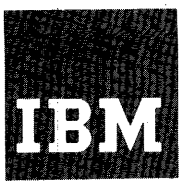
File Number 1130-28
Form C21-5002-0



Systems Reference Library

IBM 1130 Planning For RPG

This publication is a planning aid only. It is intended for use prior to the availability of the IBM 1130 RPG and shall be replaced by reference documentation when the IBM 1130 RPG program becomes available.

This reference publication contains fundamentals of RPG programming and language specifications for the IBM 1130 Computing System. For information on RPG that is beyond the purpose of this language publication, see IBM 1130 Disk Monitor System, Version 2, Programming and Operator's Guide.

For the titles and abstracts of associated publications, see the IBM 1130 Bibliography, Form A26-5916.

PREFACE

This publication is intended for users of the IBM 1130 Computing System. IBM 1130 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 printed reports,
4. Use table lookup,
5. Exit to a user's subroutine written in a language other than RPG,
6. Branch within the calculations, and
7. Sequence-check input records.

RPG uses a set of specifications 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.

First Edition

This edition is intended for planning purposes only. When the program described by this publication is made available for distribution, any changes made to the specifications herein 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 is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to IBM Corporation, Programming Publications, Department 425, Rochester, Minnesota 55901.

This publication provides the information required to use the disk resident RPG for the IBM 1130.

RPG consists of a problem-oriented symbolic programming language and a compiler program. The RPG language is highly flexible. It provides an easy-to-use technique for writing a wide variety of 1130 programs. The compiler program reads the program specifications written in the symbolic language and produces an object program in machine language that can be used to perform a particular application. Both compilation and execution of object programs is under control of the Disk Monitor, Version 2.

Prerequisites

The reader must be familiar with the operation of the components of his 1130 system and with basic programming concepts. For titles and abstracts of associated publications, refer to the IBM 1130 Bibliography.

Machine Requirements

The minimum machine requirements for generating an RPG object program are as follows:

- 8192 words of core storage
- One card-reading device
- One IBM 2315 Disk Cartridge
- One printer (IBM 1132, 1403, or console printer).

The minimum machine requirements for the execution of an RPG object program depend on the I/O configuration used:

- 8192 words of core storage
- Input/Output devices as required by the object program:

  IBM 1403 Printer

  IBM 1442-5 Card Punch

  IBM 1132 Printer

  IBM 2501 Card Reader

  IBM 1442 Card Read Punch

  IBM 2310 Disk Storage Drive, Model B1 and B2

The following system features are supported for program generation:

- Up to 32,768 words of core storage
- A card-punching device if the object program is to be punched
- One printer with at least a 48-character set for program listings
- One additional IBM 2310 disk unit.

FUNCTION OF RPG

When RPG is used, the IBM 1130 actually performs two separate functions:

1. Program generation
2. Program execution.

In the first case, program specifications defined by the programmer are used to produce machine-language instructions. Storage areas are automatically assigned, constants or other reference factors are included, and program routines for checking, for input/output operations, and for other functions are produced.

In the second case, the machine-language instructions are combined with the input data files and both are processed through the system to produce the desired reports and output files.

USING RPG

The preparation of a report by means of RPG consists of the general operations illustrated in Figure 1 and described as follows. (The circled numbers in Figure 1 refer to the numbers in the following text.)

1. The programmer evaluates the report requirements to determine the format of the input files and the layout of the finished report. For example, he determines what fields in the input records are to be used, what calculations are to take place, where the data is to be located in the output records and how many and what kind of totals must be accumulated. More specific information regarding the evaluation of report requirements is given in the section Problem Definition.

2. After the programmer has evaluated the requirements of the report, he provides the RPG program with information about these requirements.

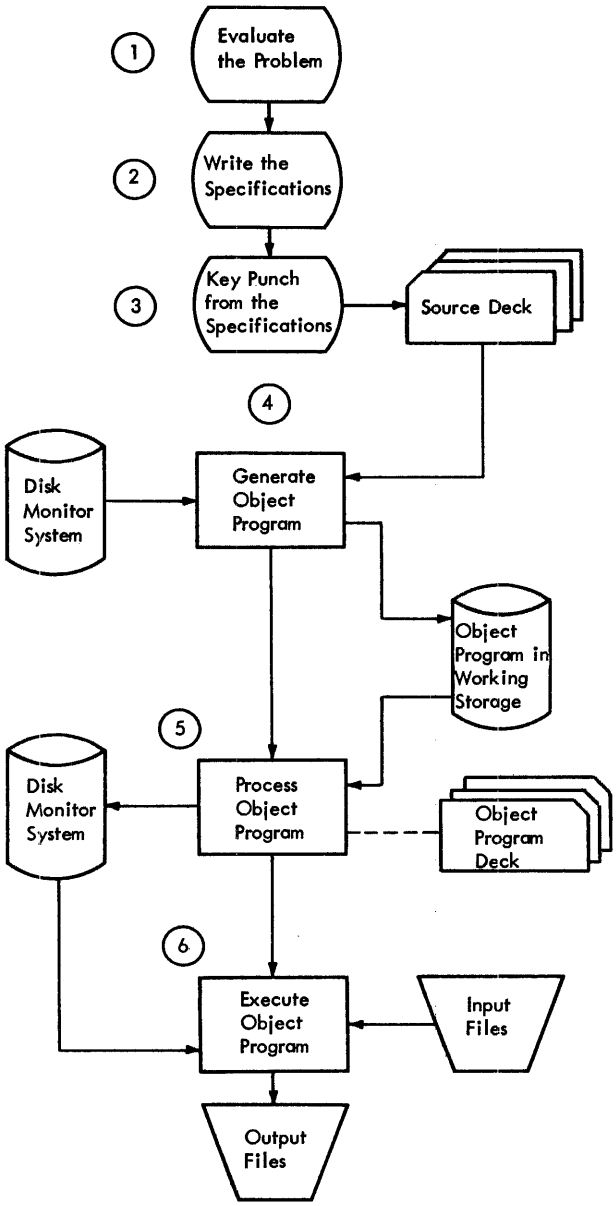


Figure 1. Producing Reports Using the RPG Program

a. He describes his input (record layout fields used, etc). This is done by making entries on the Input Specifications form.

b. He states what processing is to be done (add, subtract, multiply, divide, etc.) by means of entries on a Calculation Specifications form.

c. He defines the layout of the desired report (print positions, carriage control, etc). This is accomplished by making entries on the Output Format Specifications form.

d. He describes all files used by the object program (input files, output files, table files, etc.) by making entries on the File Description Specifications form.

e. If the programmer uses chaining files, record address files, or tables in his object program, he furnishes information about them through entries on the Extension Specifications form.

3. After the specifications have been written on the appropriate forms, cards are key-punched with the data from the forms. Each line on the form is punched into one card.

4. These punched cards (called a source deck) are combined with the RPG control card. The source deck and the control card are placed into a card-reading device and processed by the RPG compiler under control of the Disk Monitor program. At the end of this processing run (referred to as the compilation run), a program capable of preparing the report specified by the programmer has been produced and stored in the working storage area of the disk.

   This program, (known as the object program) contains all the machine instructions required to prepare the desired report.

5. The programmer may now have the object program punched into cards for storage or he may proceed directly to processing of the object program by causing it to be made a part of the Disk Monitor System.

6. The input files are then read into the system and production of the report begins. This is known as the object run.

COMPATABILITY OF 1130 RPG FILE ORGANIZATION TO OTHER 1130 FILE ORGANIZATION METHODS

1130 RPG uses its own unique file organization techniques. Disk files not created by RPG cannot be processed by RPG.

COMPATABILITY OF 1130 RPG TO OTHER SYSTEMS

The implementation of RPG for the IBM 1130 System may differ in some respects with that of other systems. Check the following language features when considering compatability to another system.

1. Halt indicators H1 through H9 are supported by 1130 RPG.
2. Matching record fields M1 through M9 are supported by 1130 RPG.
3. Chaining fields C1 through C3 are supported by 1130 RPG.
4. Calculation operations implemented for 1130 RPG include the operations CHAIN, BEGSR, ENDSR, EXSR, and EXCPT.
5. Edit codes are supported by 1130 RPG.
6. File Description Specification entries differ according to the devices used and the control program under which RPG is run.
7. Blank/zero indicators are not initialized (set on) and are not reset on a Blank After condition.
8. Reserved words under 1130 RPG are: ALTSEQ, PAGExx, TAByyy, and INxx where yyy and xx represent any alphameric characters.
9. Exception output can be produced with 1130 RPG.
10. Number and type of files supported may vary.
11. No checking for a valid sign will occur during the moving to numeric fields. Signs will be forced on all arithmetic operations.
12. Additions to an indexed sequential file cannot be made during the program that is also retrieving that file.
13. Packed data may only be specified on disk records.
14. Numeric fields may be from one to 14 digits in length.

FUNDAMENTALS OF RPG PROGRAMMING

Although RPG programs can process files contained on magnetic disk, only card-input files are used for the introduction to RPG. Programmers familiar with the basic concepts of RPG may omit this section and concentrate on the detailed descriptions contained in the section RPG Specification forms of this manual.

DEFINITION OF TERMS

The following is a definition of terms used in this publication as they apply to 1130 RPG.

EBCDIC Notation

EBCDIC (Extended Binary Coded Decimal Interchange Code) is the 1130 machine code. It consists of 256 characters. (Refer to IBM 1130 Functional Characteristics, Form A26-5881.)

Alphabetic Characters

Alphabetic characters include the 26 letters of the alphabet, A through Z.

Numeric Characters

Numeric characters include the digits 0 through 9.

Special Characters

All EBCDIC characters that are neither alphabetic nor numeric are referred to as special characters.

Hexadecimal

A number system using the equivalent of the decimal number sixteen as a base.

Alphameric Fields

All fields for which a decimal-positions specification has not been made in the appropriate column of any of the pertinent specifications forms, regardless of whether their contents are alphabetic, numeric, or alphameric. The contents of alphameric fields are placed into core storage in unpacked format.

Numeric Fields

All fields that have a decimal-positions specification in the appropriate column of any of the pertinent specification forms. The contents of numeric fields are placed into core storage in unpacked decimal format. Numeric fields contain only numeric characters and possibly a plus or minus sign over the rightmost position.

Alphameric Literals

Alphameric literals may consist of any of the 256 EBCDIC characters. They must be enclosed in apostrophes (').

Numeric Literals

Numeric literals may consist of the digits 0 through 9, and may contain one decimal symbol and/or a plus or minus sign. Numeric literals must not be enclosed in apostrophes.

FUNCTIONS DESCRIBED

This section shows how to describe files and records, and how to specify the most frequently used RPG functions.

DESCRIBING THE FILES

In this example, the File Description Specifications form illustrated in Figure 2 is used to furnish the RPG program with the required information about the files used in the job.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
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	File Designation	End of File	Sequence	File Format	Block Length	Record Length	L/R	Mode of Processing	Length of Key Field or of Record Address Field	Record Address Type	Type of File Organization	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind
		7 8 9 10 11 12 13 14	I/O/U/C	P/S/C/R/T	E	A/D	F/V	20 21 22 23	24 25 26 27	28	29 30	K/I	I/O/T	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
01	F	DETLABORCP					F		80									READ42								
02	F	DAILYRPTO					F	132	120									PRINTER								
03	F																									
04	F																									

Figure 2. Example of File Description Specifications

Div.	Dept.	Emp. No.	Employee Name	Code	Field A	Field B	Field C	Code
0000	0000	0000000	00000000000000000	0000000000	00000	00000	00000	0
1111	1111	1111111	11111111111111111	1111111111	11111	11111	11111	1
2222	2222	2222222	22222222222222222	2222222222	22222	22222	22222	2
3333	3333	3333333	33333333333333333	3333333333	33333	33333	33333	3
4444	4444	4444444	44444444444444444	4444444444	44444	44444	44444	4
5555	5555	5555555	55555555555555555	5555555555	55555	55555	55555	5
6666	6666	6666666	66666666666666666	6666666666	66666	66666	66666	6
7777	7777	7777777	77777777777777777	7777777777	77777	77777	77777	7
8888	8888	8888888	88888888888888888	8888888888	88888	88888	88888	8
9999	9999	9999999	99999999999999999	9999999999	99999	99999	99999	9

Figure 3. One of the Input Cards from a Detail Labor File

The first line provides information about the detail labor file. The file name DETLABOR identifies this file. The letter C in file type indicates that the file is a combined file. A combined file is a file that contains both cards read into the system and cards used for punching. The P in file designation is required. The particular input/output device that is to be used with this file is specified in Device. In this example, the IBM 1442 Reader Punch is used.

The second line describes the output report on the printer. The O in file-type indicates that the file is an output file.

DESCRIBING A RECORD AND ITS FIELDS

Entries on the Input Specifications form describe the data that is to be read into the system.

Example

A detail labor file contains card records as shown in Figure 3. Three fields from

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3356
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 04

Program Identification: REPT01

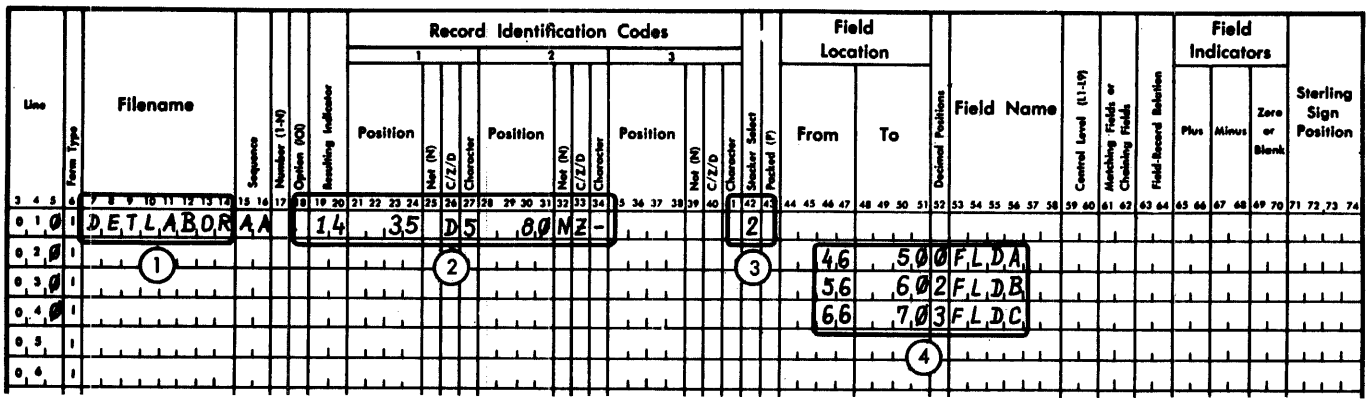


Figure 4. Input Specifications Form

each card are to be read into storage. Field A is contained in columns 46-50, field B is contained in columns 56-60, field C is contained in columns 66-70 of each record.

Each card that contains fields A, B, and C is identified by two distinct attributes: It must contain a digit 5 in column 35 and no 11-punch in column 80. When these two conditions exist in the card record, the record is valid; that is, it contains fields A, B, and C.

Specifications: In Figure 4, the entries on the Input Specifications form that apply to this problem are circled and identified by numbers referring to the items in the text below. The remaining fields on the form are described later in the publication.

1. Each file must be given a name. Each file name can be used to identify one file only. This file name must be the same as specified on the File Description Specifications form.

2. If several card types exist in a file, each card type must be identified by its particular card code. The identifying codes must be entered into the Record Identification Codes (columns 21-41) of the Input Specifications form.

