones.

NOTE: Record Identification Codes may be continued in the subsequent specification line by means of an AND entry for AND-relationships or an OR entry for OR-relationships.

Stacker Select 42

Enter number of stacker to which input cards are to be selected. Leave blank for single-stacker devices and for combined files.

Packed 43

P if input data in packed-decimal format. Blank if input data in standard format.

Field Location 44-51

This specification describes location of fields in input records.

From. Enter number of input card column containing first position of field specified in Field Name.

To. Enter number of input card column containing last position of field defined in Field Name. Entries must be right-justified; leading zeros may be omitted.

Decimal Positions 52

Used only for numeric fields. Enter a digit 0 through 9 to indicate number of decimal positions in input field. Leave blank for alphameric fields.

NOTE: Each input field that is to be used in arithmetic operations must have an entry in Decimal Positions. Also use this specification for fields that are to be edited or zero-suppressed.

Field Name 53-58

Enter the name of each field defined in Field Location. Field names may be up to 6 characters in length; left justified. First character must be alphabetic; remaining characters may be alphabetic or numeric. Special characters or embedded blanks may not be used.

Control Level 59-60

Enter any one of the control-level indicators L1 through L9 to identify control fields. (L1 for lowest level, L9 for highest level of control.)

Matching Fields or Chaining Fields 61-62

Enter any one of the codes M1, M2, or M3 to specify record-matching for two input files, or to specify sequence-checking for the fields of a single input file.

Enter the codes C1 through C9 to specify a chaining field.

Field-Record Relation 63-64

Enter any one of the indicators defined in columns 19-20 of the Input Specifications to provide field-record relation for identical fields contained in different locations (OR-relationships), or for selective processing of chaining fields.

Field Indicators 65-70

If the field is alphameric, i.e., if column 52 is blank, only the Zero or Blank specification may be used. Enter any one of the indicators 01 through 99 or H0 through H9, as required, in each of the fields. Indicator in Plus (columns 65 and 66) is turned on, if the field specified in columns 53-58 contains a positive value, except +0. Indicator in Minus (columns 67 and 68) is turned on if the field contains a negative value, except -0. Indicator in Zero or Blank (columns 69 and 70) is turned on if the field contains no other character than zero or blank. It is also turned on if the field is numeric and contains no other character than +0 or -0, or when a Blank After output specification is executed.

Sterling Sign Position 71-74

Used only for programs processing sterling currency amounts. If sign of Sterling field is in normal position, enter S in Column 74. If sign is not in normal position, enter the position in the record that contains the sign. Leave blank in all other RPG programs.

alphanumeric: left-justified must be enclosed in apostrophe symbols ('), maximum length 8 characters, any one of the 256 EBCDIC characters may be used.

Apostrophe symbols required within constants must be represented as two consecutive apostrophe symbols.

CALCULATION SPECIFICATIONS

Control Level 7-8

Enter one of control level indicators L1 through L9, L0, or LR to specify that the calculation contained in this line is to be performed at total time. Leave blank if calculation is to be performed at detail time.

Indicators 9-17

Enter one to three indicators to establish conditions controlling calculation specified in the line. Any of the indicators 01 through 99, L1 through L9, L0, LR, MR, or any halt or overflow indicator may be used. Columns 9, 12, and 15 may contain blank or N.

NOTE: If there is more than one indicator in a line, RPG assumes an AND-relationship between the individual indicators.

Factor 1 18-27

Field name — left-justified, maximum length 6 characters. Do not use special characters or embedded blanks. First character must be alphabetic. Must be defined in Input Specifications or as a result field in another calculation specification.

Literal — numeric: left-justified, maximum length ten characters, characters must be numeric (0 through 9), one decimal symbol and/or one sign (plus or minus) allowed. If a sign is present, it must be in the first position of the field.

Operation 28-32

Enter one of the RPG operation codes. An entry in this specification is required in each line, except in a comments line defined by an asterisk in column 7. Entries must be left-justified.

Factor 2 33-42

Enter field name or literal to be used in the specified operation. (For a definition of field name and literal, see Factor 1.) Entries must be left-justified.

Result Field 43-48

Enter field name to designate location in storage into which result of pertinent operation is to be placed. First character of field name must be alphabetic; remaining characters may be alphabetic or numeric, cannot contain special characters or embedded blanks. Must be left-justified.

Field Length 49-51

Enter number of storage positions to be reserved for result field on this line. Maximum length of numeric result fields is 15 digits. Maximum length of alphanumeric result fields is 256 characters. Must be right-justified. May be left blank, if length of result field specified in this line is defined in a previous line of calculation specifications, or in input specifications.

Decimal Positions 52

Enter number of decimal positions (0 through 9) to be reserved in result field. Required for all numeric result fields

used with arithmetic operations. Must be blank if the result field is alphameric.

NOTE: The number of decimal positions is included in Field Length specification in columns 49-51. E.g., if a number of decimal positions specified is 7, the maximum number of positions to the left of the decimal symbol is $15 - 7 = 8$. (The maximum length of a numeric field is 15.)

Half Adjust 53

Enter H to specify half-adjustment of result field. Must be blank if result field is alphameric.

Resulting Indicators 54 - 59

Any one of the indicators 01 through 99, H0 through H9 and L1 through L9 may be used for this specification.

Arithmetic Operations: enter up to 3 indicators to be turned on whenever the result of an arithmetic operation is positive (Plus, columns 54 and 55), or negative (Minus, columns 56 and 57), or zero (Zero, columns 58 and 59).

Compare Operations: enter up to 3 indicators to be turned on whenever result of COMP operation is Factor 1 > Factor 2 (High, columns 54 and 55), or Factor 1 < Factor 2 (Low, columns 56 and 57), or Factor 1 = Factor 2 (Equal, columns 58 and 59).

LOKUP Operations: enter one or two indicators in High, or Low, or Equal, or High and Equal, or Low and Equal. (This defines type of entry to be located by means of LOKUP operation.)

TESTZ Operations: enter up to 3 indicators to be turned on whenever a 12-zone (High, columns 54 and 55), or an 11-zone (Low, columns 56 and 57), or any other zone or no zone (Equal, columns 58 and 59) is detected in the field specified in Result Field of this line.

SETOF, SETON Operations: enter up to 3 indicators to be turned on (SETON) or off (SETOF). If more than 3 indicators are to be turned on (or off), specify another SETON (SETOF) statement in subsequent line. Any RPG indicator can be used, except L0 and 00.

NOTE: Headings High, Low, Equal do not apply to indicators specified in conjunction with SETOF or SETON operations.

Comments 60 - 74

Enter any desired comments. An asterisk in column 7 must not be used if this type of comment is in a line containing specifications.

OUTPUT-FORMAT SPECIFICATIONS

Filename 7 - 14

Enter name of each output file. Names must be left-justified.

AND/OR Relationships 14-16

Enter AND for records in an AND relationship. Enter OR for records in an OR relationship (Columns 14 and 15).

Type 15

H — identifies heading line
D — identifies detail time output
T — identifies total time output

Stacker Select 16

Enter number of stacker to which cards are to be selected. Leave blank for single-stacker devices.

Space 17 - 18

NOTE: At least one entry is required in columns 17-22 if the line is to be printed.

Before. Enter 0, 1, 2, or 3 to specify 0, 1, 2, or 3 lines spacing before printing. Leave blank if no space before printing is required.

After. Enter 0, 1, 2, or 3 to specify 0, 1, 2, or 3 lines spacing after printing. Enter zero to specify no space after printing.

Skip 19-22

Before. Enter any number from 01 through 12 to specify skipping before printing to channel 01 through 12 of the carriage control tape. Leave blank for no skip before printing.

After. Enter any number from 01 through 12 to specify skipping after printing 01 through 12 of the carriage control tape.

Output Indicators 23 - 31

Enter up to three RPG indicators to identify files or to describe fields. Columns 23, 26, 29 must contain blank or the letter N.

Field Name 32 - 37

Enter any field name defined in either input specifications or calculation specifications. Leave blank for constants specified in Constant or Edit Word. Use field name PAGE to cause automatic page numbering.

Zero Suppress 38

Enter Z if leading zeros of a numeric field are to be suppressed and the sign is to be stripped from the rightmost position. Leave blank if leading zeros are not suppressed, or if field specified in Field Name is alphanumeric or if line contains a constant or an edit word.

Blank After 39

Enter B to reset alphanumeric output fields to blanks or numeric output fields to zeros. Reset occurs after execution of the specified output operation.

End Position in Output Record 40 - 43

Enter position in output record to contain rightmost character of output field.

Packed Field 44

Enter P if output data in packed-decimal format. Leave blank if data in standard format.

Constant or Edit Word 45 - 70

Constant. Enter any required constant. Must be enclosed in apostrophe symbols. May consist of any of the 256 EBCDIC characters. Maximum length is 24 characters. Apostrophe symbols required within constants must be represented as two consecutive apostrophe symbols.

Edit Word. Enter any edit word to specify editing with respect to punctuation, printing of dollar symbols, sign status, zero suppression, etc. Edit words must be enclosed in apostrophe symbols.

Sterling Sign Position 71 - 74.

Provided for programs processing sterling currency amounts. Leave blank for all other RPG programs.

APPENDIX G. DIAGNOSTIC MESSAGES

IES000I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES001I FILE TYPE (COLUMN 15) IS INVALID. SPECIFICATION IS NOT PROCESSED.

IES002I INVALID ENTRY IN COLUMNS 28, 31, OR 32. SPECIFICATION IS NOT PROCESSED.

IES003I RECORD ADDRESS FIELD (COLUMNS 29-30) HAS INVALID LENGTH, IS MISSING OR IS NOT RIGHT-JUSTIFIED. ENTRY OF 03 IS ASSUMED.

IES004I MORE THAN ONE RECORD ADDRESS FILE IS PRESENT. SUCCEEDING ONES ARE NOT PROCESSED.

IES005I EXTENSION CODE (COLUMN 39) IS INVALID. ENTRY OF L IS ASSUMED.

IES006I INPUT FILE DESIGNATION (COLUMN 16) IS INVALID OR MISSING. ENTRY OF S IS ASSUMED.

IES007I OVERFLOW INDICATOR (COLUMN 33) IS NOT 0. ENTRY OF 0 IS ASSUMED.

IES008I OVERFLOW INDICATOR (COLUMNS 33-34) IS INVALID. ENTRY OF 0A IS ASSUMED.

IES009I MORE THAN ONE PRIMARY FILE IS SPECIFIED. FILE IS ASSUMED TO BE A SECONDARY FILE.

IES010I MODE OF PROCESSING (COLUMN 28) IS INVALID. ENTRY OF R IS ASSUMED.

IES011I FIXED FORMAT IS SPECIFIED, YFT BLOCK LENGTH IS NOT A MULTIPLE OF RECORD LENGTH. BLOCK LENGTH IS INCREASED TO NEXT HIGHEST MULTIPLE.

IES012I TYPE OF FILE ORGANIZATION (COLUMN 32) IS NOT BLANK. ENTRY OF BLANK IS ASSUMED.

IES013I END-OF-FILE CODE (COLUMN 17) IS INVALID. ENTRY OF BLANK IS ASSUMED.

IES014I SEQUENCE (COLUMN 18) IS INVALID. ENTRY OF BLANK IS ASSUMED.

IES015I MODE OF PROCESSING (COLUMN 28) IS NOT BLANK. ENTRY OF BLANK IS ASSUMED.

IES016I RECORD ADDRESS TYPE (COLUMN 31) IS NOT BLANK. ENTRY OF BLANK IS ASSUMED.

IES017I EXTENSION CODE (COLUMN 39) IS INVALID. ENTRY OF E IS ASSUMED.

IES018I FILE FORMAT (COLUMN 19) IS INVALID. ENTRY OF F IS ASSUMED FOR INDEXED-SEQUENTIAL. OTHERWISE ENTRY OF V IS ASSUMED.

IES019I BLOCK LENGTH (COLUMNS 20-23) IS MISSING, INVALID, LESS THAN THE RECORD LENGTH, OR NOT RIGHT-JUSTIFIED. BLOCK LENGTH IS ASSUMED EQUAL TO RECORD LENGTH.

IES020I RECORD LENGTH (COLUMNS 24-27) IS MISSING, INVALID, OR NOT RIGHT-JUSTIFIED. RECORD LENGTH OF 0080 IS ASSUMED.

IES021I FILENAME (COLUMNS 7-14) IS MISSING, INVALID, OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.

IES022I OUTPUT FILE DESIGNATION (COLUMN 16) IS NOT BLANK. ENTRY OF BLANK IS ASSUMED.

IES023I PROGRAM EXCEEDS LIMIT OF TEN VALID FILE NAMES. ADDITIONAL FILE DESCRIPTION SPECIFICATIONS WILL BE TREATED AS COMMENTS.

IES024I KEY FIELD STARTING LOCATION (COLUMNS 35-38) IS INVALID, NOT RIGHT-JUSTIFIED, OR NOT LESS THAN RECORD LENGTH. ENTRY OF 0001 IS ASSUMED.

IES025I DEVICE (COLUMNS 40-46) IS INVALID. SPECIFICATION IS NOT PROCESSED.

IES026I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES027I 'LABELS' (COLUMN 53) IS INVALID. ENTRY OF S IS ASSUMED.

IES028I NAME OF LABEL EXIT (COLUMNS 54-59) IS MISSING, INVALID, OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.

IES029I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES030I MORE THAN ONE RECORD ADDRESS FILE IS SPECIFIED ON FILE EXTENSION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.

IES031I 'FROM FILENAME' (COLUMNS 11-18) IS NOT SPECIFIED AS ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.

IES032I EXTENSION CODE (COLUMN 39) IS NOT E. SPECIFICATION IS NOT PROCESSED.

IES033I LENGTH OF TABLE ENTRY (COLUMNS 40-42 OR 52-54) EXCEEDS 256 CHARACTERS FOR AN ALPHAMERIC FIELD. ENTRY OF 256 IS ASSUMED.

IES034I CHAINING FIELD (COLUMNS 9-10) IS MISSING, INVALID, OR NOT RIGHT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.

IES035I NUMBER OF CONVERSION ROUTINES AND TABLE NAMES EXCEEDS ALLOCATED CORE STORAGE. ADDITIONAL SPECIFICATIONS CONTAINING CONVERSION ROUTINES OR TABLE NAMES WILL NOT BE PROCESSED.

IES036I *TO FILENAME* (COLUMNS 19-26) IS NOT SPECIFIED AS ON CORRESPONDING FILE DESCRIPTION SPECIFICATION, OR FILE DESCRIPTION SPECIFICATION WAS NOT PROCESSED. SPECIFICATION IS NOT PROCESSED.

IES037I *TO FILENAME* (COLUMNS 19-26) IS NOT SPECIFIED AS A CHAINED FILE ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION IS NOT PROCESSED.

IES038I LENGTH OF TABLE ENTRY (COLUMNS 40-42 OR 52-54) EXCEEDS 15 DIGITS FOR A NUMERIC FIELD. ENTRY OF 15 IS ASSUMED.

IES039I CONVERSION ROUTINE (COLUMNS 27-32) IS MISSING, INVALID, OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.

IES040I *TO FILENAME* (COLUMNS 19-26) IS NOT SPECIFIED AS A PRIMARY OR SECONDARY FILE OR *MODE OF PROCESSING* IS NOT BETWEEN LIMITS OR RANDOM. SPECIFICATION IS NOT PROCESSED.

IES041I TABLE SEQUENCE (COLUMNS 45 OR 57) IS INVALID. ENTRY OF BLANK IS ASSUMED.

IES042I TABLE NAME (COLUMNS 27-32 OR 46-51) IS MULTI-DEFINED. SPECIFICATION IS NOT PROCESSED.

IES043I *TO FILENAME* (COLUMNS 19-26) IS NOT SPECIFIED AS ON FILE DESCRIPTION SPECIFICATION. ENTRY OF BLANKS IS ASSUMED.

IES044I *TO FILENAME* (COLUMNS 19-26) IS NOT SPECIFIED AS AN OUTPUT FILE ON FILE DESCRIPTION SPECIFICATION. ENTRY OF BLANKS IS ASSUMED.

IES045I TABLE NAME (COLUMNS 27-32 OR 46-51) IS MISSING OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.

IES046I LETTERS *TAB* ARE MISSING IN TABLE NAME (COLUMNS 27-29 OR 46-48). ENTRY OF *TAB* IS ASSUMED.

IES047I NUMBER OF TABLE ENTRIES PER RECORD (COLUMNS 33-35) IS MISSING, INVALID OR NOT RIGHT-JUSTIFIED. ENTRY OF 008 IS ASSUMED.

IES048I NUMBER OF TABLE ENTRIES PER TABLE (COLUMNS 36-39) IS MISSING, INVALID OR NOT RIGHT-JUSTIFIED. ENTRY OF 0150 IS ASSUMED.

IES049I *LENGTH OF TABLE* ENTRY (COLUMNS 40-42 OR 52-54) IS MISSING, INVALID, OR NOT RIGHT-JUSTIFIED. ENTRY OF 010 IS ASSUMED.

IES050I *PACKED* (COLUMN 43 OR 55) IS INVALID. ENTRY OF BLANK IS ASSUMED.