   In this example, the card codes are specified by writing a 5 in Character (column 27) and a 35 in Position (columns 21-24) for the first code; and by writing an 80 in Position, a minus in Character, and an N in Not (column 32) for the second code. The fields C/Z/D (columns 26 and 33) on the form indicate how the card code is to be compared:

   D = digit portion only.
   Z = zone portion only.
   C = zone and digit portions.

   After all the identification codes are established, the programmer assigns a two-digit number (from 01-99) to the card type identified. This code, known as a Record Identifying Indicator (columns 19-20), is used as a switch or selector. Whenever a card of the specified type is being read, the associated indicator is turned on. If the card read is not one of the type specified, the indicator is turned off or remains off. Thus, the programmer can use the indicator during processing as a selector or switch. The Record Identifying Indicator is used wherever it is necessary to refer to the specific card type on other RPG specification forms.

3. If the input card is to be placed into a stacker other than the one into which it would normally be selected, the stacker number is written in Stacker Select (column 42).

4. Each field of the card to be read must be defined as shown on the lines fol-

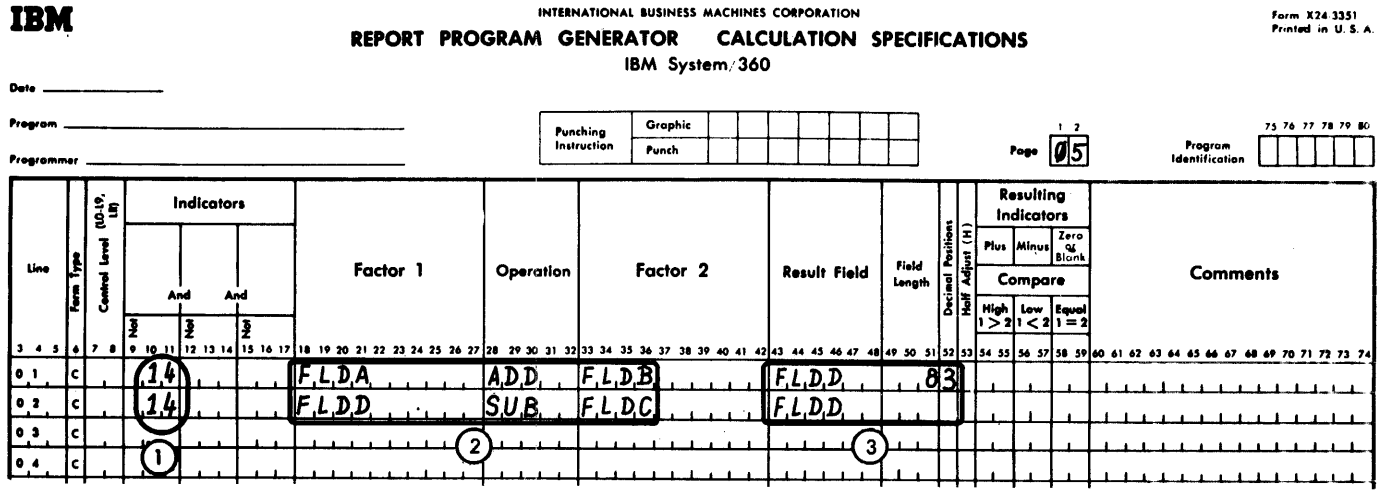


Figure 5. Calculation Specifications Form

lowing the record identification line (columns 7-42). The specifications shown in this example obtain the data from the input record and place it in fields A, B, and C in three separate locations in storage (named FLDA, FLDB, and FLDC). The entries in Decimal Position (column 52) define the associated fields as numeric.

The programmer need not know the actual location of these fields in core storage. Whenever reference is to be made to these fields on other RPG forms, the programmer writes only the appropriate field name.

A field name may be alphabetic or alphameric; blanks and special characters may not be used. The first character of a name must be alphabetic. A field name may not be more than six characters in length.

ADD AND SUBTRACT (CROSSFOOT)

Example

In this example, the calculation A + B - C = D is to be performed. This operation is sometimes referred to as crossfooting. The contents of fields A, B, and C from the previous example are used to calculate the contents of a new field named FLDD. Figure 5 shows the required entries.

Specifications: The first line of the Calculation Specifications form causes the contents of field B (FLDB) to be added to the contents of field A (FLDA) and the result of this arithmetic operation to be placed in field D (FLDD), thus replacing any previous information in this field.

The second line of the Calculation Specifications form causes the object program to subtract the contents of FLDC from the contents of FLDD, and to replace the contents of FLDD by the result of this subtraction.

The numbers in the following text refer to the items circled in Figure 5.

1. The 14 in Indicator (columns 9-11) specifies the condition under which the calculations are to be performed. This number was assigned as the Record Identifying Indicator on the Input Specifications form (see Figure 4).

2. The card columns for FLDA, FLDB, and FLDC were defined on the Input Specifications form. These fields are referred to on the Calculation Specifications form by their names.

3. The result field FLDD is defined for the program by 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 reserved for this field in core storage. Just as in the case of FLDA and FLDB, which were defined on the

input form, the name FLDD can now be used in other calculation operations or in output specifications.

Decimal Positions: If an entry is made in Decimal Positions, the field is considered a numeric field. In operations involving quantities that have decimal fractions, 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 of an arithmetic operation, a shifting of the factors to align the decimal point is usually required.

RPG automatically shifts factors and aligns decimal points. The programmer need only indicate the number of decimal positions required in the result field. If a field is to be used in arithmetic operations, the number of decimal positions must be specified. If there are no decimal fractions, a 0 must be entered in column 52. The fields in Figure 4 have decimal positions as indicated in column 52 of the Input Specifications form.

A set of values for these fields might appear in the program as follows.

FIELD	DECIMAL POSITIONS	CONTAINED IN CARD	ACTUAL VALUE
A	0	000126	126.
B	2	201123	11.23
C	3	304264	4.264

If, as in Figure 5, the programmer had specified three decimal positions for the result field in Decimal Positions (column 52), the calculation A + B - C would result in a shifting of the values as follows, (result field has three decimal positions):

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

The result 00132.966 is stored as the value of FLDD.

DETAIL PRINTING

The printing of information obtained from each record as it is read, is called Detail Printing.

Example

This example illustrates how the input fields FLDA, FLDB, and FLDC and the calculated result (FLDD) can be specified for listing.

Specifications: Figure 6 shows the output specifications required. The numbers in the following text refer to the numbers circled in Figure 6.

IBM

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 06

Program Identification REPT01

Line	Form Type	Filename	Type (H/D/T)	Space Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	DAILYRPTD			2				14									
02	O	(1)							(2)			FLDA	Z		40			
03	O											FLDB	Z		48			
04	O											FLDC	Z		56			
05	O											FLDD	Z		66			
06	O											(3)						
07	O																	
08	O																	

Figure 6. Output-Format Specifications Form

1. A filename must be assigned to the output listing. In this example it is named DAILYRPT.

   The D in Type H/D/T/E (column 15) indicates that the line to be printed is a detail line. (The other possible entries, H, T, and E are described later.)

   Space After (column 18) contains a 2 and provides a double-spaced listing. (Each printed line is followed by one blank line.)

2. The 14 in Output Indicator (columns 24-25) governs the printing of the line.

3. Each field to be printed must be specified in Field Name (columns 32-37). The Z in column 38 means that all zeros to the left of significant digits and the sign are not printed (suppressed). For example, if the value 00126 is stored in FLDA, it is printed as 126.

   The positions where the data is to be printed on the output report are specified in End Position in Output Record (columns 40-43). All specifications in these columns must be right justified. The programmer indicates only the last (or rightmost) print position of each field. The length of each field has been established either as part of the input specifications, or in the Result Field of the Calculation Specifications form.

   When preparing a simple listing such as the one in this example, the programmer would probably mentally select print positions based upon the number of positions in each field plus a number of positions for spaces between two fields. More elaborate reports are laid out before the pertinent program is written. Usually, this indicates both the location on the form where data is to be printed and the source of the data (the card columns of the input data or the name of data fields developed in the program). This is normally a function of job definition, which is discussed later under Problem Definition.

Figures 3, 4, and 5 show how to read data into the system, how calculations can be performed upon the data, and how the original data and the data developed in the program can be printed. The examples so far have shown the items required to prepare a report. A significant item not yet described is the specification of control fields to obtain various totals.

ESTABLISHING CONTROL FIELDS

A field containing information to be compared from record to record is called a control field. A control break occurs when the information in the control field of a record is different from the information in the control field of the preceding record. A control level establishes the relative importance of the control fields.

Example

This example illustrates the specification of three control levels.

Specifications: Figure 7 contains the specifications shown in Figure 4 with the addition of the appropriate control data (the items inside the circle).

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 04

Program Identification: REPT01

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	DETLABORAA				14	35		D	5	80		M	Z-					2												
02	I																				1	4		DIVSON	L3						
03	I																				5	8		DEPT	L2						
04	I																				9	14		EMPNO	L1						
05	I																				15	28		NAME							
06	I																				46	50		FLDA							
07	I																				56	60		FLDB							
08	I																				66	70		FLDC							
09	I																														

Figure 7. Specifying Control Levels on the Input Specifications Form

IBM

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 05

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And	Indicators: And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1 > 2	Compare: Low 1 < 2	Compare: Equal 1 = 2	Comments
010	C		14		FLDA	ADD	FLDB	FLDD	7	0								
020	C		14		FLDD	SUB	FLDC	FLDD	7	0								
030	C		14		TOTE	ADD	FLDD	TOTE	8	0								
040	C	L1			TOTF	ADD	TOTE	TOTF	8	0								
050	C	L2			TOTG	ADD	TOTF	TOTG	8	0								
060	C	L3			FINTOT	ADD	TOTG	FINTOT	8	0								
070	C																	
080	C																	

Figure 8. Specifying Control Levels on the Calculation Form

However, in this example the numeric fields are specified as having no decimal fractions.

This example shows three levels of totals:

- The lowest level is the employee number (EMPNO).
- The next highest level is the department (DEPT).
- The highest level is division (DIVSON).

These levels are designated by the control level codes L1, L2, and L3 placed in columns 59-60.

Up to nine control levels plus a final total can be used in RPG. Level 1 is the lowest control level; level 9 is the highest. A control break at one level forces control breaks for all lower levels.

To designate a control field and to establish a level of control on the Input Specifications form:

- Enter the card columns of the field in Field Location (columns 44-51).
- Enter the appropriate name in Field Name (columns 53-58).
- Enter the correct control level (L1, L2,. ..L9) in Control Level (columns 59-60).

TOTAL CALCULATIONS

When a control break occurs, certain special operations (called total operations) are normally performed before processing the record that caused the control break.

Example

This example illustrates the accumulation of totals at each control level.

Specifications: Figure 8 shows calculations that can be performed on the input data shown in Figure 7. The calculation examples specified in the first and second lines of Figure 8 are the same as in Figure 5. The third line adds the result, which is contained in FLDD, to a field called TOTE. Thus, TOTE is the accumulated amount for each employee group.

The operations specified in the first three lines are performed each time a detail card is read.

When the level 1 (L1) control break occurs (i.e., a card containing a different employee number has been read), specification line 04 adds the contents of TOTE (accumulated from each detail card) into a field named TOTF. This is used to accumulate the total for the employee group of each department.

When the level 2 (L2) control break occurs (change in department) in addition to the previous calculation, the contents of TOTF are added to a field named TOTG.

When the level 3 (L3) control break occurs, all three total calculations specified with L1, L2, and L3 are performed.

DETAIL AND TOTAL PRINTING

Example

This example shows the specifications required to print the following:

- Three control fields.
- The name.
- The amount accumulated in FLDD.
- The three accumulated totals (when a control break occurs at level 1, level 2, and level 3).
- The final total at the end of the report.

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

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System 360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 06

Program Identification: REPT0L

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	DAILYRPTD			1				1P								
02	O										DIVSON			4			
03	O										DEPT			10			
04	O										EMPNO			17			
05	O										NAME			33			
06	O										FLDD	Z		64			
07	O		T		3				L1								
08	O										TOTE	ZB		66			
09	O		T		3				L2								
10	O										TOTF	ZB		66			
11	O		T		3				L3								
12	O										TOTG	ZB		66			
13	O		T		0				LR								
14	O										FINTOT	Z		66			
15	O																

Figure 9. Specifications on the Output-Format Specifications Form

1. This information is similar to the specifications in Figure 6. The 1 in Space After (line 06010, column 18) provides one space between the printed lines of the report; i.e., there are no blank lines between the printed lines of the report.

2. The data fields named 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 Type H/D/T/E (Column 15) indicates that the line is a total line.

   b. The 3 in Space After (column 18) provides two blank lines after each total line to make the report easier to read.

   c. The level 1 indicator (L1) specified in Output Indicator (columns 23-25) indicates that the line is to be printed each time a level 1 or any higher control break occurs.

   d. The field to be printed (TOTE) is indicated under Field Name (columns 32-37).

   e. The Z in Edit Codes (column 38) indicates that zeros to the left of significant digits and the sign are not to be printed.

   f. The B in Blank After (column 39) causes the core-storage positions containing the field TOTE to be set to zeros after the total has been transferred to the print line. This is done to avoid adding the total for one group to the total for the previous groups.

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

4. The specifications for printing the final total contain the code LR (Last Record) in Output Indicators (columns 23-25). LR is turned on by the program at the end of the job. This indicator is used to cause the final total to be printed. At this time all control-

level indicators (L1-L9) are turned on and all total lines are printed in the specified sequence.

```
DIVSON           NAME
     DEPT EMPNO

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

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.

```
DIVSON DEPT EMPNO   NAME

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

Figure 11. Group-Printed Report

The specifications for the detail-printed report shown in Figure 9 could be altered as illustrated in Figure 12 to provide a group-printed report. The difference between Figure 9 and 12 is:

1. The detail line specification in Figure 9 (line 06010) has been changed to a total line specification on an L1 control break (Figure 12, line 06010).

2. The total line in Figure 9 (line 06070) has been combined with the first total line 06010 of Figure 12.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3282
Printed in U. S. A.

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 06

Program Identification: REPORT

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign 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 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
010	O	DAILYRPT	T		1				L1									
020	O											DIVSON			4			
030	O											DEPT			10			
040	O											EMPNO			17			
050	O											NAME			33			
060	O											TOTE	Z	B	64			
070	O		T		1				L2									
080	O											TOTF	Z	B	66			
090	O		T		1				L3									
100	O											TOTG	Z	B	66			
110	O		T		0				LR									
120	O											FINTOT	Z		66			

Figure 12. Specifying a Group-Printed Report

3. The space-after specification on lines 06090 and 06110 of Figure 9 has been changed in Figure 12 from three spaces to one space after printing.

In group-indication operations, each detail card is processed. However, only the control fields that identify the specific detail card are printed.

Figure 13 shows an example of a group-indicated report. In this example, name and number of each employee are printed at each control change (level 1). The fields for division and department are printed only when a control break occurs at L2 or L3, respectively.

```
DIVSON  EMPNO

01120246011426SMITH  101 ------FLDD
                     100
                     102
                     101
                       404 ---TOTE

         011428JONES 120
                     122
                     123
                     121
                       486

         001430BROWN 100
                     103
                     102
                     104
                       409
                      1299 ----TOTF

      0310011296GREEN 121
                     120
                     144
                       487

         011298BLAND  98
                      86
                       184

                       671
                       970 ----TOTG

01140069001262ADAMS  146
                     237
                     184
                     197
                       764

         001278JAMES 182
                     176
                     160
                     164
                       682

                      1446

                     1446
                     3416 --FINTOT
```

Figure 13. Example of a Group-Indication Report

Figure 14 illustrates how the specification for the detail-printed report shown in Figure 9 could be altered to specify a group-indicated report.