IES051I *DECIMAL POSITIONS* (COLUMN 44 OR 56) IS INVALID. ENTRY OF ZERO IS ASSUMED.

IES052I RECORD SEQUENCE OF THE CHAINING FILE (COLUMNS 7-8) IS INVALID. BOTH POSITIONS MUST BE EITHER NUMERIC OR ALPHABETIC. SPECIFICATION IS NOT PROCESSED.

IES053I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES054I MAXIMUM NUMBER OF FILE EXTENSION SPECIFICATIONS CONTAINING TABLES HAS BEEN EXCEEDED. ADDITIONAL TABLE FILE EXTENSION SPECIFICATIONS WILL BE TREATED AS COMMENTS.

IES055I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES056I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES057I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES058I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES059I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES060I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES061I WARNING - MULTI-FILE PROGRAM (WITH PRIMARY AND SECONDARY FILES) IS SPECIFIED WITHOUT MATCHING FIELDS FOR THE PRIMARY FILE.

IES062I WARNING - MULTI-FILE PROGRAM (WITH PRIMARY AND SECONDARY FILES) IS SPECIFIED WITHOUT MATCHING FIELDS FOR THE SECONDARY FILE(S).
 IES063I THE SUM OF THE LENGTHS OF THE MATCHING FIELDS FOR THE PRIMARY FILE DOES NOT EQUAL THAT OF EACH SECONDARY FILE. EXECUTION IS DELETED.
 IES064I 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.
 IES065I WARNING- PAGE/LINE ENTRY IS OUT OF SEQUENCE.
 IES066I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES067I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES068I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES069I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES070I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES071I DETAIL CALCULATION SPECIFICATION FOLLOWS A TOTAL CALCULATION SPECIFICATION. DETAIL SPECIFICATION IS NOT PROCESSED.
 IES072I UNDEFINED TABLE SPECIFIED IN LOKUP OPERATION. SPECIFICATION IS NOT PROCESSED.
 IES073I KEYCV IS VALID ONLY WHEN PRECEDED BY RPGCV OR EXTCV. SPECIFICATION IS NOT PROCESSED.
 IES074I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES075I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES076I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES077I THERE ARE NO VALID INPUT SPECIFICATIONS IN THIS PROGRAM. EXECUTION IS DELETED.
 IES078I DECIMAL POSITION IS INVALID. ENTRY OF ZERO IS ASSUMED FOR NUMERIC FIELD. ENTRY OF BLANK IS ASSUMED FOR ALPHAMERIC FIELD.
 IES079I CONVERSION NAME CANNOT BE USED TO DEFINE A FIELD. SPECIFICATION IS NOT PROCESSED.
 IES080I FIELD INDICATOR IS SPECIFIED BUT IS NOT VALID. INDICATOR IS NOT PROCESSED.
 IES081I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES082I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES083I 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 3 IS ASSUMED FOR EXTCV AND RPGCV. ENTRY OF 4 IS ASSUMED FOR ALL OTHER OPERATION CODES.
 IES084I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES085I RESULT FIELD LENGTH (COLUMNS 49-51) IS GREATER THAN ALLOWED. A LENGTH OF 256 IS ASSUMED FOR AN ALPHAMERIC FIELD. A LENGTH OF 15 IS ASSUMED FOR A NUMERIC FIELD.
 IES086I OPERATION CODE (COLUMNS 28-32) IS INVALID OR MISSING. SPECIFICATION IS NOT PROCESSED.
 IES087I REQUIRED ENTRY IN FACTOR 1 (COLUMNS 18-27) IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
 IES088I REQUIRED ENTRY IN FACTOR 2 (COLUMNS 33-42) IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
 IES089I REQUIRED ENTRY IN RESULT FIELD (COLUMNS 43-48) IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
 IES090I FORM TYPE (COLUMN 6) IS INVALID. SPECIFICATION IS NOT PROCESSED.
 IES091I 'NOT' (COLUMNS 9, 12, OR 15) IS NOT N OR BLANK. ENTRY OF N IS ASSUMED.
 IES092I CONTROL LEVEL IS IMPROPERLY SPECIFIED. ENTRY OF LO IS ASSUMED.
 IES093I RESULTING INDICATOR IS INVALID. INDICATOR IS NOT PROCESSED.

IES094I INDICATOR LO IS SPECIFIED AS A FIELD INDICATOR, BUT IS NOT ALLOWED. INDICATOR IS IGNORED.

IES095I FIELD-RECORD RELATION (COLUMNS 63-64) IS INVALID. ENTRY OF 99 IS ASSUMED.

IES096I 'HALF ADJUST' ENTRY (COLUMN 53) IS INVALID. ENTRY OF H IS ASSUMED.

IES097I FIELD NAME IS IMPROPERLY USED. SPECIFICATION IS NOT PROCESSED.

IES098I INDICATOR (COLUMNS 10-11, 13-14, OR 16-17) IS INVALID. ENTRY OF 00 IS ASSUMED.

IES099I REQUIRED RESULTING INDICATOR (COLUMNS 54-55, 56-57, OR 58-59) IS NOT SPECIFIED. SPECIFICATION IS NOT PROCESSED.

IES100I 'MVR' DOES NOT FOLLOW 'DIV', OR FOLLOWS A 'DIV' WITH HALF ADJUST SPECIFIED. SPECIFICATION IS NOT PROCESSED.

IES101I 'FROM' (COLUMNS 44-47), 'TO' (COLUMNS 48-51) OR RECORD IDENTIFICATION POSITION (COLUMNS 21-24, 28-31, OR 35-38) IS ZERO. ENTRY OF 1 IS ASSUMED.

IES102I RESULT FIELD OF RPGCV OR EXTCV DOES NOT HAVE LENGTH OF 3. ENTRY OF 3 IS ASSUMED.

IES103I IF IBM SHILLING IS SPECIFIED, STERLING INPUT FIELD MUST HAVE MORE THAN THREE NON-DECIMAL POSITIONS. IF BSI SHILLING IS SPECIFIED, STERLING INPUT FIELD MUST HAVE MORE THAN TWO NON-DECIMAL POSITIONS. FIELD IS INITIALIZED TO ZERO AND NO INPUT IS ACCEPTED.

IES104I WARNING - INDICATOR 00 SHOULD BE USED ONLY IN OUTPUT SPECIFICATIONS.

IES105I FIELDS USED IN AN ALPHAMERIC COMPARE MUST BE EQUAL IN LENGTH OR MUST BE LESS THAN OR EQUAL TO 200 BYTES.

IES106I FIELD LENGTHS ARE INVALID FOR THIS OPERATION. SPECIFICATION IS NOT PROCESSED.

IES107I PLUS AND/OR MINUS RESULTING INDICATORS (COLUMNS 54-55 OR 56-57) ARE NOT ALLOWED FOR TESTING ALPHAMERIC FIELDS. INDICATORS ARE IGNORED.

IES108I FIELD TYPE IS INVALID FOR THIS OPERATION. SPECIFICATION IS NOT PROCESSED.

IES109I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES110I FORM TYPE (COLUMN 6) DOES NOT CONTAIN I, C, OR O, AND COLUMN 7 IS NOT AN ASTERISK. SPECIFICATION IS NOT PROCESSED.

IES111I UNDEFINED FILENAME. SPECIFICATION IS NOT PROCESSED.

IES112I FILENAME HAS BEEN PREVIOUSLY REFERENCED OR DEFINED AS A TABLE OR OUTPUT FILE TYPE. SPECIFICATION IS NOT PROCESSED.

IES113I 'AND' SPECIFICATION (COLUMNS 14-16) IS FIRST INPUT SPECIFICATION OR FOLLOWS FIELD DESCRIPTION SPECIFICATION, INVALID RECORD IDENTIFICATION, OR INVALID FILE NAME. SPECIFICATION IS NOT PROCESSED.

IES114I THERE ARE NO RECORD IDENTIFICATION CODES (COLUMNS 21-41) IN THE CARD BEFORE AN 'AND' CARD. SPECIFICATION IS NOT PROCESSED.

IES115I 'OR' SPECIFICATION (COLUMNS 14-15) IS FIRST INPUT SPECIFICATION OR FOLLOWS FIELD DESCRIPTION SPECIFICATION, INVALID RECORD IDENTIFICATION, OR INVALID FILE NAME. SPECIFICATION IS NOT PROCESSED.

IES116I RECORD IDENTIFICATION SPECIFICATION FOLLOWS INVALID FILE TYPE SPECIFICATION. SPECIFICATION IS NOT PROCESSED.

IES117I FIELD NAME CONTAINS EMBEDDED BLANK. SPECIFICATION IS NOT PROCESSED.

IES118I FILE AND FIELD NAMES ARE BOTH PRESENT ON THE SAME SPECIFICATION. FILENAME IS ASSUMED.

IES119I SEQUENCE (COLUMNS 15-16) IS BLANK. ENTRY OF AA IS ASSUMED.

IES120I ALPHAMERIC SEQUENCE FOUND AFTER NUMERIC SEQUENCE. NUMERIC SEQUENCE 1 GREATER THAN PREVIOUS NUMERIC SEQUENCE IS ASSUMED.

IES121I NUMERIC SEQUENCE (COLUMNS 15-16) IS INVALID. ENTRIES MUST BEGIN WITH 01 AND BE CONSECUTIVE IN ASCENDING ORDER. NUMERIC SEQUENCE 1 GREATER THAN PREVIOUS VALID NUMERIC SEQUENCE IS ASSUMED.

IES122I NUMBER (COLUMN 17) IS NOT N OR 1, AND NUMERIC SEQUENCE IS FOUND. ENTRY OF N IS ASSUMED.

IES123I OPTION (COLUMN 18) IS NOT 0 OR BLANK. ENTRY OF 0 IS ASSUMED.
 IES124I RESULTING INDICATOR (COLUMNS 19-20) IS BLANK OR INVALID. INDICATOR OF 99 IS ASSUMED.
 IES125I STACKER SELECT (COLUMN 42) IS NOT BLANK OR NUMERIC. ENTRY OF BLANK IS ASSUMED.
 IES126I 'POSITION' (COLUMNS 21-24, 28-31, OR 35-38) CONTAINS EMBEDDED BLANK. ENTRY OF ZERO IS ASSUMED.
 IES127I 'POSITION' (COLUMNS 21-24, 28-31, OR 35-38) CONTAINS NON-NUMERIC CHARACTER. ENTRY OF ZERO IS ASSUMED.
 IES128I 'NOT' (COLUMNS 25, 32, OR 39) IS NOT BLANK OR N. ENTRY OF N IS ASSUMED.
 IES129I 'C/Z/D' (COLUMNS 26, 33, OR 40) IS NOT C, Z, OR D. ENTRY OF C IS ASSUMED.
 IES130I FIELD DESCRIPTION SPECIFICATION IS FIRST INPUT SPECIFICATION OR FOLLOWS AN INVALID RECORD IDENTIFICATION OR FILE NAME. SPECIFICATION IS NOT PROCESSED.
 IES131I FIELD NAME (COLUMNS 53-58) IS MISSING OR NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
 IES132I FIELD NAME (COLUMNS 53-58) BEGINS WITH A NUMERIC CHARACTER. SPECIFICATION IS NOT PROCESSED.
 IES133I 'FROM' (COLUMNS 44-47) OR 'TO' (COLUMNS 48-51) IS BLANK. ENTRY OF 1 IS ASSUMED.
 IES134I 'FROM' (COLUMNS 44-47) OR 'TO' (COLUMNS 48-51) CONTAINS EMBEDDED BLANK. ENTRY OF ZERO IS ASSUMED FOR EACH BLANK.
 IES135I 'FROM' (COLUMNS 44-47) OR 'TO' (COLUMNS 48-51) CONTAINS NON-NUMERIC CHARACTER. ENTRY OF ZERO IS ASSUMED.
 IES136I 'FROM' (COLUMNS 44-47) SHOULD BE LESS THAN OR EQUAL TO 'TO' (COLUMNS 48-51). FIELD LENGTH OF 1 IS ASSUMED. 'TO' IS ASSUMED EQUAL TO 'FROM'.
 IES137I DECIMAL POSITION (COLUMN 52) IS NOT NUMERIC. ENTRY OF ZERO IS ASSUMED.
 IES138I EITHER AN UNPACKED NUMERIC FIELD IS MORE THAN 15 BYTES LONG, OR A PACKED NUMERIC FIELD IS GREATER THAN 8 BYTES LONG. LENGTH OF 15 IS ASSUMED FOR UNPACKED NUMERIC FIELD. LENGTH OF 8 IS ASSUMED FOR PACKED NUMERIC FIELD.
 IES139I STERLING FIELD IS INDICATED WITH MORE THAN THREE DECIMAL POSITIONS. THE DECIMAL PORTION OF THE FIELD IS TRUNCATED TO THREE POSITIONS. THE 'TO' POSITION OF THE FIELD IS ALTERED TO ALLOW FOR THIS TRUNCATION.
 IES140I PACKED INDICATOR (COLUMN 43) IS NEITHER BLANK NOR P. ENTRY OF P IS ASSUMED.
 IES141I CONTROL LEVEL DOES NOT START WITH L IN COLUMN 59 (L IS ASSUMED), OR COLUMN 60 IS NOT NUMERIC (1 IS ASSUMED).
 IES142I MATCHING/CHAINING FIELD (COLUMN 61) IS NOT C OR M (M IS ASSUMED), OR COLUMN 62 IS NOT NUMERIC (1 IS ASSUMED).
 IES143I MATCHING VALUE (COLUMN 62) IS INVALID. ENTRY OF 3 IS ASSUMED IF COLUMN IS NUMERIC. ENTRY OF 1 IS ASSUMED IF COLUMN IS NON-NUMERIC.
 IES144I ALPHAMERIC FIELD LENGTH IS GREATER THAN 256. LENGTH OF 256 IS ASSUMED.
 IES145I NUMERIC RESULTING INDICATOR (COLUMNS 65-66 OR 67-68) IS SPECIFIED FOR AN ALPHAMERIC FIELD. INDICATOR IS SET OFF.
 IES146I STERLING SIGN POSITION (COLUMNS 71-74) IS INVALID. NORMAL POSITION FOR SIGN IS ASSUMED.
 IES147I STERLING SPECIFIED IN COLUMNS 71-74, BUT NOT SPECIFIED ON PROCESSOR CONTROL CARD. ENTRY OF BLANKS IS ASSUMED.
 IES148I THE CALCULATED LENGTH OF THE STERLING FIELD IS GREATER THAN 15. LENGTH OF 15 IS ASSUMED.
 IES149I FIELD EXTENDS BEYOND RECORD LENGTH. FIELD IS ASSUMED TO END IN LAST CHARACTER POSITION OF RECORD.
 IES150I COLUMNS 16-18 OF 'OR' TYPE INPUT SPECIFICATION MUST BE BLANK. ENTRY OF BLANK IS ASSUMED.
 IES151I COLUMNS 17-20 AND 42 OF 'AND' TYPE INPUT SPECIFICATION MUST BE BLANK. ENTRY OF BLANK IS ASSUMED.

IES152I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES153I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES154I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES155I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES156I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES157I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES158I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES159I INAPPROPRIATE OUTPUT FIELD. SPECIFICATION IS NOT PROCESSED.
 IES160I FILENAME (COLUMNS 7-14) IS MISSING, OR RECORD TYPE (COLUMN 15) IS IN WRONG ORDER. SPECIFICATION IS NOT PROCESSED.
 IES161I CORRESPONDING FILENAME (COLUMNS 7-14) CANNOT BE DETERMINED. SPECIFICATION IS NOT PROCESSED.
 IES162I 'STACKER SELECT' (COLUMN 16) IS INVALID. ENTRY OF BLANK IS ASSUMED.
 IES163I 'SPACE BEFORE' (COLUMN 17) IS INVALID. ENTRY OF 1 IS ASSUMED.
 IES164I 'SPACE AFTER' (COLUMN 18) IS INVALID. ENTRY OF 1 IS ASSUMED.
 IES165I 'SKIP BEFORE' (COLUMNS 19-20) IS INVALID. ENTRY OF 01 IS ASSUMED.
 IES166I 'SKIP AFTER' (COLUMNS 21-22) IS INVALID. ENTRY OF 01 IS ASSUMED.
 IES167I RECORD TYPE (COLUMN 15) IS NOT H, D, OR T. SPECIFICATION IS NOT PROCESSED.
 IES168I COLUMNS 17-22 MUST BE BLANK FOR 'AND' TYPE OUTPUT SPECIFICATIONS. ENTRY OF BLANK IS ASSUMED.
 IES169I COLUMNS 7-13 MUST BE BLANK FOR 'AND' OR 'OR' TYPE OUTPUT SPECIFICATIONS. ENTRY OF BLANK IS ASSUMED.
 IES170I CORRESPONDING RECORD IDENTIFICATION SPECIFICATION IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
 IES171I 'ZERO SUPPRESS' (COLUMN 38) IS INVALID. ENTRY OF BLANK IS ASSUMED.
 IES172I 'PACKED FIELD' (COLUMN 44) IS INVALID. ENTRY OF BLANK IS ASSUMED.
 IES173I FIELD NAME (COLUMNS 32-37) IS NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
 IES174I 'END POSITION' (COLUMNS 40-43) IS INVALID OR MISSING. SPECIFICATION IS NOT PROCESSED.
 IES175I LEADING OR CLOSING APOSTROPHE (') IN EDIT WORD IS NOT CORRECT. ENTRY OF BLANKS IN COLUMNS 45-70 IS ASSUMED.
 IES176I 'BLANK AFTER' (COLUMN 39) IS INVALID. ENTRY OF BLANK IS ASSUMED.
 IES177I PUNCI. AND PRINT FUNCTIONS ARE SPECIFIED FOR THE SAME FILE. ENTRY OF BLANKS IS ASSUMED FOR COLUMNS 17-22.
 IES178I ZERO SUPPRESSION (COLUMN 38) MAY NOT BE SPECIFIED FOR CONSTANTS OR EDIT WORDS. ENTRY OF BLANK IN COLUMN 38 IS ASSUMED.
 IES179I FIELD NAME (COLUMNS 32-37) IS UNDEFINED. SPECIFICATION IS NOT PROCESSED.
 IES180I WARNING - 'BLANK AFTER' (COLUMN 39) IS SPECIFIED FOR CONSTANT. ALL IDENTICAL CONSTANTS WILL BE BLANKED.
 IES181I CONSTANT (COLUMNS 45-70) IS NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.
 IES182I EDIT WORD (COLUMNS 45-70) IS NOT LEFT-JUSTIFIED. ENTRY OF BLANKS IN COLUMNS 45-70 IS ASSUMED.
 IES183I 'PACKED FIELD' (COLUMN 44) MAY NOT BE SPECIFIED WITH CONSTANT, EDIT WORD OR STERLING ENTRY. ENTRY OF BLANK IN COLUMN 44 IS ASSUMED.
 IES184I FILENAME (COLUMNS 7-14) IS NOT LEFT-JUSTIFIED. SPECIFICATION IS NOT PROCESSED.