OVERFLOW PRINTING

Overflow printing, another function used in preparing reports, can be performed by RPG. An overflow occurs when a punch in channel

IBM

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 06

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	DAILYRPT	D		1				1P									
02	O								L3			DIVSON			4			
03	O								L2			DEPT			10			
04	O								L1			EMPNO			17			
05	O								L1			NAME			33			
06	O											FLDD	Z		64			
07	O																	
08	O																	
09	O																	

Figure 14. Specifying a Group-Indication Report

12 of the carriage-control tape is encountered.

Example

This example shows the specifications required to print eight column headings at the top of each printed form. Heading lines may contain information from an input record (e.g., the date) or constant information that is defined on the Output-Format Specifications form.

Specifications: The output specifications required for this operation are illustrated in Figure 15. The numbers in the margin of the text refer to the circled numbers in Figure 15. The filename (DAILYRPT) is the same as in the other specifications required for the report.

1. On the first line of the form, the H in Type H/D/T/E (column 15) indicates that the line to be printed is a heading line. The 3 in Space After (column 18) provides two blank lines after each printed heading. The 01 in Skip Before (columns 19-20) causes the printer carriage to skip to a punch in channel 1 of the carriage-control tape before printing the line.

   The OF in Output Indicator (columns 23-25) causes the heading line to be printed each time an overflow condition occurs.

   The indicator OF is specified on the File Description Specifications form for the device associated with the filename DAILYRPT.

   The indicator 1P is on only at the very beginning of an RPG program.

   The letters OR on the second line (columns 14-15) of Figure 15 indicate a second condition that can cause the heading line to be printed. This is specified by means of the entry 1P in columns 23 and 25. The entry 1P is required to have the heading printed on the first page of the report because the overflow condition does not occur until after at least one page has been printed.

2. The entries on lines 03-10 of Figure 15 specify the actual or constant information to be printed in the heading of the report. Note that an apostrophe must be placed before and after each constant defined.

3. The specifications for printing the contents of detail cards follow the last specifications for the heading line.

IBM

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	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign 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 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
01	O	DAILYRPT	H			3	Ø1		OF									
02	O		OR						1P									
03	O						(1)								3		'DIV'	
04	O														10		'DEPT'	
05	O														17		'EMP NO'	
06	O												(2)		24		'NAME'	
07	O														40		'FLD A'	
08	O														48		'FLD B'	
09	O														56		'FLD C'	
10	O														64		'FLD D'	
11	O	DAILYRPTD				1			14									
12	O											DIVSON			4			
13	O											DEPT			10			
14	O											EMPNO			17			
15	O										(3)	NAME			33			
16	O											FLDA	Z		40			
17	O											FLDB	Z		48			
18	O											FLDC	Z		56			
19	O											FLDD	Z		64			
	O																	

Figure 15. Specifications for Overflow Printing

SUMMARY PUNCHING

Example: In this example, a summary total of employee totals (TOTE) is to be punched for each department together with the appropriate department and division numbers.

Specifications: Before the processing of the object program begins, a blank card is inserted behind each department group. The blank card is specified on the Input Specifications form as illustrated in Figure 16, line 09.

The calculation specifications for the summary totals to be punched are illustrated in Figure 17. Line 04 is processed when the blank card is read. It causes the accumulation of the level 2 total for the summary card.

In this example, this specification is required because the summary cards are inserted behind the control groups rather than obtained from a second file of blank cards. The level 2 control break does not occur until the summary card is punched and the first card of the next group is read.

Control level L0, which is explained later in this publication, has been entered only to facilitate identification as a total calculation.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7 8 9 10 11 12 13 14	Sequence 15 16	Number (1-N) 17	Option (O) 18	Resulting Indicator 19 20	Record Identification Codes 1: Position 21 22 23 24	1: Not (N) 25	1: C/Z/D 26	1: Character 27	2: Position 28 29 30 31	2: Not (N) 32	2: C/Z/D 33	2: Character 34	3: Position 35 36 37 38	3: Not (N) 39	3: C/Z/D 40	3: Character 41	Stacker Select 42	Packed (P) 43	Field Location: From 44 45 46 47	To 48 49 50 51	Decimal Positions 52	Field Name 53 54 55 56 57 58	Control Level (L1-L9) 59 60	Matching Fields or Chaining Fields 61 62	Field-Record Relation 63 64	Field Indicators: Plus 65 66	Minus 67 68	Zero or Blank 69 70	Sterling Sign Position 71 72 73 74
0 1	I	DE,TLABOR	AA			14	35		D	5	80												DIVSON	L3						
0 2	I																			1	4		DEPT	L2						
0 3	I																			5	8		DEPTNO	L1						
0 4	I																			9	14		EMPNO							
0 5	I																			15	28		NAME							
0 6	I																			46	50		FLDA							
0 7	I																			56	60		FLDB							
0 8	I																			66	70		FLDC							
0 9	I		NS			16	35		C																						
1 0	I																														

Figure 16. Summary Punching Example, Input Specifications

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And Not 9 10 11	And Not 12 13 14	And Not 15 16 17	Factor 1 18 19 20 21 22 23 24 25 26 27	Operation 28 29 30 31 32	Factor 2 33 34 35 36 37 38 39 40 41 42	Result Field 43 44 45 46 47 48	Field Length 49 50 51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus	Minus	Zero or Blank	Compare: High 1 > 2	Low 1 < 2	Equal 1 = 2	Comments 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
0 1	C		14			FLDA	ADD	FLDB	FLDD	70									
0 2	C		14			FLDD	SUB	FLDC	FLDD	70									
0 3	C		14			FLDD	ADD	TOTE	TOTE	80									
0 4	C	L0	16			TOTE	ADD	TOTF	TOTPCH	80									
0 5	C	L1				TOTE	ADD	TOTF	TOTF	80									
0 6	C	L2				TOTF	ADD	TOTG	TOTG	80									
0 7	C	L3				TOTG	ADD	FINTOT	FINTOT	80									
0 8	C																		

Figure 17. Summary Punching Example, Calculation Specifications

The entries required for the punching of the accumulated values are illustrated on the Output-Format Specifications form in Figure 18. The specifications required for this operation are circled. The overflow specifications of the former example have been omitted to make the example easier to read. The cards to be punched will become a new output file. Therefore, the combined file is given the name DETLABOR (line 12, columns 7-14 of Figure 18).

The T in column 15 indicates that the operation is to be performed at total time

The entry in Output Indicators (columns 23-25) indicates that the operation is to occur at the time the blank card is encountered.

To identify the card as a summary card, an 8 is to be punched in column 35. This is specified on line 13. The remaining three fields are specified as described before. The format of the summary card is shown in Figure 19.

TESTING TO DETERMINE A ZERO, POSITIVE, OR NEGATIVE CONDITION OF THE FIELD STATUS

During the execution of an object program, operations are performed in the same sequence in which the specifications are written on the specifications form, unless a different sequence is specified.

IBM

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 04

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	DAILYRPT	D		1				14								
02	O										DIVSON			4			
03	O										DEPT			10			
04	O										EMPNO			17			
05	O										NAME			33			
06	O										FLDA	Z		40			
07	O										FLDB	Z		48			
08	O										FLDC	Z		56			
09	O										FLDD	Z		64			
10	O	DAILYRPT	T		3				L1								
11	O										TOTE	Z	B	66			
12	O	DETLABOR	T						16								
13	O													35	1	8	
14	O										DIVSOM			4			
15	O										DEPT			8			
16	O										TOTPCH			79			
01	O	DAILYRPT	T		3				L2								
02	O										TOTF	Z	B	66			
03	O	DAILYRPT	T		3				L3								
04	O										TOTG	Z	B	66			
05	O	DAILYRPT	T		0				LR								
06	O										FINTOT	Z		66			
07	O																

Figure 18. Summary Punching Example, Output Format Specifications

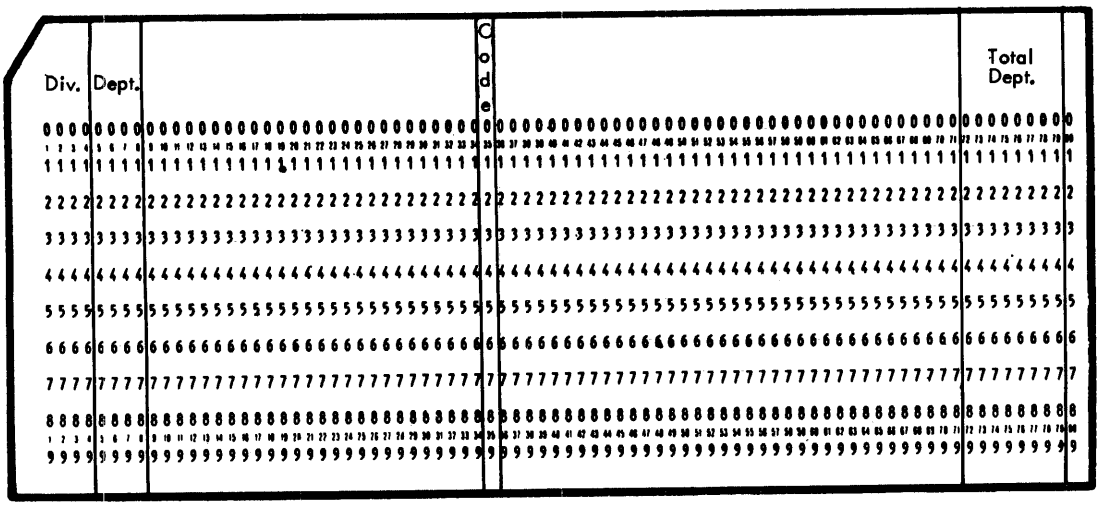


Figure 19. Summary Punching Example, Summary Card Format

If specifications had to be followed sequentially in a fixed pattern, a program would follow a single path of operation without any possibility of dealing with predefined exceptions to the procedure. Moreover, the program could not be coded to follow a predefined alternative to the procedure, based upon conditions encountered during the execution of the program.

For example, assume that a program contains 5 consecutive specifications that are known to be meaningless whenever the quantities they refer to are zero. In this case, the execution time could be reduced if the program could be made to bypass these specifications whenever it detects that the quantity to be processed is zero.

An entry on the Input Specifications form can be used to test a quantity and, depending upon the value of that quantity, set an indicator that will direct the program to some other specification. Three types of tests can be made on the Input Specifications form:

1. Testing an input field to determine if it contains a positive value.

2. Testing an input field to determine if it contains a negative value.

3. Test an input field to determine if it is blank or contains zero.

Three types of tests can also be made on the Calculation Specifications form to determine whether the result of a calculation is positive, negative, or zero.

Example

Figure 20 illustrates a test to determine if FLDA contains zero. Figure 22 shows the specifications for a test to determine if FLDD (as the result field) contains a negative value.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350 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-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	DETLABOR	AA			14	35		D	5	80	N	Z	-					2												
02	I																				1	4		DIVSON	L3						
03	I																				5	8		DEPT	L2						
04	I																				9	14		EMPNO	L1						
05	I																				15	28		NAME							
06	I																				46	50		FLDA						18	
07	I																				56	60		FLDB							
08	I																				66	70		FLDC							
09	I																														

Figure 20. Specifying a Test for a Zero Balance

IBM

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 0 5

Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And Not	Indicators: And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Resulting Indicators: Compare High 1>2	Resulting Indicators: Compare Low 1<2	Resulting Indicators: Compare Equal 1=2	Comments
01	C		14N18		FLDA	ADD	FLDB	FLDD	7	0								
02	C		14N18		FLDD	SUB	FLDC	FLDD	7	0								
03	C		14N18		FLDD	ADD	TOTE	TOTE	8	0								
04	C	L1			TOTE	ADD	TOTF	TOTF	8	0								
05	C	L2			TOTF	ADD	TOTG	TOTG	8	0								
06	C	L3			TOTG	ADD	FINTOT	FINTOT	8	0								
07	C																	

Figure 21. Calculations Based on a Test for a Zero-Balance

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 05

Program Identification: REPORT

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And Not	Indicators: And Not	Indicators: And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1>2	Compare: Low 1<2	Compare: Equal 1=2	Comments
010	C		14			FLDA	ADD	FLDB	FLDD	7	0								
020	C		14			FLDD	SUB	FLDC	FLDD	7	0						(19)		
030	C		14	N18		FLDD	ADD	TOTE	TOTE	8	0								
040	C		18	14			Z-ADD	0	FLDD	7	0								
050	C	L1				TOTE	ADD	TOTF	TOTF	8	0								
060	C	L2				TOTF	ADD	TOTG	TOTG	8	0								
070	C	L3				TOTG	ADD	FINTOT	FINTOT	8	0								
080	C																		

Figure 22. Testing for a Minus Condition

Specifications: Figure 20 illustrates a typical test of the contents of an input field. To make this test, the programmer places one of the numbers from 01 up to 99 in Field Indicators (columns 69-70). In this example, the indicator 18 is used. If a card that is read into the system contains zeros or blanks in columns 46-50 (FLDA), indicator 18 is turned on. Thus, a special condition has been created.

The program must consider this condition during the execution of calculation and/or output specifications. This indicator condition has the same effect as having a resulting indicator turned on by a specific record identification code from an input record.

The entries circled in Figure 21 are the additional specifications that bypass the associated calculations for detail cards that contain zeros or blanks in FLDA. The specifications 14N18 in columns 10-14 of lines 010-030 cause the calculation to be performed if indicator 14 is on and indicator 18 is not on. (The N in column 12 stands for Not.)

The same indicator specification, N18, can also be used on the Output Format Specifications form to prevent the printing of detail cards when FLDA is zero, if that is a requirement of the program.

Example

The calculation specifications in Figure 22 illustrate the use of a field indicator in testing a value to determine if it is negative. The result of this text can be used to bypass certain specifications and/or to process other specifications whenever the tested field contains a negative value.

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 into FLDD. Line 02 specifies that FLDC is to be subtracted from FLDD and that the result is to be placed into FLDD. In addition, line 02 causes FLDD to be tested to determine if its contents are negative. It is assumed in this example that, if the value in FLDD is negative, the calculations involving FLDD are meaningless and, therefore, should be bypassed. If FLDD contains a negative value, the following action is required:

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

2. The contents of FLDD must be reset to zeros. (This step is required whenever the contents of a result field are used in a subsequent operation in which a

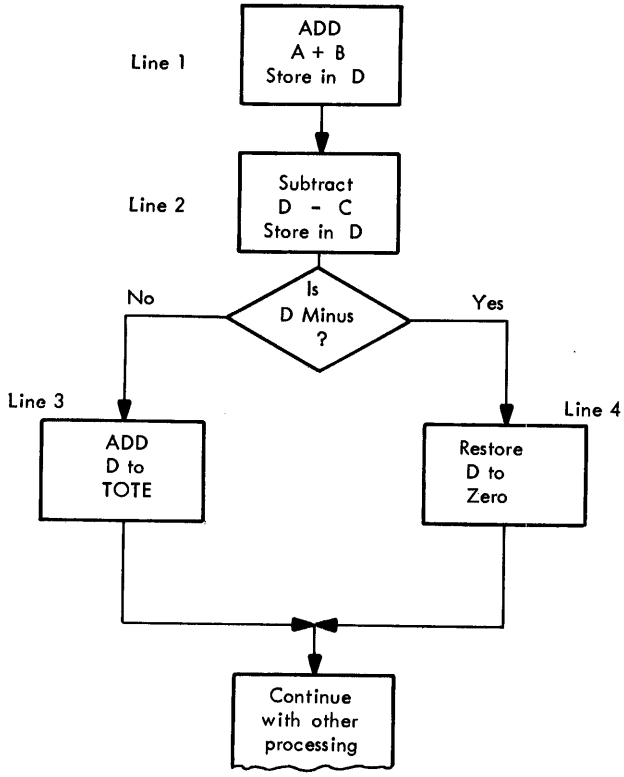


Figure 23. Testing Indicators to Govern Processing

negative quantity would produce incorrect results.)

This is done on the specifications form (see Figure 22) by placing an indicator (code 19) into Resulting Indicators (columns 56-57). Indicator 19 is turned on whenever FLDD contains a negative value.

The operation of adding FLDD into TOTE is specified on line 03. It is performed only if indicator 19 is off. This is specified by entering N19 in columns 12-14.

The specifications on line 4 cause FLDD to be reset to zeros whenever indicator 19 is on. This is specified by entering 19 in columns 12-14.

The 0 in Factor 2 on line 4 is a numeric literal and is used to set FLDD to zeros. This is done by the zero-and-add (Z-ADD) operation. A literal is an actual value (constant) to be used in a calculation. The remaining specifications in Figure 22 are the same as those from previous examples.

If the program had required suppression of the printing of detail cards whenever the result of the specification in line 02 was negative, indicator 19 would also have been used on the pertinent Output-Format Specifications form.

COMPARING

The calculation specifications in this example illustrate a comparison of two fields and the subsequent processing

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And 9 10 11	And 12 13 14	And 15 16 17	Factor 1 18 19 20 21 22 23 24 25 26 27	Operation 28 29 30 31 32	Factor 2 33 34 35 36 37 38 39 40 41 42	Result Field 43 44 45 46 47 48	Field Length 49 50 51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus / Minus / Zero or Blank — Compare: High 1>2 / Low 1<2 / Equal 1=2 54 55 56 57 58 59	Comments 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
01	C					FLDA	COMP	FLDB					18	
02	C		18				MOVE	FLDA	HOLD	102				
03	C		N18			FLDA	SUB	FLDB	SAVE	102				
07	C					DATE	COMP	'06-09-65'					19	
08	C													

Figure 24. Example of Using the COMP Operation, Coding

governed by the result of this comparison. The COMP operation is used to compare two fields and to determine whether the contents of the field in Factor 1 are greater than (high), equal to (equal), or smaller than (low) those in Factor 2. The COMP specification turns on the appropriate indicators specified in Resulting Indicators. These indicators can be used to bypass subsequent specifications or to have some specific operations performed whenever a particular condition has occurred.

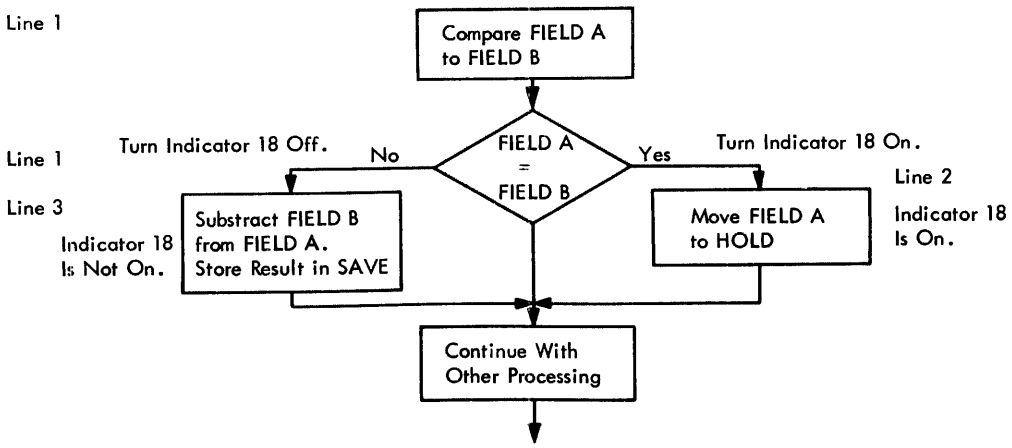


Figure 25. Example of Using the COMP Operation, Logic

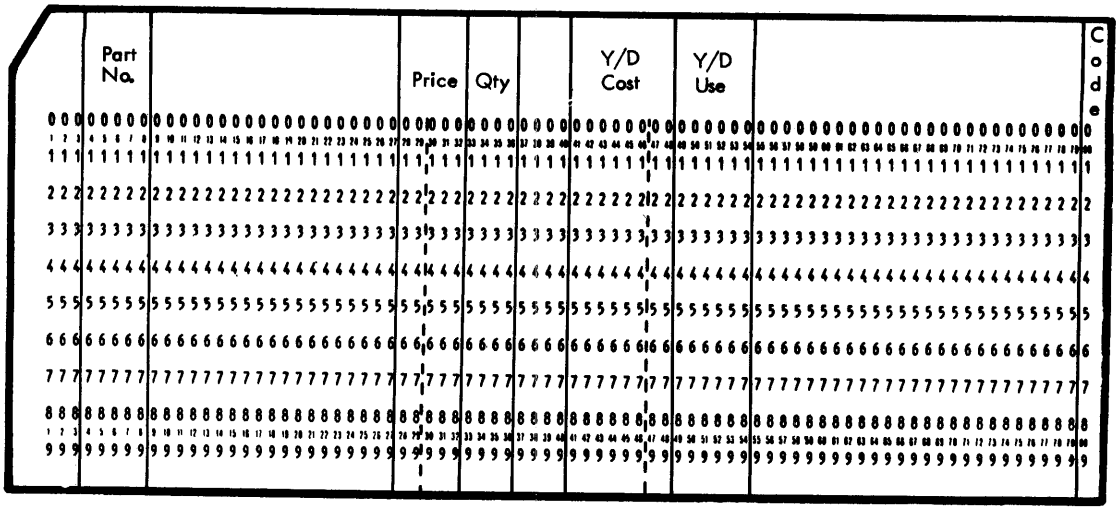


Figure 26. Sample Inventory Card

Specifications: The calculation specifications for a compare problem are shown in Figure 24. The program logic for the problem is shown in Figure 25.

Line 01 specifies that FLDA is to be compared with FLDB. If the two fields are equal, indicator 18 is turned on and the contents of FLDA are placed into HOLD (line 02). If the two fields are not equal (line 03), FLDB is subtracted from FLDA and the result is placed in SAVE.

A literal may also be used in a compare specification. In the lower half of Figure 24, the contents of the input field DATE are compared to 06-09-65. Indicator 19 is turned on if they are equal.

MULTIPLYING AND DIVIDING

Example

This example illustrates an inventory problem. The input for this example consists of inventory cards whose format is shown in Figure 26. For each part number, the quantity (QTY) is to be multiplied by the price to obtain the total cost per week.

The second part of the problem is to update the price each week so that the price used in the subsequent weekly calculation reflects the average fabrication cost and scrap losses on a cumulative (year-to-date) basis.

Specifications: The specifications to solve this problem are shown in Figures 27 and 28. Figure 27 shows the input specifications for all five fields. The programmer must be certain of the exact length of each field. He must also know the number of decimal positions within each field. The fields in this example are:

PARTN	xxxxx	No decimals (alphameric)
PRICE	xx.xxx	3 decimals
QUANTY	xxxx.	0 decimals
YDCOST	xxxxxx.xx	2 decimals
YDUSE	xxxxxx.	0 decimals

As shown in Figure 27, the number of decimal positions in each field is entered under Decimal Positions on the Input Specifications form.

The calculation specifications for this example are shown in Figure 28.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 04

Program Identification: REPORT

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	I	INVENTRY	A	A		29	80		Z	1											4	8		PARTN							
020	I																				28	323		PRICE							
030	I																				33	360		QUANTY							
040	I																				41	482		YDCOST							
050	I																				49	540		YDUSE							
060	I																														
070	I																														

Figure 27. Sample Input Specifications

IBM

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 07

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And	Indicators: And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1 > 2	Compare: Low 1 < 2	Compare: Equal 1 = 2	Comments
010	C		29		PRICE	MULT	QUANTY	COST	92		H							
020	C		29		YDCOST	DIV	YDUSE	PRICE	53		H							
030	C																	

Figure 28. Sample Calculation Specifications

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System 360

Form X24-3350
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 06

Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	I	INVOICE	01	1		21	78		C	5																					
020	I																				1	16		ACOUNT							
030	I																				17	20		CUSNO							
040	I																				40	46		ADDRS							
050	I		02	1	O	22	78		C	3																					
060	I																				1	6		SHIPER							
070	I																				28	31		LOCATN							
080	I																				60	66		RTE							
090	I		03	N		23	78		C	2																					
100	I																				1	6		PARTNO							
110	I																				7	10		NUMORD							
120	I																				40	46		DESC							
130	I		04	N	O	24	78		D	8																					
140	I																				1	6		DISCNT							
150	I		(2)	(3)			(1)														28	31		NEWSAL							
	I																				60	66		DUEDAT							

Figure 29. Input Specifications

PRICE is multiplied by QUANTY and the product is placed in the result field named COST. The overall length of the field COST is specified as nine positions. This includes two decimal positions, specified in column 52. The H in column 53 specifies that the product of PRICE and QUANTY is to be half adjusted (rounded).

The following illustrates this operation.

```
PRICE          1.213
QUANTY       x 216
               7278
              1213
             2426
             262.008
```

Since half adjustment is always made in the position to the right of the last decimal position to be retained in the result field, the half adjustment in this example is made to the digit 8. To do this:

```
262.008
+     5
262.013
```

The rightmost decimal position (3) is truncated and the value 262.01 is placed in the result field COST.

The second line of the Calculation Specifications form in Figure 28 provides the specifications for dividing the year-to-date cost by the year-to-date usage. The result is placed in a field named PRICE. The length of the field PRICE is specified as 5, including three decimal positions. The result is to be half adjusted as specified by the H in Half Adjust. The following is an example of the arithmetic of this

operation, using actual values:

$$PRICE = \frac{YDCOST}{YDUSE} = \frac{1739.23}{1296} = 1.3419$$

The result field was specified to have three decimal positions. Therefore, the half adjustment is made to the digit 9 in the rightmost position since this is in the position to the right of the last position to be retained in the result field.

```
  1.3419
+ ____5
  1.3424  (half adjust)
```

The result (1.342) is stored in PRICE.

SEQUENCE CHECKING

Two types of sequence checking can be performed by RPG:

1. Checking the sequence of different record types.
2. Checking the sequence of control groups. (See the section Using the Matching Fields Specification for Sequence Checking.)

Sequence Checking of Different Record Types

This example illustrates the updating of an invoice file that contains from two to four of the following card types for each part number:

Card	Code
Master customer	5 in column 78
Shipped via	3 in column 78
Part number	2 in column 78
Comments	8 in column 78

Figure 29 illustrates the specifications required to check the sequence of a group of four record types. The item numbers in the following text refer to the circled numbers in Figure 29.

1. The record-identification codes for the four card types are specified in the same manner as in the previous example. The C in C/Z/D (column 26) specifies that the entire character punched in the card is to be examined to establish the record-identification code.

   In the specifications for the last card-type (line 130 on the form), a D is written in C/Z/D because there is a possibility that some of these record types may have a zone punch. The D specifies that only the digit punches in the card are examined to identify the card type. Thus, zone punches which could cause unequal comparisons are ignored.

2. The sequence established for the file is determined by the sequence in which the specifications for each record type are written on the form. The numbers must be in ascending sequence. In this example, the digits 01, 02, 03, and 04 are used. For each file, the numbers assigned must begin with 01.

   Alphabetic characters under Sequence in preceding examples indicate that no sequence checking is to take place. Alphabetic specifications (indicating no sequence checking) must always be written before numeric specifications for any one file.

   If a sequence error is detected, the execution of the object program is discontinued.

3. If a numeric specification is provided in Sequence, a specification must be provided in Number.

   On the first two record identification lines, the digit 01 in Number indicates that no more than one record of that type must be present before the next record type is checked. In this example, only one master record and one shipped-via record for each invoice must be present. If there is more than one of either of these records, the program assumes an error in the input file.

   The letter N in Number of the specifications for the last two record types means that one or more records of this type may be present for each invoice. In this example, one or more part number and comments records may be present for each invoice.

   The letter O in Option specifies that the presence of record(s) of this type is optional.

   If the letter O is not specified, at least one record of the particular type must be present. This requirement applies only if Sequence contains a numeric specification.

   While sequence checking of record types is normally done to assure the proper makeup of a control group, the checking is completely independent of control levels. This checking assures only that the record types are in sequence; it does not assure that they all belong to the same control group.

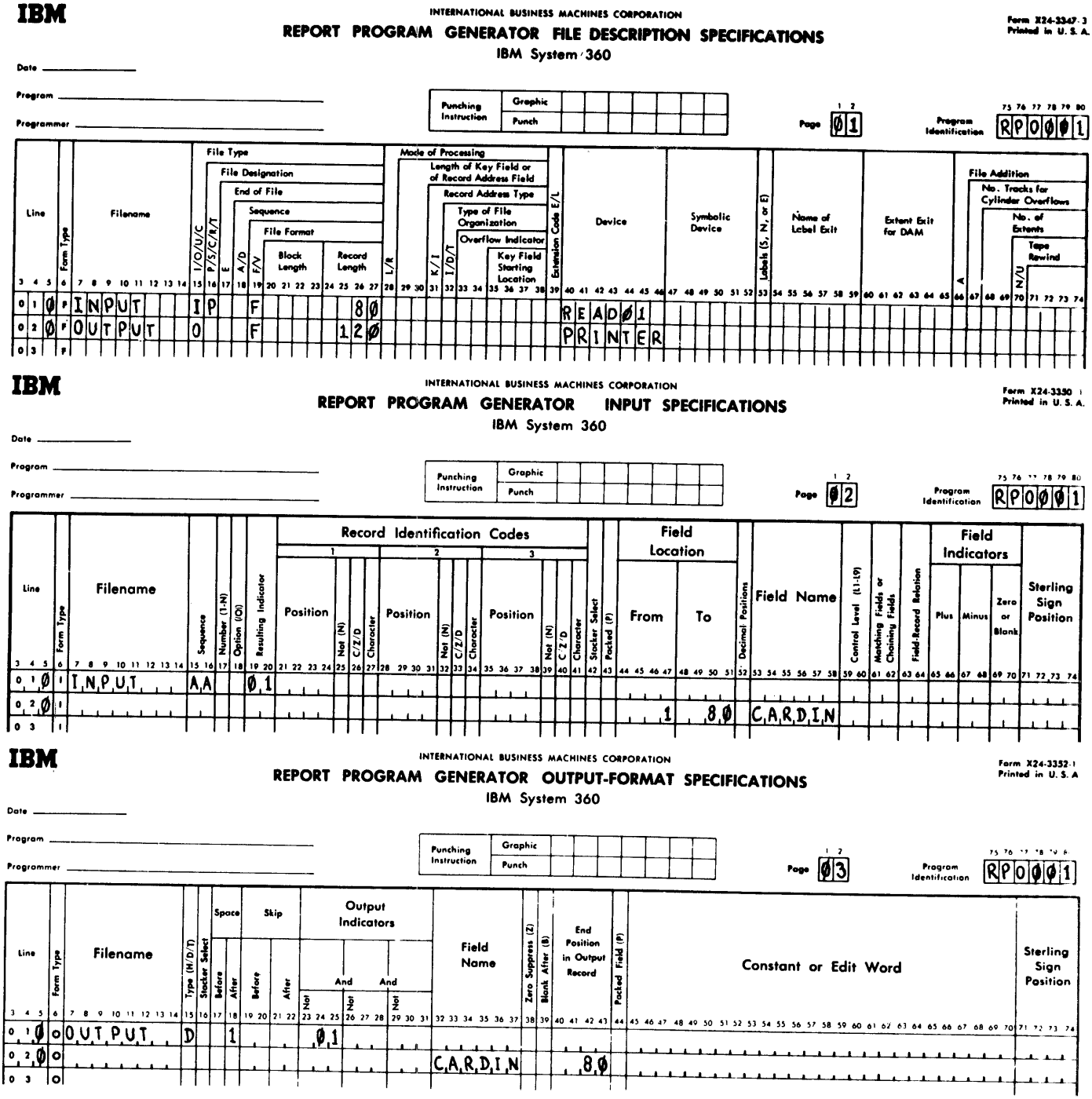


Figure 30. Sample File-to-File Program

CORRELATION OF THE RPG SPECIFICATION FORMS

Figure 30 shows a file-to-file sample program that illustrates the correlation of the RPG specification forms.