IES185I FILENAME (COLUMNS 7-14) CONTAINS NON-ALPHABETIC CHARACTER IN FIRST POSITION.
 SPECIFICATION IS NOT PROCESSED.

IES186I EDIT WORD (COLUMNS 45-70) CONTAINS NO DIGIT POSITIONS OR MORE THAN FIFTEEN
 (SIXTEEN FOR STERLING). ENTRY OF BLANKS IN COLUMNS 45-70 IS ASSUMED.

IES187I LEADING OR CLOSING APOSTROPHE (') IN CONSTANT IS NOT CORRECT. SPECIFICATION IS
 NOT PROCESSED.

IES188I 'AND' OR 'OR' FOLLOWING A FIELD NAME SPECIFICATION OR AS FIRST OUTPUT
 SPECIFICATION IS INVALID. SPECIFICATION IS NOT PROCESSED.

IES189I FIELD NAME (COLUMNS 32-38) CONTAINS NON-ALPHABETIC CHARACTER IN FIRST POSITION.
 SPECIFICATION IS NOT PROCESSED.

IES190I STERLING ENTRY (COLUMNS 71-74) MAY NOT BE SPECIFIED WITH CONSTANT OR PAGE(N).
 ENTRY OF BLANK IN COLUMNS 71-74 IS ASSUMED.

IES191I STERLING ENTRY (COLUMNS 71-74) IS INVALID. ENTRY OF BLANKS IS ASSUMED.

IES192I OUTPUT INDICATOR (COLUMNS 24-25, 27-28, OR 30-31) IS INVALID OR UNDEFINED.
 ENTRY OF LO IS ASSUMED.

IES193I OUTPUT INDICATORS SHOULD START IN COLUMNS 23-25, THEN 26-28, AND FINALLY 29-31.
 ENTRY IS SHIFTED LEFT.

IES194I 'NOT' (COLUMNS 23, 26, OR 29) IS NOT BLANK OR N. ENTRY OF N IS ASSUMED.

IES195I WARNING - OVERFLOW INDICATOR IS SPECIFIED IN 'AND' TYPE SPECIFICATION. RECORD
 WILL NOT BE PUT OUT AS OVERFLOW LINE.

IES196I DECIMAL POSITIONS MUST BE ZERO FOR PAGE(N) FIELD. ENTRY OF ZERO IS ASSUMED.

IES197I SPECIFICATION TYPE CANNOT BE DETERMINED. RECORD AND FIELD DEFINITIONS ARE
 SPECIFIED IN SAME LINE OR BOTH ARE BLANK. SPECIFICATION IS NOT PROCESSED.

IES198I FORM TYPE (COLUMN 6) IS INVALID (NOT 0). SPECIFICATION IS NOT PROCESSED.

IES199I NO OUTPUT INDICATOR (COLUMNS 24-25, 27-28, OR 30-31) IS SPECIFIED FOR 'AND' OR
 'OR' TYPE SPECIFICATION. SPECIFICATION IS NOT PROCESSED.

IES200I FORM TYPE (COLUMN 6) IS INVALID OR OUT OF SEQUENCE. SPECIFICATION IS NOT
 PROCESSED.

IES201I FILE DESCRIPTION SPECIFICATIONS ARE MISSING. COMPILATION IS ABORTED.

IES202I WARNING - LINE COUNTER SPECIFICATION IS MISSING.

IES203I PRIMARY FILE IS NOT SPECIFIED. EXECUTION IS DELETED.

IES204I WARNING - FILE EXTENSION SPECIFICATION IS MISSING.

IES205I FILENAME (COLUMNS 7-14) IS MULTI-DEFINED. SPECIFICATION IS NOT PROCESSED.

IES206I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES207I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES208I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES209I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.

IES210I WARNING - FILENAME (COLUMNS 7-14) IS DEFINED BUT NEVER USED.

IES211I FILENAME HAS BEEN REFERENCED PREVIOUSLY IN INPUT SPECIFICATIONS. SPECIFICATION
 IS NOT PROCESSED.

IES212I RESULTING INDICATOR IS INVALID OR UNDEFINED. ENTRY OF LO IS ASSUMED.

IES213I WARNING - RESULTING INDICATOR IS UNREFERENCED.

IES214I FIELD NAME IS UNDEFINED. FIELD IS PROCESSED WITH ASSUMED LENGTH OF 004.

IES215I WARNING - FIELD NAME IS MULTI-DEFINED.

IES216I WARNING - FIELD NAME IS UNREFERENCED.

IES217I THE COMBINED LENGTHS OF LITERALS AND FIELD NAMES EXCEEDS ALLOCATED CORE STORAGE.
 EXECUTION IS DELETED.