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 listed on the printer. Three specifications forms are required for this program:

File Description, Input, and Output-Format Specifications.

FILE DESCRIPTION FORM

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, and the program identification is entered in columns 75-80. An F in column 6 indicates that each entry is a filenames-description entry. The file is entered in columns 7-14 (Filename). Column 15 contains an I or an O to indicate whether the file is an input or an output file.

Column 16 on line 010 of the File Description Specifications form contains a P because the file described is the primary input file for the job.

Column 19 contains an F in line 010 and in line 020 to indicate that the file formats are fixed length. Record length (columns 24-27) is 80 for the card input file and 120 for the printer output file.

The entry READ01 in Device specifies that the input file will be read from an IBM 2501 Card Reader. The Device specification for the output file is PRINTER.

INPUT SPECIFICATIONS FORM

The Input Specifications form also has the program identification entered in columns 75-80 and the page and line numbers in columns 1-5. The I in column 6 of each line identifies this as an input specification.

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

OUTPUT-FORMAT SPECIFICATIONS FORM

On the Output-Format Specifications form, the name of the output file is entered in columns 7-14. The D in column 15 indicates that each line printed is a detail line. A single space after printing is specified by the entry in column 18. Indicator 01 from the Input Specifications form is specified in columns 23-25. Since indicator 01 is on for each card read, all records are printed.

Each object program generated by RPG uses the same general logic. 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, detail time and total time, when operations specified on the Calculation and Output-Format Specifications forms are performed. These instances are called detail time and total time. Exception output lines and calculation subroutines are only performed when required.

Two levels of program logic flow are described here. The first is a simplified version that shows only the most important steps. The second version is more complete and detailed in its explanation of logic flow.

SIMPLIFIED PROGRAM LOGIC

The simplified program logic diagram shown in Figure 30.1 can be used for most discussions of logic flow. A brief discussion of each function follows. The item numbers in this text refer to the numbers in Figure 30.1.

1. RPG begins by performing heading and detail output lines if the conditioning indicators are satisfied. Following this , all control level and record identifying indicatcrs are turned off.

2. If multifile logic is required it is performed at this time. The input record to be processed on this cycle is selected.

3. The record type is determined and the appropriate record identifying indicator is turned on. If control fields are specified, they are checked at this time and the appropriate control level indicators turned on if necessary.

4. Total calculations are performed if the conditioning indicators L0, L1-9, and LR are satisified. Exception output and/or closed subroutines may be performed at this point.

5. Total output lines are performed if the conditioning indicators are satisfied.

6. The data fields described on the Input Specifications for this record type are moved to the fields named. Any field indicators are set at this time.

7. Detail calculations (blanks in columns 7-8 of the Calculation Specifications) are performed if the conditioning indicators are satisfied. Exception output and/or closed subroutines may be performed at this point.

8. If exception output is called for, exception output lines are performed if the conditioning indicators are satisfied. The program returns to the next calculation statement following the EXCPT operation that was performed.

9. If a closed subroutine is called for, the individual calculations within that closed subroutine are performed if the conditioning indicators are satisfied. The closed subroutine may also call other closed subroutines or exception output. The program returns to the next calculation statement following the EXSR operation that was performed.

A detailed flowchart of an RPG object program is shown in Figure 31.

The item numbers in the following text refer to the numbers in Figure 31. A program cycle begins with item 1 and continues through item 25.

1. The object program performs all specified heading and detail output operations whose conditions are satisfied. This does not include output line specifications that are conditioned by the OF or OV indicator.

2. The object program performs a test to determine if a punch in channel 12 of the carriage-control tape was encountered at detail time. If this is the case, the OF/OV indicator is turned on. Otherwise, the indicator is turned off.

3. The object program makes the next input record available. At the beginning of processing, if it is a multifile job, one input record from each file is read into core storage.

4. The object program tests the halt indicators. If the halt indicators are off, the program branches to step 6.

5. The execution of the program is discontinued.

6. All Record Identifying indicators and the indicators 1P, LR, L1 through L9, and H1 through H9 are turned off.

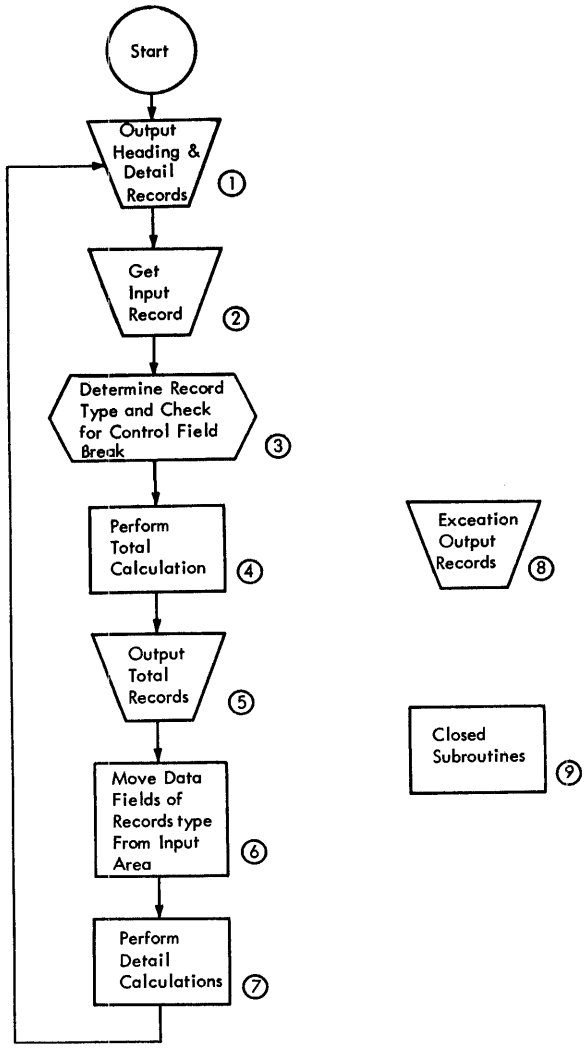


Figure 30.1 Simplified Logic Flow of an RPG Object Program.

Figure 30.1 Simplified Logic Flow of an RPG Object Program.

7. The data is moved from the input area to the input work area.

8. The object program performs a test to determine if the record is an end-of-file record. If an end-of-file condition has occurred, the program branches to step 26.

9. The object program performs a test to determine if the input records are in the sequence specified for them on the Input Specifications form. If the sequence is incorrect, the program branches to step 36. The program also branches to step 36 if nonsequential input records are specified and the record cannot be identified.

10. The object program branches to step 29 if matching fields are specified.

11. The record identifying indicator specified for the current record type is turned on.

12. The object program performs a test to determine if a control break has occurred (i.e., the contents of this control field are not equal to the contents of a previously stored control field). If a control break has not occurred, the program branches to step 14. (The control fields are initially set to hexadecimal zeroes or the lowest possible EBCDIC value.)

13. If a control break has occurred, the appropriate control-level indicators are turned on and the control fields are moved into a save area.

14. If this is the first program cycle, the program bypasses the total calculations and the total output specifications and branches to step 17.

    If no control fields are specified, total calculations and total output lines are bypassed for the first record read.

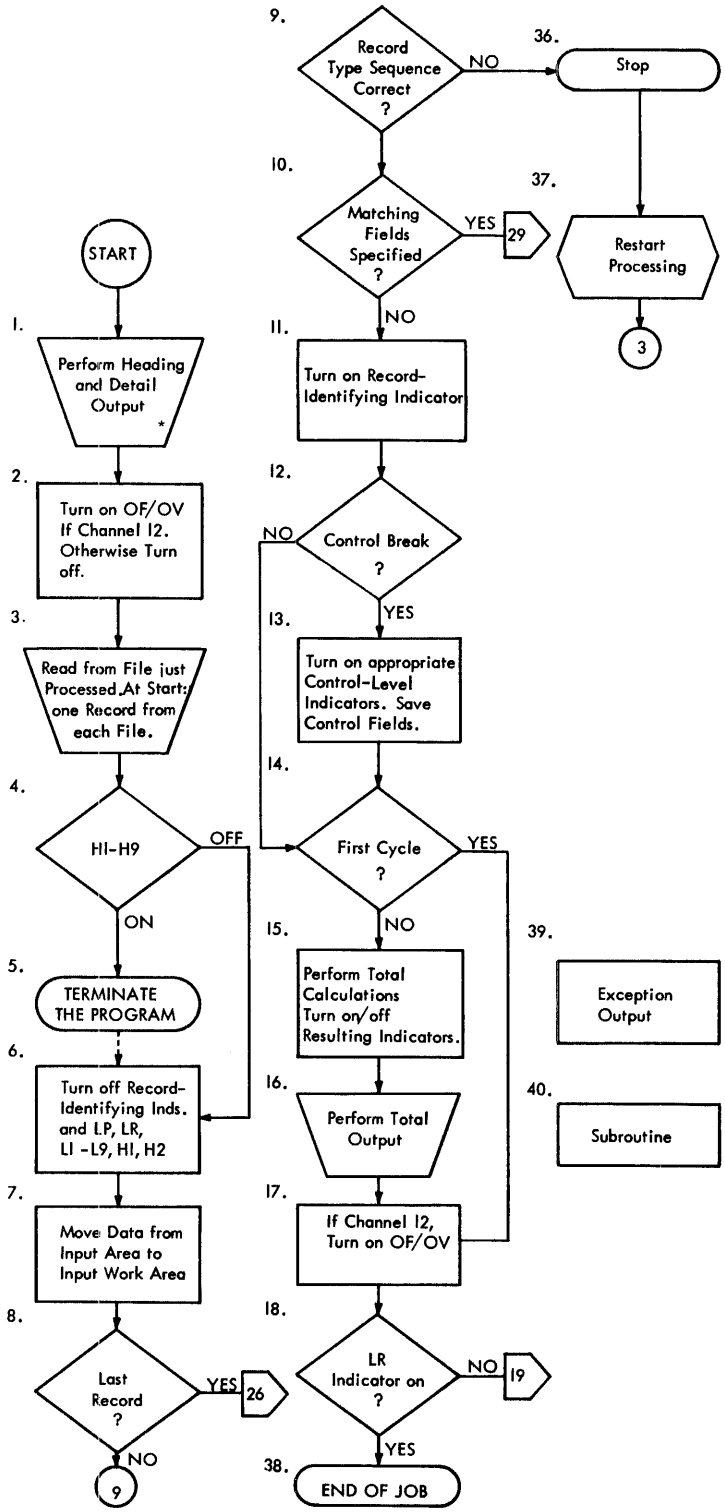
    If control fields are specified, total calculations and total output lines are bypassed for all records read until the first record that contains control field information has been processed. This applies also to the calculations specified with an L0 indicator. If control fields are specified in some and not others, at least one record with control fields should be processed. If not, the LR indicator will not be turned on at end of job.

15. All total calculations are performed and resulting indicators are turned on or off as specified.

    Note 1 If the user specifies the use of a Closed Subroutine, the program branches to the specific subroutine mentioned (step 40) and then returns to the next total calculation.

    Note 2 If the user specifies the use of the EXCPT operation code, the program branches to step 39 and then returns to the next total calculation.

16. Next, all total output that is not conditioned by an overflow indicator is performed.



* refer to Item 1 of Significance of Program Logic.

Figure 31. General Logic Flow of an RPG Object Program (Part 1 of 2)

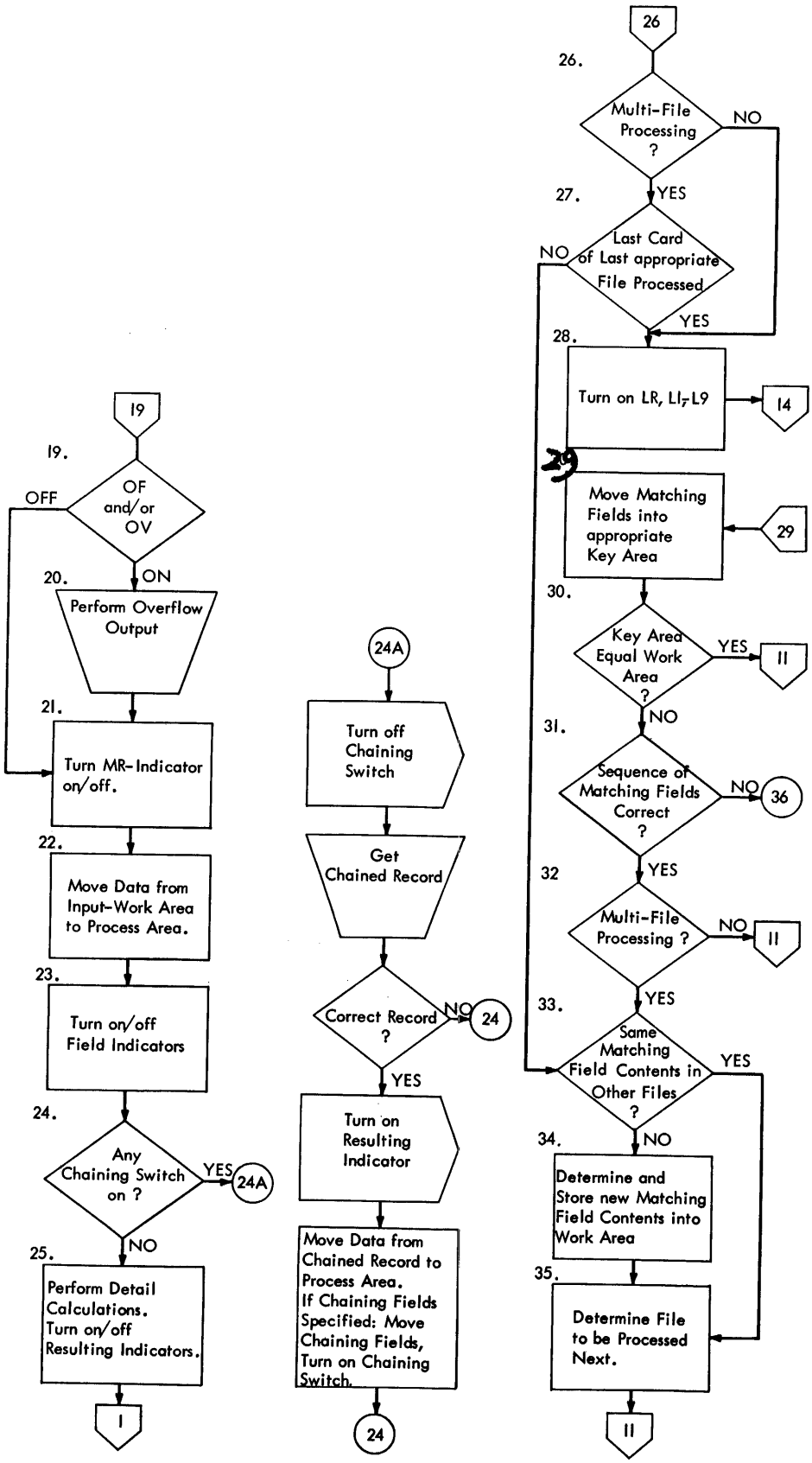


Figure 31. General Logic Flow of an RPG Object Program (Part 2 of 2)

17. The object program performs a test to determine if an overflow condition has occurred (i.e., a punch in channel 12 of the carriage-control tape). If an overflow condition has occurred at any time during this cycle, the indicator (OF or OV) associated with the file that overflowed is turned on.

18. The object program performs a test to determine if the last-record indicator (LR) is on. If the indicator is on, the program branches to step 38.

19. The object program performs a test to determine if one or both indicators, OF and OV, are on. If no overflow indicator is on, the program branches to step 21.

20. The specified overflow output is performed. If no overflow cutput is specified the object program performs an automatic skip until a punch is sensed in channel 1 of the carriage-control tape.

21. The MR indicator is turned on if this is a multifile job and the record to be processed is a matching record. Otherwise, the MR indicator is turned off. If the user has coded a loop from Detail to Total Calculations, the MR indicator will only reflect the correct status the first time Detail Calculations are performed.

22. The input data is moved from the input work area to the process area (i.e., the fields that are specified on the Input Specifications form). No sign checking is performed on numeric fields.

23. Field indicators are turned on or off as specified.

24. If chaining fields are specified, they are moved into the appropriate fields, C1 to C3, and internal chaining switches are turned on by the program. Once all the fields of a chaining record have been moved to the process area, chaining begins with the lowest chaining field (C1). A chained file record may also contain chaining fields (see Figure 31, 24a).

25. Any specified detail calculations are performed, and resulting indicators are turned on or off as specified. Processing continues with step 1.

    Note 1 If the user specifies the use of a closed subroutine, the program branches to the specific subroutine mentioned (step 39) and then returns to the next detail calculation.

    Note 2 If the user specifies the use of the EXCPT operation code, the program branches to step 40 and then returns to the next detail calculation.

26. If only one input file is to be processed, the program continues with step 28.

27. The object program performs a test to determine if the processing of all the files specified with an E in column 17 of the File Description Specifications form has been completed. If not, the program branches to step 33.

    If the primary file is the only file with an E specified, but a matching secondary is present, all records which match the primary will be processed. The LR indicator will then be turned on.

28. All control-level indicators (L1-L9), and the last-record indicator (LR) are turned on and processing continues with step 14.

29. The specified matching fields are moved into the appropriate key area (i.e., the area that contains the control information for the file just processed).

30. Then the contents of this key area are compared to the contents of the matching-field work area. If the contents are equal, the program branches to step 11.

31. The program performs a test to determine if the sequence of matching fields is correct. If the sequence is incorrect, the program branches to step 36.

32. The program performs a test to determine if more than one file is to be processed. If only one input file is to processed, the program branches to step 11.

33. The contents of the key area of the other input files are compared to the contents of the matching field work area. If the contents are equal, the program branches to step 35.

34. Otherwise, the program determines new matching-field contents and moves the lowest or highest contents into the work area depending on whether ascending or descending sequence is specified.

35. The program determines the next file to be processed and branches to step 11.

36. The execution of the program is discontinued because of a sequence or record-type error.

37. Restart processing after elimination of the error condition.

38. The execution of the program is discontinued.

39. Exception output. The program performs all exception output whose conditions are satisfied. No overflow lines may be specified during exception output. The program returns to the next calculation statement in either Step 15 or Step 25 depending upon where it came from.

40. Closed subroutines. All calculations are performed for the called subroutine and any resulting indicators are set. If another subroutine is called within the called subroutine, it too is executed. The program returns to Step 15 or Step 25 depending on where it came from when the originally called subroutine is completed.

SIGNIFICANCE OF PROGRAM LOGIC

Four important conclusions can be drawn from the program logic:

1. At the beginning of execution of the RPG object program, all headings and detail output records whose conditions are satisfied are processed. At this time, the indicators L0 and 1P are turned on. All other indicators are turned off. The user must ensure that only the output conditions for those records that are to be printed as heading information are satisfied before an input record is read into the system. Heading information is usually specified with the indicator 1P. Other heading records must be specified with indicator N1P or with record identifying indicators whose conditions are not satisfied at the beginning of the object program run. As soon as the first input record is read into the system, indicator 1P is turned off. Therefore, all records specified with indicator 1P are not printed or punched again during the object program run. Ensure that the punching of records in a combined file is specified in such a manner that their conditions are not satisfied at the beginning of the object program run. Otherwise, the first card of a combined file is punched prior to being read. The input data contained in this card is lost.

2. When a record is read that causes a control break, the data from the record is not operated upon until after all total calculations and total output for the previous group have been performed. However, the record identifying indicator specified for this record is already turned on.

   Control fields are initialized to hexadecimal zeros. Consequently, a record with a blank or zero punched (hexadecimal F0) control field will cause a control break.

3. If, during the execution of total output specifications, a hole was sensed in channel 12 of the carriage-control tape, all remaining total output specifications are executed whose conditions are satisfied. After the total lines are printed, overflow lines are printed.

   If a hole in channel 12 is sensed during the printing of a detail line, all the remaining detail and total lines specified (whose output conditions are satisfied) are printed before the printing of overflow lines. Therefore, the hole in channel 12 must be correctly placed to permit printing of these lines on the page.

4. If a halt indicator (H1 through H9) specified in Field Indicators on the Input Specifications form or (for detail calculations) in Record Identifying Indicators is turned on, the object program terminates processing immediately after completing this detail cycle.

   If a halt indicator is set on during the processing of total calculations, the object program continues processing until the end of the next detail cycle is reached. The execution of any specifications can be conditioned or prevented by placing H1 through H9 or their negation NH1 through NH9 in Indicators of the appropriate specifications form.

   This termination of the program must not be mistaken for the regular end of job. It is denoted by a different end of job signal.

PROBLEM DEFINITION

The programming examples in the preceding section were intended to introduce the reader to the use of RPG and were, therefore, elementary. More complex applications may require a thorough analysis of the existing cr proposed problem before the

program is written. This analysis should include a description of source data and its format, and the processing required to develop the report and other desired output information.

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

1. Available data.
2. Input/output formats to be used.
3. Information required in the input and output formats.
4. Codes to be used to identify the various types of input and output and their elements.
5. Handling of the various transactions and exceptions.

After all application data has been collected, it should be documented for easy reference during the writing of the specification forms. One method of documenting an application is to produce a layout of the complete format of the report on a Printer Spacing Chart. If the output is tobe punched into cards, card layout should be used instead of a Printer Spacing Chart to define the output fields. These methods provide a pictorial representation of the final output.

PRINTER SPACING CHART

Before the report specifications are written, the programmer should have a clear picture of what he wants as the final output. If the report is to be printed, he must know the number of fields to be printed on each line of the report, the spacing between lines, and the positioning of the information within each line of the report. The pictorial representation in this chart serves as a guide fcr completing the Output-Format Specifications form. It plays an important role in writing report specifications.

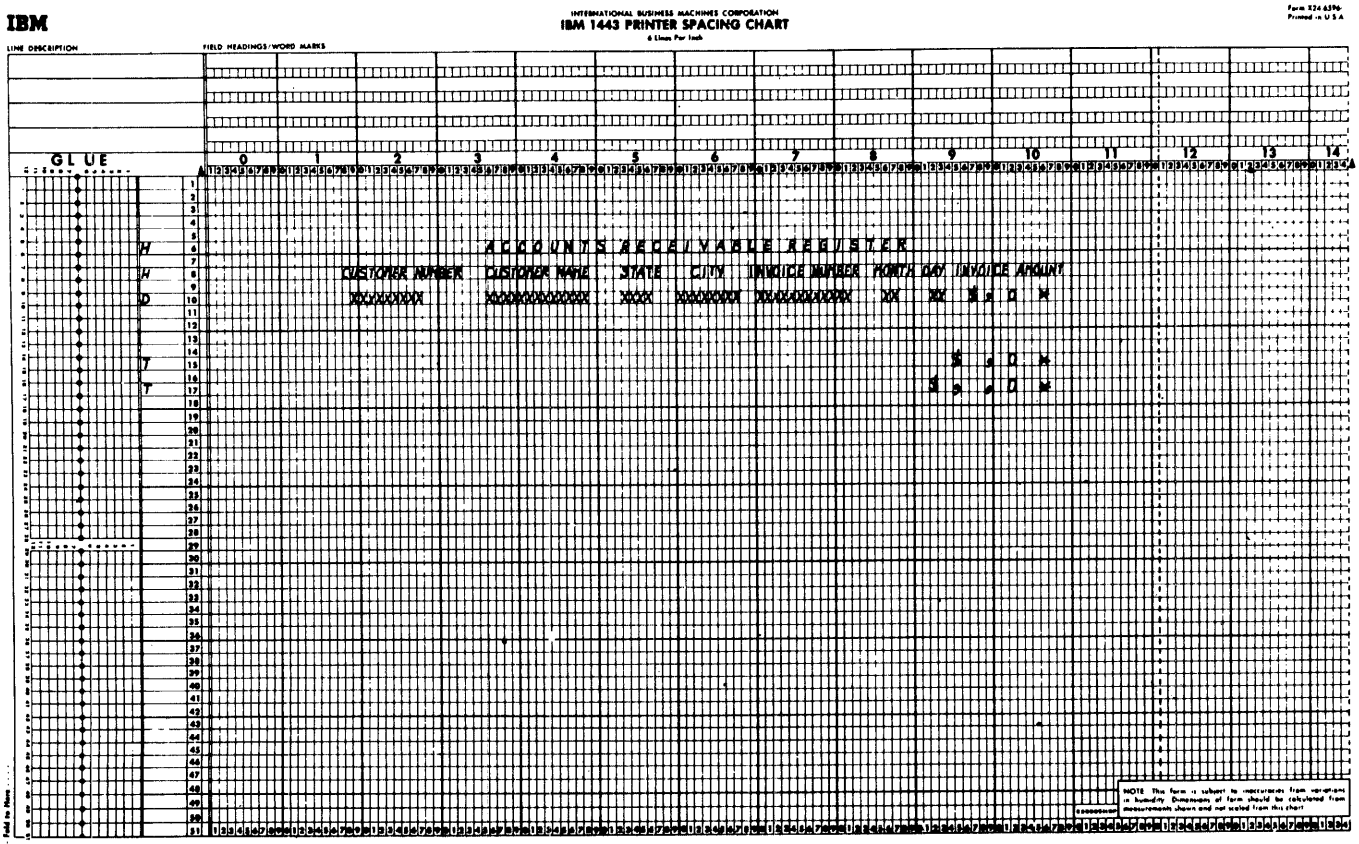


Figure 32. Printer Spacing Chart

Layout of Lines and Fields

Figure 32 illustrates the layout of a report on the Printer Spacing Chart. Its two most important functions are:

1. To establish the positions of the data to be printed and tc indicate the spacing between printed lines.

2. To assign each line an identification code representing the type of line.

The numbers across the top and bottom of the spacing chart refer to 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 should be spelled out completely in the print positions assigned to them. Fields of variable information should be identified by X's. Where applicable, such fields should include credit symbols, punctuation, etc. The position in an amount field where zero suppression ends should be 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, E for an exception line, and T for a total line. All lines should be identified as belonging to one of these categories.

CROSS REFERENCES

To make this reference manual a more effective tool, numerous cross references are included for readers not familiar with disk-storage processing and related functions.

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 (located at the back of the manual) are intended for the reader as he encounters the related specifications for them throughout the manual.

The sections File Organization and File Processing provide a general introduction to disk-file organization and processing, including the terminology associated with these functions.

GENERAL INFORMATION

The forms are discussed in this publication in the sequence in which they are listed below. This is the sequence in which they are normally used by the programmer:

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

Figure 33 shows the five specification forms. In addition to these forms, an RPG Control Card is necessary for each job. Refer to Appendix B for more information about this card.

Input Specifications Form

This form is used to:

1. Specify the input file or files.
2. Identify the various types of records contained in each input file.
3. Describe the location of the data fields in each input record.

Calculation Specifications Form

This form is used to specify the operations to be performed upon the input data, upon data obtained as the result of previous calculations, and upon data obtained from tables.

Output-Format Specifications Form

This form is used to specify:

1. The type of output files to be produced: printed reports, summary reports, etc.
2. The location of the data fields in the output reports and records.

File Description Specifications Form

This form is used to provide additional information about the input/output files and to specify the input/output devices used in the program.

Extension Specifications Form

This form is used to provide additional information about files used by the object program, such as tables, chaining files, and RA (Record Address) files.

COMMON FIELDS

Five entries have the same function in each of the five forms. These five common entries are described as follows.

PAGE NUMBER (COLUMNS 1-2)

Each specification page of the source program may be numbered for reference and sorting convenience. The pages are numbered beginning with 01 and continuing in ascending order as follows:

1. File Description Specifications form
2. Extension Specifications form
3. Input Specifications form
4. Calculation Specifications form
5. Output-Format Specifications form

More than one sheet of a particular specifications form may be used.