IES218I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES219I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES220I FILE SPECIFIED ON OUTPUT-FORMAT SPECIFICATIONS IS UNDEFINED OR NOT AN OUTPUT
 FILE (U, C, OR O). ENTIRE FILE IS DELETED FROM PROCESSING.
 IES221I WARNING - FILENAME (COLUMNS 7-14) IS NOT REFERENCED ON OUTPUT SPECIFICATIONS.
 IES222I NO VALID OUTPUT SPECIFICATIONS ARE PRESENT. COMPILATION IS ABORTED.
 IES223I ALL OUTPUT LINES OF A PRINTER FILE MUST INDICATE SPACING AND/OR SKIPPING.
 SINGLE SPACING IS ASSUMED FOR ALL OUTPUT LINES OF NAMED FILE WHICH HAVE NO
 PRINT FUNCTION.
 IES224I STACKER SELECT MAY NOT BE SPECIFIED WITH PRINT FILE. STACKER SELECT IS IGNORED
 AND SINGLE SPACING IS ASSUMED FOR ALL LINES OF NAMED FILE.
 IES225I PRINT OR PUNCH FUNCTION MAY NOT BE SPECIFIED FOR AN OUTPUT RECORD OF TAPE OR
 DISK FILE. STACKER SELECT, SPACING, OR SKIPPING IS IGNORED ON ALL RECORDS OF
 NAMED FILE.
 IES226I PRINT FUNCTION MAY NOT BE SPECIFIED FOR OUTPUT RECORD OF PUNCH FILE. SPACE AND
 SKIP ENTRIES ARE IGNORED FOR ALL RECORDS OF NAMED FILE.
 IES227I IMPROPER USE OF PACKING OR ZERO SUPPRESSION ON ALPHAMERIC OR PACKED FIELD.
 ENTRY OF BLANK IS ASSUMED FOR INVALID CODE.
 IES228I END POSITION SPECIFIED FOR THE FIELD IS GREATER THAN THE RECORD LENGTH. ALL OR
 PART OF THE FIELD IS LOST, STARTING FROM RIGHTMOST POSITION.
 IES229I END POSITION IS LESS THAN THE FIELD LENGTH. FIELD IS NOT PROCESSED.
 IES230I FIELD TO BE EDITED IS GREATER THAN THE EDIT WORD. RIGHTMOST DIGITS WILL BE
 LOST.
 IES231I FIELD TO BE EDITED IS NOT NUMERIC. NO EDITING IS PERFORMED.
 IES232I STERLING IS SPECIFIED FOR ALPHAMERIC FIELD. STERLING IS IGNORED.
 IES233I STERLING SIGN POSITION IS SPECIFIED AS OTHER THAN NORMAL FOR PRINTED LINE.
 NORMAL POSITION IS ASSUMED.
 IES234I LOCATION FOR STERLING SIGN EXCEEDS RECORD LENGTH. NORMAL POSITION FOR SIGN IS
 ASSUMED.
 IES235I STERLING FIELD IS SPECIFIED WITH DECIMAL LENGTH GREATER THAN THREE. FIELD IS
 NOT PROCESSED.
 IES236I STERLING FIELDS MAY BE EDITED ONLY FOR PRINT FILES. NO EDITING IS PERFORMED
 IES237I NUMBER OF LINES OF OUTPUT EXCEEDS THE CAPACITY OF RPG. MAXIMUM NUMBER IS 1023.
 EXECUTION IS DELETED.
 IES238I STERLING FIELD IS SPECIFIED WITH MORE THAN NINE POUNDS POSITIONS. LEFTMOST
 DIGITS WILL BE LOST.
 IES239I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES240I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES241I FILENAME IS NOT SPECIFIED AS ON FILE DESCRIPTION SPECIFICATION. SPECIFICATION
 IS NOT PROCESSED.
 IES242I CHANNEL 1 OR 12 IS MISSING OR INVALID. SPECIFICATION IS NOT PROCESSED.
 IES243I FILENAME IS NOT SPECIFIED AS AN OUTPUT FILE OR AN OUTPUT FILE REQUIRING A LINE
 COUNTER SPECIFICATION. SPECIFICATION IS NOT PROCESSED.
 IES244I LINE NUMBER OR CHANNEL NUMBER IS INVALID OR MISSING. SPECIFICATION IS NOT
 PROCESSED.
 IES245I CHANNEL NUMBER IS MULTI-DEFINED. SPECIFICATION IS NOT PROCESSED.
 IES246I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES247I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
 IES248I COLLATING SEQUENCE (COLUMN 26 OF RPG CONTROL CARD) IS INVALID. ALTERNATE
 SEQUENCE IS ASSUMED.

IES249I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
IES250I END OF FILE HAS BEEN ENCOUNTERED IN INPUT STREAM PRIOR TO AN OUTPUT
SPECIFICATION. COMPILATION IS ABORTED.
IES251I NO ERROR MESSAGE ASSIGNED FOR THIS NOTE.
IES252I STERLING INPUT SPECIFICATION IS INVALID. IBM FORMAT IS ASSUMED.
IES253I STERLING OUTPUT SPECIFICATION IS INVALID. IBM FORMAT IS ASSUMED.
IES254I INVERTED PRINT ENTRY IS INVALID. ENTRY OF I IS ASSUMED.
IES255I RPG CONTROL CARD IS MISSING. COMPILATION IS ABORTED.

APPENDIX H. CONDITIONS THAT AUTOMATICALLY TURN ON HALT INDICATOR (H0)

The object program:

- Read on input record that was not defined on the Input Specifications sheet (columns 21-41).
- Found on input record out of the predetermined sequence of card types specified by the entry in Sequence (columns 15-16) on the Input Specifications sheet.
- Found on input record out of sequence when the entry in Matching Fields (columns 61-62) on the Input Specifications sheet was used for sequence checking a single input file.
- Encountered a chaining field in the chaining file that does not appear in the chained file during random processing of multiple input files.
- Did not find a record with the correct key at the designated track address during random processing by record key of a directly organized file.
- Did not find the record key that designates the lower limit (obtained from the RAF) during sequential processing between limits of an indexed-sequential file.
- Found a wrong length record during processing of an indexed-sequential file.
- Found an invalid length record (zero or too long) during random processing by record identification of a file on a DASD.
- Found a difference between the key length of a DASD record in an indexed-sequential file and the length as specified in Length of Record Address Field (columns 29-30) on the File Description Specifications sheet during processing with RAF support (random, or between limits).
- Found a difference between the key length in the chained indexed-sequential file and the length as specified (columns 44-51) on the Input Specifications sheet during chaining of multiple input files.
- Encountered a data check on the DASD during random processing of a directly organized file.
- Encountered a DASD error during sequential or random processing of an indexed-sequential file.
- Found a record identification outside the defined limits during random processing of a directly organized file.
- Did not find keys in a directly organized file when the entry in Record Address Type (column 31) on the File Description Specifications sheet specified processing by key.
- Did not find a record with the specified key during random processing of an indexed-sequential file.
- Encountered an irrecoverable input/output error during processing of a sequential file.
- Encountered an irrecoverable input/output error during processing of a combined file.

NOTE: Unless the H0 indicator is turned off by a SETOF Operation entry on the Calculation Specifications sheet (see Turning Indicators On or Off) the program terminates before the next input record is read.

APPENDIX I. RPG COMPILER RETURN CODES

| NOTE NUMBER | RETURN CODE |
|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| 000 | NA | 064 | 12 | 128 | 4 | 192 | 4 |
| 001 | 8 | 065 | 4 | 129 | 4 | 193 | 4 |
| 002 | 8 | 066 | NA | 130 | 8 | 194 | 4 |
| 003 | 4 | 067 | NA | 131 | 8 | 195 | 4 |
| 004 | 8 | 068 | NA | 132 | 8 | 196 | 4 |
| 005 | 4 | 069 | NA | 133 | 4 | 197 | 8 |
| 006 | 4 | 070 | NA | 134 | 4 | 198 | 8 |
| 007 | 4 | 071 | 8 | 135 | 4 | 199 | 8 |
| 008 | 4 | 072 | 8 | 136 | 8 | 200 | 8 |
| 009 | 4 | 073 | 8 | 137 | 4 | 201 | 16 |
| 010 | 4 | 074 | NA | 138 | 4 | 202 | 4 |
| 011 | 4 | 075 | NA | 139 | 4 | 203 | 12 |
| 012 | 4 | 076 | NA | 140 | 4 | 204 | 8 |
| 013 | 4 | 077 | 12 | 141 | 4 | 205 | 8 |
| 014 | 4 | 078 | 4 | 142 | 4 | 206 | NA |
| 015 | 4 | 079 | 8 | 143 | 4 | 207 | NA |
| 016 | 4 | 080 | 8 | 144 | 4 | 208 | NA |
| 017 | 4 | 081 | NA | 145 | 4 | 209 | NA |
| 018 | 4 | 082 | NA | 146 | 4 | 210 | 4 |
| 019 | 4 | 083 | 4 | 147 | 4 | 211 | 8 |
| 020 | 4 | 084 | NA | 148 | 4 | 212 | 4 |
| 021 | 8 | 085 | 4 | 149 | 8 | 213 | 4 |
| 022 | 4 | 086 | 8 | 150 | 4 | 214 | 4 |
| 023 | 4 | 087 | 8 | 151 | 4 | 215 | 4 |
| 024 | 4 | 088 | 8 | 152 | NA | 216 | 4 |
| 025 | 8 | 089 | 8 | 153 | NA | 217 | 12 |
| 026 | NA | 090 | 8 | 154 | NA | 218 | NA |
| 027 | 4 | 091 | 4 | 155 | NA | 219 | NA |
| 028 | 8 | 092 | 4 | 156 | NA | 220 | 8 |
| 029 | NA | 093 | 4 | 157 | NA | 221 | 4 |
| 030 | 8 | 094 | 4 | 158 | NA | 222 | 16 |
| 031 | 8 | 095 | 4 | 159 | 8 | 223 | 4 |
| 032 | 8 | 096 | 4 | 160 | 8 | 224 | 4 |
| 033 | 4 | 097 | 8 | 161 | 8 | 225 | 4 |
| 034 | 8 | 098 | 4 | 162 | 4 | 226 | 4 |
| 035 | 4 | 099 | 8 | 163 | 4 | 227 | 4 |
| 036 | 8 | 100 | 8 | 164 | 4 | 228 | 4 |
| 037 | 8 | 101 | 4 | 165 | 4 | 229 | 8 |
| 038 | 4 | 102 | 4 | 166 | 4 | 230 | 8 |
| 039 | 8 | 103 | 4 | 167 | 8 | 231 | 4 |
| 040 | 8 | 104 | 4 | 168 | 4 | 232 | 4 |
| 041 | 4 | 105 | 4 | 169 | 4 | 233 | 4 |
| 042 | 8 | 106 | 8 | 170 | 8 | 234 | 4 |
| 043 | 4 | 107 | 4 | 171 | 4 | 235 | 8 |
| 044 | 4 | 108 | 8 | 172 | 4 | 236 | 4 |
| 045 | 8 | 109 | NA | 173 | 8 | 237 | 12 |
| 046 | 4 | 110 | 8 | 174 | 8 | 238 | 8 |
| 047 | 4 | 111 | 8 | 175 | 4 | 239 | NA |
| 048 | 4 | 112 | 8 | 176 | 4 | 240 | NA |
| 049 | 4 | 113 | 8 | 177 | 4 | 241 | 8 |
| 050 | 4 | 114 | 8 | 178 | 4 | 242 | 8 |
| 051 | 4 | 115 | 8 | 179 | 8 | 243 | 8 |
| 052 | 8 | 116 | 8 | 180 | 4 | 244 | 8 |
| 053 | NA | 117 | 8 | 181 | 8 | 245 | 8 |
| 054 | 4 | 118 | 4 | 182 | 4 | 246 | NA |
| 055 | NA | 119 | 4 | 183 | 4 | 247 | NA |
| 056 | NA | 120 | 4 | 184 | 8 | 248 | 4 |
| 057 | NA | 121 | 4 | 185 | 8 | 249 | NA |
| 058 | NA | 122 | 4 | 186 | 4 | 250 | 16 |
| 059 | NA | 123 | 4 | 187 | 8 | 251 | NA |
| 060 | NA | 124 | 4 | 188 | 8 | 252 | 4 |
| 061 | 4 | 125 | 4 | 189 | 8 | 253 | 4 |
| 062 | 4 | 126 | 4 | 190 | 4 | 254 | 4 |
| 063 | 12 | 127 | 4 | 191 | 4 | 255 | 16 |