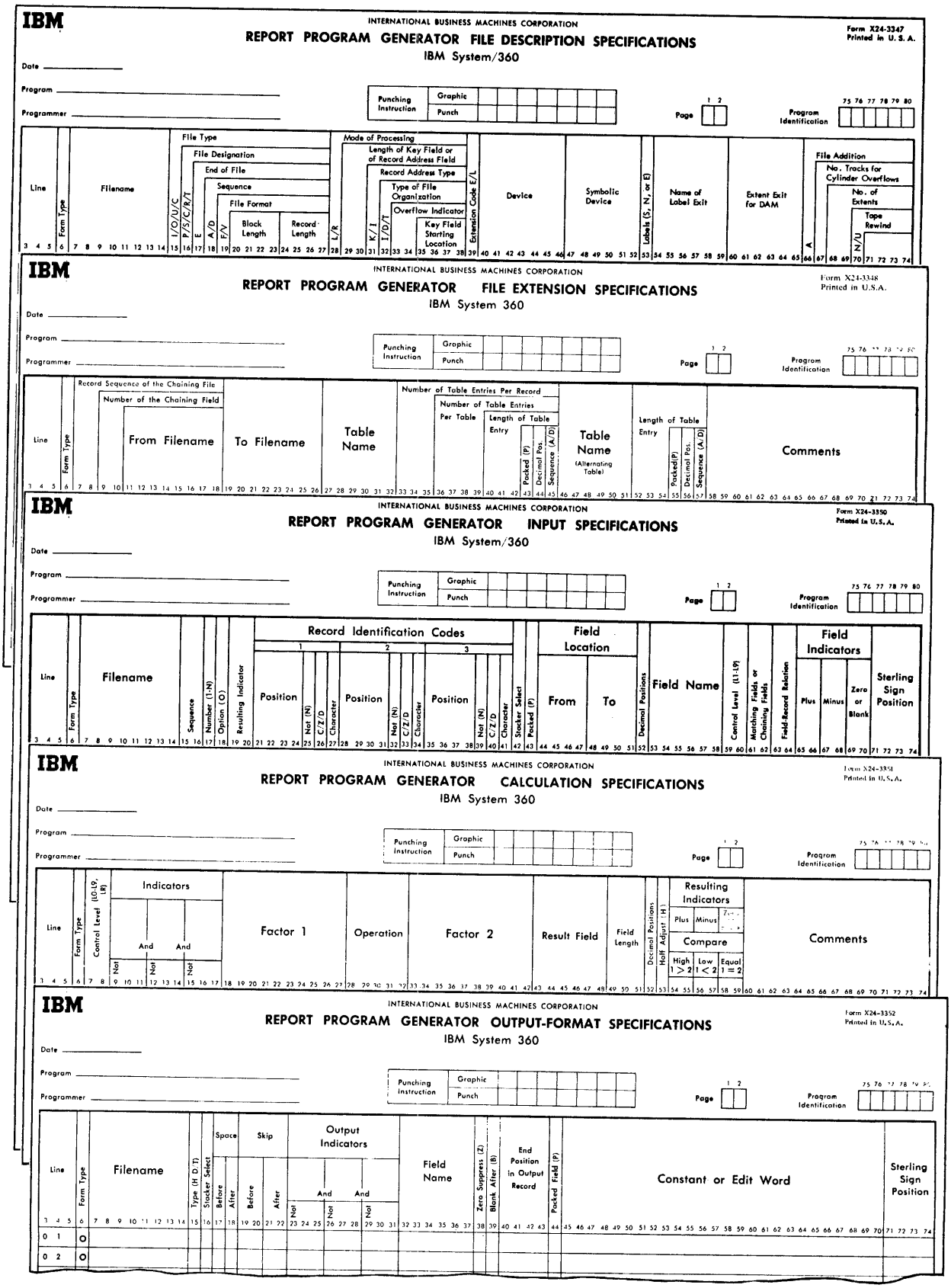


Figure 33. RPG Specification Forms

LINE NUMBER (COLUMNS 3-5)

Each specification line should be identified by a line number. The first two digits of the line number are preprinted on the form. The third position (column 5) is used when, after all specifications have been written, it becomes necessary to insert an additional line between two previously written lines. For example, if a line is to be inserted between lines 14 and 15, it must be given the number 14 in columns 3 and 4 and a subnumber in column 5. This would place the line in the proper numeric sequence.

The cards should be in numeric sequence by columns 1-5 when y are read by RPG. If not, an out of sequence warning notation will be given.

FORM TYPE (COLUMN 6)

Each form has an appropriate type-code preprinted in column 6 which must be punched into all specification cards. The codes are:

I Input Specifications

C Calculation Specifications

O Output-Format Specifications

F File Description Specifications

E Extension Specifications

COMMENTS (ASTERISK * IN COLUMN 7)

This feature enables the programmer to place identifying comments on the specification forms and cards. This facility could be used, for example, to identify the end of one section of a program. The comments are written on the specification line, preceded by an asterisk (*) in column 7. The asterisk in column 7 identifies the contents of a specification line as comments so that they are not used as a specification.

PROGRAM IDENTIFICATION (COLUMNS 75-80)

This entry identifies the specification cards for a particular program or for a specific section of a large program.

The specifications that are made on this form are divided into two categories (Figure 34).

1. Record Identification (columns 7-42). These specifications identify the input record by specifying the identifying record codes it contains. They also specify the relationship of the record to other records in the file. Every RPG program must contain at least one record identification entry.

2. Field Description (columns 43-74). These entries describe the fields of the input record used in the particular application. Each field is described on a separate line below its corresponding record-identification entry.

RECORD IDENTIFICATION ENTRIES

FILENAME (COLUMNS 7-14)

A filename must be assigned to each input file. Use the same name as the one on the File Description form. The name must be left-justified (i.e., it must begin in column 7) and its first character must be alphabetic. It must not contain special characters or embedded blanks. A filename must be specified for the first record identification line of the appropriate file. The maximum length of a filename is eight characters. However, 1130 RPG utilizes only the first five characters for all files. Therefore, it is mandatory that the first five characters provide a unique filename. The same file name may be specified for another record type. All record types for one file must be specified consecutively.

SEQUENCE (COLUMNS 15-16)

This specification designates the sequence of input-record types.

Alphabetic Entries

If the input records are not in a predetermined sequence or if it is not required to terminate processing when the input records are out of sequence, the entry in columns 15-16 should be a two-character alphabetic code.

In Figure 35 the input data records, which contain part numbers and unit numbers, do not appear in a predetermined sequence. Consequently, each input-record type is assigned an alphabetic code because the order of processing is not important to the program.

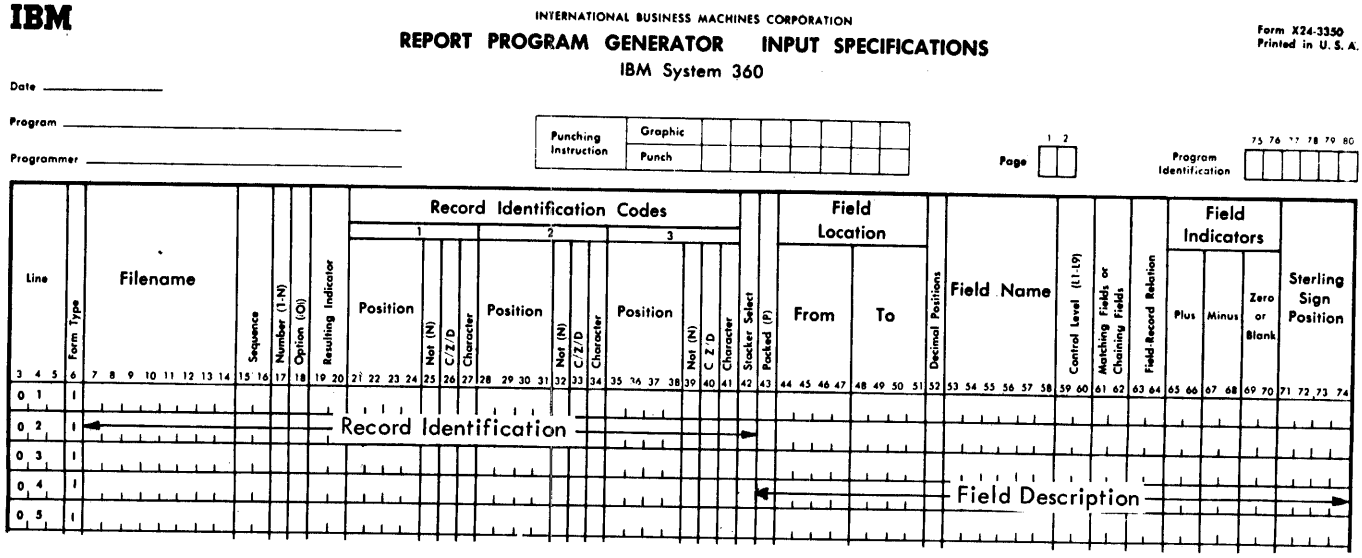


Figure 34. RPG Input Specifications Form

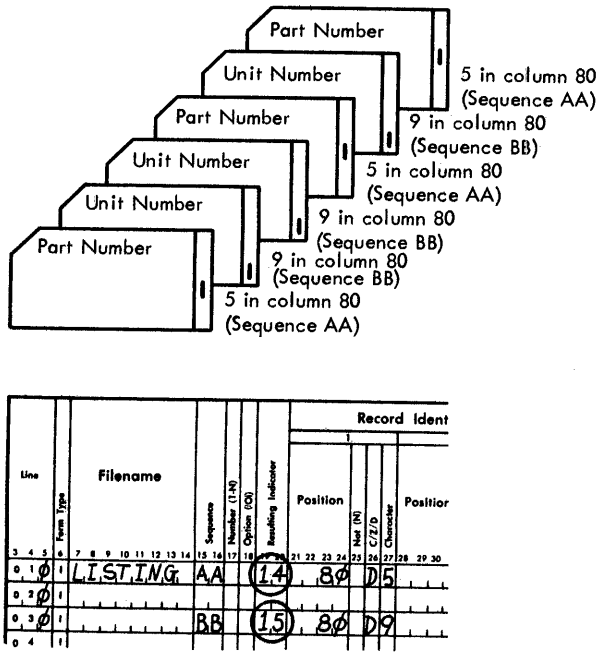


Figure 35. Input Data Records

Numeric Entries

If there is more than one record type within an input file, the records may appear in some predetermined sequence. To process the input data records in a predetermined sequence, they are specified on the Input Specifications form in the same sequence in which they are to be read by the object program. For example, assume that the input records appear as shown in Figure 36. The Name record is assigned 01 because it is the first type. The Street record is the second type, and is assigned 02. The City-State record is assigned 03, and the Item Number Record is assigned 04.

If the input records do not appear in this predetermined sequence, the processing of the program is discontinued (information is out of sequence). Figure 37 illustrates the sequence-checking specifications for this example. Numeric entries must consist of two digits.

Note: If a numeric specification is given in Sequence (columns 15-16), specifications must also be provided in column 17.

Restrictions

- The entries for the checking of a predetermined sequence (columns 15-16) must begin with 01 for each file.
- The entries for the checking of a predetermined sequence must be in ascending order.
- Header cards or other cards that are not in sequence must be specified on the Input Specifications form before any specifications for sequence-checking. Enter alphabetic sequence record-types first, followed by numeric sequence for any one file.

NUMBER (COLUMN 17)

If a numeric code is assigned under Sequence (columns 15-16), a 1 or an N must be entered in Number (column 17). If an alphabetic code has been assigned under Sequence, leave this column blank.

The entries for Number (column 17) are:

1 One and only one record of a type must be present before another record type is read.

N One or more records of a type may be present before another record type is read.

Blank The records do not appear in a predetermined sequence.

In Figure 37, there must be only one Name record, one Street record, and one City-State record, but there may be several Item Number records.

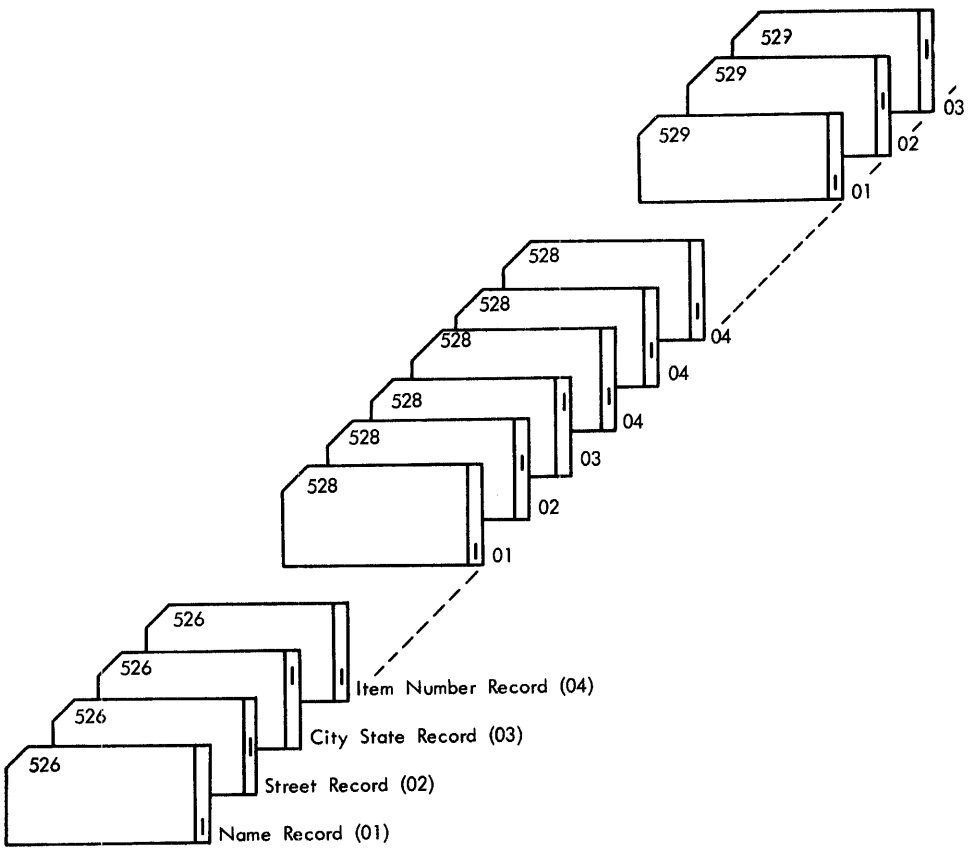


Figure 36. Sample of Input Records in Sequence

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System 360

Form X24-3350
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-N)	Option (O)	Resulting Indicator	Record Identification Codes: 1 Position	1 Not (N)	1 C/Z/D	1 Character	2 Position	2 Not (N)	2 C/Z/D	2 Character	3 Position	3 Not (N)	3 C/Z/D	3 Character	Stacker Select	Packed (P)	Field Location: From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Minus	Zero or Blank	Sterling Sign Position	
0 1 Ø	1	CUSTRECD	Ø1	1		12	8Ø		D	9																						← Name Record
0 2 Ø	1		Ø2	1		13	8Ø		D	5																						← Street Record
0 3 Ø	1		Ø3	1		14	8Ø		D	2																						← City State Record
0 4 Ø	1		Ø4	N	Ø	15	8Ø		D	7																						← Item Number Record
0 5	1																															

Figure 37. Sample of Specifying Sequence Checking (Part 1 of 2)

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
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-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	Ø I	RECORD	Ø1	1		2Ø	8Ø		C1																						
02	I		Ø2	N	O	25	8Ø		C2																						

Figure 37. Sample of Specifying Sequence Checking (Part 2 of 2)

OPTIONAL (COLUMN 18)

If the presence of a record is optional, the letter O is entered in this column. It is not permitted to specify all records of a file as optional. At least one record must be "required". If a specific record type must be present or if records are non-sequential, this column must be left blank.

In Figure 37, the letter O in column 18 indicates there may be one, several, or no Item Number records in each group. In this example, Name, Street, and City-State records (one per type) must be present.

Note: If a numeric entry is specified in columns 15-18:

1. These columns have no relationship with control levels (columns 59-60). If two card types are mandatory, the RPG program will only check that one follows the other. No checking is performed to ensure that they belong to the same control group.

2. If the file contains two record types and one is specified as mandatory and the other optional (see Figure 37 Part 2 of 2), no errors will ever be detected. The program can only perform a valid check when two or more records are mandatory.

RECORD IDENTIFYING INDICATOR (COLUMNS 19-20)

The entry in these columns is used in conjunction with the entry in record identification codes (columns 21-41). The record identifying indicator serves two functions:

1. It establishes a unique two-digit code for each input-record type.

2. A special indicator is turned on in the object program each time the input record is present. Thus, the object program may test for this condition during the processing of the calculation and output specifications.

To understand the first function of this entry, assume that an input record is identified by all of the following codes:

1. An X in column 80.

2. A digit 5 in column 79.

3. No R in column 78.

By assigning one two-digit record identifying code to represent all these codes, it is easier to refer to this type of input record during the writing of the calculation and output specifications.

The second function of record-identifying indicators can be considered analogous to the function of selectors in unit record equipment, or to internal and external switches on electronic data processing equipment. The use of indicators, like the use of selectors and switches, permits the user to have certain operations occur only on specific conditions.

In Figure 37 the four record types have been assigned the record-identifying indicators 12, 13, 14, and 15. When one of the records is present, the appropriate record-identifying indicator is turned on, and all specifications pertaining to this type of record are performed. Specifications associated with other record types are not performed.