INDEX

- Absolute compare routine 64, 65
- ADD 59, 69
- Add, operation 59, 69
- Addition and Subtraction 6
- Alphabetic characters in RPG 31, 36
- Alphameric field length 42
- Alphameric literals, forming 58
- Alphameric literals, output 86
- Alternate collating sequence 51, 148
- Alternating arguments and functions 101, 104, 109
- Ampersand, & 39
- Ampersand, edit word 87
- AND relationship 40, 79, 178, 181
- Apostrophe symbol 58
- Argument 102, 104
- Arithmetic Operations 59, 72, 181
- Asterisk protection 88
- At sign, @ 31, 36
- Automatic skipping 77, 89

- Blank After 11, 82, 182
- Blank after, effect of 53, 72
- Block Length 93, 175
- Blocking records 116
- Body, edit word 86
- Branching and Exit operations 59, 64
- Branching or GOTO 59, 66, 69, 114.1
- BSI format 148, 173

- C/Z/D 5, 39, 179
- Calculations Specifications sheet 55, 180
- Calculations, when performed 55
- Cataloged procedures 144
- Chaining 131
- Chaining Fields 51, 179
- Chaining fields, split 133
- Channel Number (1) (Line Counter) 89, 178
- Character 5, 40, 179
- Checking Sequence 24
- Checking sequence of input files 49, 92
- Collating sequence, alternate 51, 148
- Combined file 91
- Common fields 33, 175
- Comments 72, 98, 102, 176, 178, 181
- Comments, * column 7 34, 175
- COMP 59, 63, 69
- Compare — Equal 63, 181
- Compare — High 63, 181
- Compare — Low 63, 181
- Compare Operations 59, 63, 69, 72, 181
- Comparison of two fields 21
- Compatability 1
- Compiler options 139
- Compiler output 140
- Compiler processing 138
- Compiler return codes 141, 193
- CONTD 123
- Control card 148, 172
- Control Fields 8
- Control Level — Calculation 55, 180
- Control Level — Input 8, 47, 179
- Control-field holding area 48

- Constant or Edit Word 86, 182
- Constants 86
- Conversion of chaining fields 134
- Conversion Operation codes 123
- Conversion routine, RAF 123
- Conversion routine operation codes 174
- Conversion routine operations 59
- CR symbol, edit word 87
- Creating record-address files 124
- Creating table records 105
- Cross-References
 - Appendix D 172
 - Appendix H 192
 - Cataloged procedures 144
 - Chaining 131
 - Creating record-address files 124
 - Disk storage concepts 115
 - Executing RPG-input stream variations 149
 - Exit to a subroutine 111
 - Field location 42
 - Halt Indicators 52
 - Load module execution 142
 - Matching fields 49
 - Output-Format Specifications sheet 75
 - Overriding cataloged procedures 146
 - Problem definition 29
 - Processing multiple input files 127
 - Processing single input files 120
 - Records in an OR relationship 44
 - RPG Specifications sheets 31
 - Sample program one 152
 - Sample program two 158
 - Turning indicators On or Off 67
 - Using cataloged procedures 146
 - Using exit routines in object module 104
 - Using tables in the object module 104
 - Using the calculation sheet 72
 - Using the matching fields specification for sequence checking 49
 - Variable-length records 116

- Data Files 115
- Deck arrangement 148
- Decimal Positions-Calculation 70, 180
- Decimal Positions-File Extension 101, 177
- Decimal Positions-Input 7, 44, 179
- Decimal symbol location 7
- Defining indicators 52, 71
- Defining the problem 29
- Describing a record 4
- Detail and total printing 10
- Detail printing 7
- Detail records 76
- Detail time 28
- Detail-to-Total Branch 114.1
- Device 96, 176
- Digit record identification 40
- Direct files 118
- Direct organization 117, 119
- Direct organization, processing 119, 122
- Disk storage concepts 115
- DIV 59, 60, 69

Divide, operation 59, 60, 69
 Division 22
 Dollar sign 31, 36, 87

 Edit words 86
 Embedded blanks, definition of 36
 End of File 92, 175
 End of RPG conversion 59, 68, 69
 End Position in Output Record 8, 82, 182
 ERPGC 59, 68, 69, 174
 Executing RPG 149
 EXIT 59, 64, 69
 Exit operations 64
 Exit to a user's subroutine 69, 104, 111
 Exit to external translate subroutine 51
 EXTCV 59, 68, 69
 Extension Code 89, 96, 176
 Extent area 115
 Extent Exit for DAM 176
 External conversion routine 59, 66, 69, 123

 Factor 1 58, 180
 Factor 2 58, 180
 Field description, input 35, 42
 Field description, output 80
 Field indicators 19, 52, 179
 Field Length 6, 70, 180
 Field length, maximum 42
 Field location 42, 179
 Field Name-Input 9, 46, 179
 Field Name-Output 8, 80, 182
 Field-Record relation 44, 48, 51, 179
 Field status conditions 52
 Fields, exit routines 112
 File Description Specifications Sheet 91, 175
 File Designation 92, 175
 File Extension Specifications Sheet 99, 177
 File Format 93, 175
 File identification and control, output 75
 File organization 116
 File processing 117
 File Processing, summary 136
 File Type 91, 175
 Filename — File Description 91, 175
 Filename — Input 91, 178
 Filename — Line counter 89, 178
 Filename — Output 13, 75, 181
 Files, maximum number of 91
 Fixed dollar sign 87
 Fixed-length records 115
 Floating dollar sign 87
 Flowcharts, logic 169
 Form Type 34, 175
 Forming tables, rules for 104
 From: Field Location 42, 179
 From Filename 100, 177
 Function, table operations 102, 104
 Function of RPG 1
 Fundamentals of RPG programming 4

 GOTO 59, 66, 69, 114.1
 Group indication 12

 Group printing 12

 Half Adjust 23, 71, 181
 Halt indicators 52, 192
 Heading records 76
 Holding area, control-field 48
 Holding area, table 68, 107, 113
 H0 indicator 36, 53, 192

 IBM format 148, 173
 ID field 123
 Indexed-sequential files 117, 120
 Indexed-sequential organization 116
 Indicator chart (Appendix B) 168
 Indicators 6, 56, 180
 Indicators, exit routine 112
 Input file 91
 Input record sequence 35
 Input Specifications sheet 35, 172, 178
 Input stream 149
 Inverted print 144, 148

 Job control language symbolic parameters 146
 Job processing 138
 Justifying field entries 31

 KEYCV 59, 68, 69, 174
 Key field 123
 Key Field Starting Location 96, 176

 Label for GOTO 66, 69
 Label options in RPG 98
 Labels 98, 176
 Last record indicator 56
 Left-justification 31
 Length of Record Address Field 95, 176
 Length of Table Entry 101, 177
 Level Zero Indicator 56
 Line 34, 175
 Line Counter Specifications sheet 89, 178
 Line-identification code 29
 Line Number (1), (Line Counter) 89, 178
 Linkage Editor processing 140
 Linkage Field, table 112
 Literal 58
 L0 indicator 56
 Load module execution 142
 Load module return codes 143
 Logic flowcharts 169
 Logic, program 28
 Logical file 115
 LOKUP 59, 67, 69, 107, 181
 LR indicator 56

 M1, M2, M3 49
 Machine features required 2
 Matching-field holding area 49
 Matching Fields or Chaining Fields 49, 179
 Matching Record indicator 57, 128
 Methods of Processing Tables 107
 Minus-Calculations 71, 181

Minus-input 52, 179
 Minus symbol, edit word 87
 Minus test 18
 Minus symbol, edit word 87
 Minus test 18
 Minus zone 39
 Move Operations 59, 61, 69
 High-to-High (MHHZO) 63, 69
 High-to-Low (MHLZO) 62, 69
 Low-to-High (MLHZO) 62, 69
 Low-to-Low (MLLZO) 62, 69
 Move (MOVE) 61, 69
 Move Left (MOVEL) 61, 69
 Move Remainder (MVR) 60, 69
 Mode of Processing 95, 176
 MR indicator 57, 128
 MULT 59, 60, 69
 Multiple input files 127
 Multiple printers 78, 86
 Multiplication and Division 22
 Multiply, operation 59, 60, 69

 Name of Label Exit 98, 176
 No zone 39
 Not 5, 39, 179
 Number 25, 37, 178
 Number of the Chaining Field 99, 177
 Number of files allowed 91
 Number of Table Entries per Record 101, 177
 Number of Table Entries per Table 101, 177
 Numbering pages 84
 Numeric field maximum length 42
 Numeric field format 31, 112
 Numeric literals, forming 58

 Object module sequence 140
 Object module cancellation 142
 Omitting record identification 41
 Operation 59, 180
 Option 25, 37, 178
 OR relationship 41, 79, 178, 181
 Output Deck 140
 Output file 91
 Output-Format Specifications Sheet 75, 172, 181
 Output Indicators 7, 78, 80, 182
 Output Listing 140
 Output Units, specifying 75
 Overflow indicator 77, 96, 176
 Overflow printing 13
 Overflow, printing lines conditioned by 77
 Overriding statements in cataloged procedures 146

 Packed Field — Output 84, 182
 Packed — File Extension 101, 177
 Packed Format, Exit subroutine 112
 Packed — Input 42, 179
 Page 33, 175
 Page numbering 84
 Pence-format fields 173
 Physical unit 115
 Plus-Calculations 71, 181

 Plus-Input 52, 179
 Plus test 18
 Plus zone 39
 Position 5, 39, 179
 Pound sign # 31, 36
 Pound sterling formats 173
 Printer spacing chart 29
 Printing lines conditioned by overflow 77
 Problem definition 29
 Processing multiple input files 127
 Processing single input files 120
 Processor Control card 148
 Program identification 34, 149, 175
 Program logic 28
 Providing a label for GOTO 59, 66, 69

 Random processing 117
 Random processing, indexed-sequential files 117, 121
 Record, definition of 115
 Record Address File (RAF) 124
 Record Address Type 95, 176
 Record formats 93
 Record identification 35, 36
 Record identification, omitting 41
 Record Identification Codes 5, 39, 178
 Record key 59, 67, 69
 Record Length 95, 176
 Record Sequence of the Chaining File 99, 177
 Records in an OR-relationship 41, 79, 178, 181
 Registers, use of 112
 Required features 2
 Result Field 6, 8, 70, 180
 Resulting Indicator — Input 5, 37, 178
 Resulting Indicators — Calculation 21, 71, 181
 Retrieving updated tables 106
 Return codes, compiler 141, 193
 Return codes, load module 143
 Right-justification 31
 RLABL 59, 66, 69, 111
 RPG Control Card 148
 RPG conversion 59, 68, 69
 RPG Source Program Deck Arrangement 148
 RPG Label 59, 68, 69
 RPG logic flow 169
 RPG specification sheets 31
 RPGCV 59, 68, 69, 174
 Rules for DASD specifications 95

 Sample programs, (Appendix A)
 1, 152
 2, 158
 3, 162
 Search argument, table operations 107
 Sequence checking 24
 Sequence checking, matching field 49
 Sequence checking of input files 92
 Sequence — File Description 92, 175
 Sequence — File Extension 101, 177
 Sequence — Input 36, 178
 Sequence of Input records 24, 35, 93
 Sequential organization 116
 Sequential processing 117, 120
 Sequential processing, multiple input files 127

Set indicators,
 off 67
 on 67
 SETOF 59, 67, 69, 181
 SETON 59, 67, 69, 181
 Skip-After 77, 182
 Skip-Before 77, 182
 Skipping, automatic 77, 89
 Source deck arrangement 148
 Space-After 7, 77, 181
 Space-Before 77, 181
 Spacing chart, printer 29
 Special holding area (table operations) 68
 Specifying output units 75
 Specifying the kind of calculation 58
 Split chaining fields 133
 Split control fields 48
 Stacker Select--Input 5, 41, 179
 Stacker Select--Output 15, 76, 181
 Status, edit word 86
 Sterling routines 31, 172
 Sterling Sign Position--Input 53, 180
 Sterling Sign Position--Output 88, 182
 SUB 59, 69
 Subtract, operation 59, 69
 Subtraction 6
 Summary of file processing 136
 Summary of RPG specification
 (Appendix F) 175
 Summary punching 14
 Supported Features 2
 Symbolic Device 176
 Symbolic parameters, job control
 language 146

 Table linkage field 112
 Table lookup 59, 67
 Table Name 100, 177
 Table Name (alternating table) 101, 177
 Tables, exit routine 112
 TAG 59, 66, 69, 114.1
 Testing for zero, plus, and minus 18
 Testing or Compare operations 63
 Testing the results of calculations 71

 TESTZ 59, 64, 69, 181
 Test Zone, operation 59, 64, 69, 181
 To: Field location 42, 179
 To Filename 100, 177
 Total calculations 9
 Total printing 9
 Total records 76
 Total time 28
 Translate subroutine 51, 149
 Turning indicators On or Off 59, 67
 Type H/D/T 7, 76, 181
 Type of File Organization 95, 176

 ULABL 59, 66, 69, 112
 Unpacked format 31, 42
 Update file 91
 Updated tables, retrieving 106
 Updating a DASD file 136
 Use of field indicators 53
 Use of registers 112
 User cataloged procedures 146
 User's Label 66
 Using Cataloged Procedures 142
 Using exit routines in the object
 module 104
 Using indicators 57
 Using resulting indicators 19, 71
 Using RPG 1
 Using Tables in the Object Module 104
 Using the Calculations Specifications
 Sheet 72

 Variable-length records 41, 116

 Zero and Add, operation 59
 Zero and Subtract, operation 60
 Zero or Blank--Calculations 71, 181
 Zero or Blank--Input 52, 179
 Zero test 18
 Zero Suppress 8, 82, 182
 Zero Suppression, edit word 87
 Zone, definition of 39
 Zone record identification 39

READER'S COMMENT FORM

IBM System/360 Operating System
Report Program Generator
Language

Form C24-3337-3

- Your comments, accompanied by answers to the following questions, help us produce better publications for your use. If your answer to a question is "No" or requires qualification, please explain in the space provided below. Comments and suggestions become the property of IBM.

- | | Yes | No |
|--|---|--------------------------|
| ● Does this publication meet your needs? | <input type="checkbox"/> | <input type="checkbox"/> |
| ● Did you find the material: | | |
| Easy to read and understand? | <input type="checkbox"/> | <input type="checkbox"/> |
| Organized for convenient use? | <input type="checkbox"/> | <input type="checkbox"/> |
| Complete? | <input type="checkbox"/> | <input type="checkbox"/> |
| Well illustrated? | <input type="checkbox"/> | <input type="checkbox"/> |
| Written for your technical level? | <input type="checkbox"/> | <input type="checkbox"/> |
| ● What is your occupation? _____ | | |
| ● How do you use this publication? | | |
| As an introduction to the subject? <input type="checkbox"/> | As an instructor in a class? <input type="checkbox"/> | |
| For advanced knowledge of the subject? <input type="checkbox"/> | As a student in a class? <input type="checkbox"/> | |
| For information about operating procedures? <input type="checkbox"/> | As a reference manual? <input type="checkbox"/> | |
| Other _____ | | |
| ● Please give specific page and line references with your comments when appropriate. | | |

COMMENTS:

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.

fold

fold

FIRST CLASS
PERMIT NO. 387
ROCHESTER, MINN.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
Systems Development Division
Development Laboratory
Rochester, Minnesota 55901

Attention: Programming Publications, Dept. 425



fold

fold



International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

IBM

**International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]**

**IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]**