Record-identifying indicators are turned on and off during the processing of the object program, as the various record types are read. However, only the record-identifying indicator for one specified record type can be on at a time. When a record-identifying indicator is turned on, all other record -identifying indicators are turned off. This does not apply to record-identifying indicators for chained records.

Other indicator conditions that can be established in the program are field indicators (columns 65-70 of the Input Specifications form) and resulting indicators (columns 54-59 of the Calculation Specifications form). These functions are described later, but one aspect of their use is of interest at this time.

A unique two-digit code (01-99) must be assigned for each indicator used in the program. The indicator codes do not have to be assigned in any sequence. For example, four different record types could be assigned the record-identifying indicator codes 40, 62, 98, 02. The codes 01-99 can be assigned to any one of the three types of indicators. Different types of indicators with the same code are considered to be the same indicator. Therefore, care should be taken when assigning the same code to different types of indicators.

RECORD IDENTIFICATION CODES (COLUMNS 21-41)

These entries provide a means of identifying each different record type used in the object program. As mentioned earlier in this publication, once the record type has been defined on the Input Specifications form, references to the record type are made by its record-identifying indicator.

These columns provide for the entry of one to three identifying codes as indicated by the numbers 1, 2, and 3 on the Input Specifications form. A record type can be identified by specifying more than three codes in more than one line. If only one record type is contained in the input file, record identification codes can be left blank, but a record-identifying indicator and a Sequence entry must still be specified.

The three sets of entries in record identification codes are identical. Therefore, only the first set (columns 21-27) is described. The set consists of four entries as follows:

Position (columns 21-24)

Not (column 25)

C/Z/D (column 26)

Character (column 27)

Position (Columns 21-24)

Enter in these columns the number of the position in a record type than contains the identifying code. This entry must be right-justified. Leading zeros can be omitted.

Not (Column 25)

Enter an N in this column if the code described in columns 26 and 27 is not to be present in the specified position of the associated record type. Otherwise, leave this column blank.

C/Z/D (Column 26)

The object program identifies different types of input records by comparing the character written in Character (column 27) with codes contained in the records. The entry in column 26 specifies whether the object program is to compare against the

- Entire character (C).
- Zone portion (Z).
- Digit portion (D).

Enter a C, Z, or D in column 26, as applicable.

Character (Column 27)

The particular character allowed for record identification (column 27) depends on the particular specification used in C/Z/D (column 26).

1. C--Character

   If this entry is contained in C/Z/D, any one of the 256 characters of the EBCDIC (Extended BCD Interchange Code) may be specified in the Character column.

2. Z--Zone

   The zone portion of the punched card column consists of the 12, 11, hole positions.

   a. Testing card zones
   If the test is to be for a 12-zone, the characters &, or any one of the 16 EBCDIC characters that have the same zone as the letter A may be specified in the Character column. The presence of a 12-zone is recognized only if one of these characters is present in the pertinent position of the record.

   If the test is to be for an 11-zone, the character - (minus sign), or any one of the 16 EBCDIC characters that have the same zone as the letter J may be specified in Character. The presence of an 11-zone is recognized only if one of these characters is present in the pertinent position of the record. If the test is to determine the absence of both the 11 and 12 zones, any one of the 16 EBCDIC characters having the same zone as 1 (no zone) may be specified, or the Character column may be left blank. The absence of zones is recognized only if one of these characters is present in the pertinent record position, or if this record position is blank.

   b. Testing core zones
   Any other specification will identify record codes with the same zone in core storage as the character specified in the Character column of the Input Specifications form.

3. D--Digit

   The digit portion of the punched card column consists of the 1 through 9 hole positions.

   If this entry is contained in C/Z/D, the digit portion of any of the 256 EBCDIC characters can be tested. Any one of these characters may be specified in the Character column.

Note: Using the specification C in C/Z/D saves core storage and reduces the time required for the execution of the object program.

Examples of Record-Identification Codes

Examples of record-identification codes are shown in Figure 38.

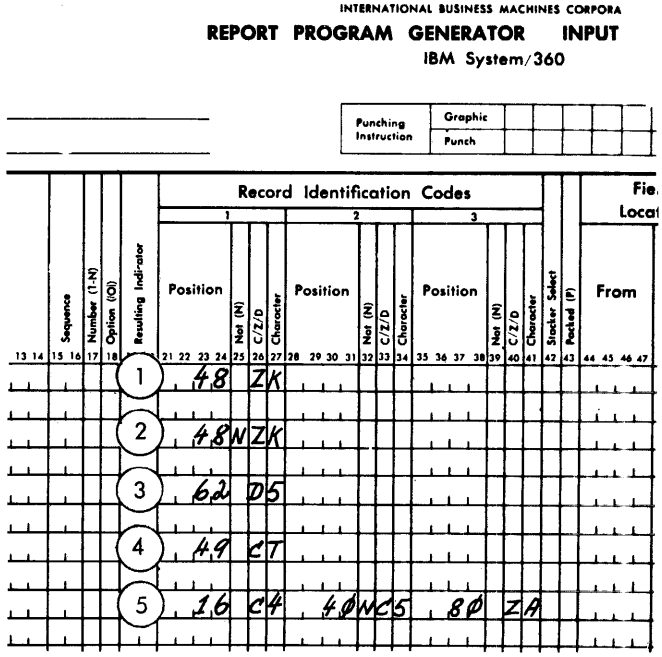


Figure 38. Sample Record Identification Codes (Part 1 of 3)

IBM INTERNATIONAL BU

REPORT PROGRAM GEN

IBM

Date _____

Program _____

Programmer _____

Punching Instruction

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Coc 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	Posi
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
0 1	I	CARDS	AA			Ø1	1		C	5					
0 2	I														
0 3	I		BB			Ø2	1		C	5	2		C	4	
0 4	I														

Figure 38. Sample Record Identification Codes (Part 2 of 3)

IBM INTERNATIONAL BU

REPORT PROGRAM GEN

IBM

Date _____

Program _____

Programmer _____

Punching Instruction

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Coc 1 Position	Not (N)	C/Z/D	Character	2 Position	Not (N)	C/Z/D	Character	Posi
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
0 1	I	INPUT	AA			Ø1	1		C	A					
0 2	I														
0 3	I		BB			Ø2	1		C	B					
0 4	I														
0 5	I		CC			Ø3	1		C	C					
0 6	I														
0 7	I		DD			Ø4	1		C	D					
0 8	I														
0 9	I		EE			Ø5									
1 0	I														
1 1	I														

Figure 38. Sample Record Identification Codes (Part 3 of 3)

The numbers circled in Figure 38 correspond to the item numbers in the following text:

1. This entry causes the specified record-identifying indicator to be set on if the character - (minus sign) or any one of the 16 EBCDIC characters that have the same zone as the letter J is contained in record position 48. A Z must be placed in column 26 so that only the zone portion is checked. (If a C were placed in this column, the object program would compare the letter K, instead of an 11-punch, against the input-record code.)

2. This entry turns on the specified record-identifying indicator whenever an input record is read that does not contain the character - (minus sign) or any one of the 16 EBCDIC characters that have the same zone as the letter K in position 48.

3. This entry turns on the specified record-identifying indicator if a character whose digit portion is 5 is contained in record position 62 (e.g., a 5, N, V, or E would meet this requirement).

4. This entry turns on the associated record-identifying indicator if the letter T is contained in record position 49.

5. These entries specify that three conditions must be met in the one input record. This is referred to as an AND relationship. In the example, the associated record-identifying indicator is turned on if a 4 appears in column 16 of the input record, column 40 does not contain a 5 and column 80 contains the character & or any one of the 16 EBCDIC characters has the same zone as the letter A.

Figure 39 illustrates the use of more than one code on more than one line to identify one type of input record. In this example, the specified record-identifying indicator is turned on if an input record is read that contains all of the following:

- A 4 in position 16.
- No 5 in position 40.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3390-1
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 04

Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	1	PAYREPT	AA			26	16		C	4	40	N	C	5	80		Z	K													
02	1	AND					20		C	2	25		C	3																	

Figure 39. Example of Specifying Record Identification Codes in an AND-Relationship

- An 11-punch in position 80.
- A 2 in position 20.
- A 3 in position 25.

Additional specification lines can be used to specify as many record-identification codes as required. Each additional line must begin with the word AND in columns 14-16. Columns 17-20 of each AND line must be blank.

An AND relationship is concerned with specifying more than three record identification codes for one record type, whereas an OR relationship is used to specify two different record types with just one set of field description specifications. The fields in the two record types may be in the same or in different positions.

To specify an OR-relationship, enter the word OR in columns 14-15. Columns 16-18 of OR lines must be left blank. For more detailed information, refer to the section Records in an OR-Relationship.

Omitting Record-Identification Codes

When all input records are to be processed alike, columns 21-41 may be left blank. However, a sequence must be specified.

If the different record types are not in a predetermined sequence and if only certain record types within a group are to be processed, they must be listed first with their record identification codes. The remaining record types can be bypassed or processed as a group. In either case, they are specified as the last record type and they must be assigned a sequence. Columns 21-41 of such lines are left blank. If the records are to be bypassed, they should not be referred to in the calculation or output specifications.

Duplicate Identification

If the program reads a record which satisfies two or more record types only the first will be recognized. In Figure 38 record identifying indicator 02 will never be turned on because the first record type (AA) is satisfied.

Undetermined Record Type

If the program cannot detect a record type, a halt will occur and the operator will be given a choice of continuing with the next record. In many applications, the user may wish to program for undetermined record types and process them differently. This can be accomplished by defining a record type with no record identification. This record type must be specified following all valid types as shown in Figure 38. Record identifying indicator 05 is turned on whenever an undetermined record type is read.

STACKER SELECT (COLUMN 42)

If the input records are cards that are to be stacker-selected, the number of the stacker into which the cards are to be selected is entered in column 42. Leave column 42 blank if the cards are to go to the normal stacker.

A different stacker may be specified for each OR line. If only one stacker is required in an OR-relationship, this stacker must be specified on each of the OR lines.

For the IBM 1442 Card Reader, the normal stacker, 1, may be specified with either a 1 or a blank in column 42. A 2 in column 42 will specify stacker number 2.

Note: If two I/O areas are specified for the file (by entering 1-9 in column 32 of the File Desciption Specification of this file), stacker select will be ignored. Using two I/O areas requires that a second card be read before stacker selection can occur for the first card.

FIELD DESCRIPTION ENTRIES

As mentioned earlier in this publication, the Input Specification form consists of two categories of entries: record identification and field description entries.

In the record identification category of entries (columns 7-42), a line contains the specifications for one record type. In the field-description category (columns 43-74), one line represents the specifications for one field of an input record type.

Field description specifications are always written on the line immediately below the specifications that identify the record type.

Field description entries describe those fields of the input record that are to be used in the application. Each field of the record requires one line on the Input Specifications form. Columns 7-42 of each field-description line must be left blank.

PACKED (COLUMN 43)

The packed decimal format may be used only with disk files. If the input is in unpacked decimal format, leave this column blank.

Enter a P in this column if the numeric input field is in packed decimal format (two decimal digits represented in one record position.)

The packed decimal format permits greater utilization of disk storage capacities. Thus, the RPG program permits numeric data in packed decimal format to be written onto disk for subsequent processing runs.

FIELD LOCATION (COLUMNS 44-51)

Columns 44-51 of the Input Specifications form are used to define the location of a field within the input record. The maximum length of a numeric field is 14 digits. The maximum length of an alphameric field is 256 characters.

From (Columns 44-47)

This entry specifies the location of the leftmost position (i.e., the high-order position) of the field. The entry in columns 44-47 must be right-justified; leading zeros may be omitted.

To (Columns 48-51)

This entry specifies the location of the rightmost position (the low-order position) of the field. The entry in columns 48-51 must be right-justified; leading zeros may be omitted. In Figure 40, the field named DEPT is located in positions 5-8 of the input record.

The fields of an input record may be described in any sequence. In Figure 40, they are shown in numerical sequence (by field location) to make the example easier to understand.

A one position field may be defined by placing its location into both From and To fields.

A field location may be defined twice with two different names. It is also possible to define fields within fields (e.g., CUSTMR in positions 1 to 27 with NUMBER in positions 1 to 10 and NAME in positions 11 to 27). Two or more fields may also overlap each other.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System 360

Form X24-3350 Printed in U.S.A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 05

Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	1	REPORT	AA			14	35		C	5	80	N	Z	K																	
020	1		OR				35		C	6																					
030	1																				1	4		DIVSON							
040	1																				5	8		DEPT							
050	1																				9	14		EMPNO							
060	1																				15	28		NAME							
070	1																				46	50	0	FLDA							
080	1																				56	60	0	FLDB							
090	1																				66	70	0	FLDC							
100	1																				71	71		FLDE							

Figure 40. Example of Specifying Input Fields

Records in an OR-Relationship

In many applications, records may contain almost identical fields or the same record type may be denoted by two or more record identification codes. This could be described to RPG as two entirely different record types. This however, will cause the programmer to describe the same information two or more times and additional object core requirements. To simplify this, RPG provides for records in an OR-relationship. It is possible to use one set of specifications to describe fields that are contained in two or more different records in the same locations. In Figure 40, OR has been entered in columns 14-15. Columns 16 to 18 must be blank. This specifies that the fields described in lines 030-100 are contained in input records that have a 5 in column 35 and no 11-punch in column 80, and in input records that have a 6 in cclumn 35.

Figure 40 shows that identifying indicator 14 need not be entered in the OR line because it has been previously specified. In OR-relationships, record identifying indicators need not be respecified. However, a different record identifying indicator may be specified in each OR line.

It is also possible to describe two or more record types by means of an OR-relationship even if the fields are not in the same locations. This is illustrated in the example shown in Figure 41. In this example, record identifying indicator 14 is turned on whenever an input record contains a 5 in column 35 and no 11-punch in column 80. Record identifying indicator 16 is turned on whenever an input record contains a 6 in column 35. The OR-relationship makes it possible to describe both record types by means of one set of field specifications.

In this example, the field locations are not the same in both record types. Therefore, a separate record identifying indicator code must be specified for the second record type.

The OR line indicates there is a second type of input record that contains the required information. However, field FLDC is not contained in the same locations. For this reason, the field FLDC must be described twice. Each of the two entries for FLDC must be related to the appropriate record type. This is accomplished by entering the pertinent record identifying indicator in Field-Record Relation (columns 63-64). (An indicator entered in Field-

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350 Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 04

Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
030	I	REPORT AA				14	35	C	5		80	N	Z	K																	
040	I	OR				16	35	C	6																						
050	I																				1	4		DIVSON							
060	I																				5	8		DEPT							
070	I																				9	14		EMPNO							
080	I																				15	28		NAME							
090	I																				46	50	0	FLDA							
100	I																				56	60	0	FLDB							
110	I																				66	70	0	FLDC		14					
120	I																				61	65	0	FLDC		16					

Figure 41. Example of Using Field Record Relation Indicators

Record Relation is referred to as a field-record relation indicator.)

Thus, if record identifying indicator 14 is on, FLDC is taken from locations 66-70 of the input record. If record identifying indicator 16 is on, FLDC is taken from locations 61-65 of the input record.

The use of an OR-relationship is not restricted to two record types. Any number of record types may be specified.

DECIMAL POSITIONS (COLUMN 52)

Any entry (0-9) in this column defines the associated field as a numeric field. Each entry in Decimal Positions:

- Is used to determine the number of decimal positions contained in the field specified in Field Location.
- Causes all zones of the field specified in Field Location to be ignored during comparisons caused by a control-level or matching-fields specification.

Enter in column 52 the digit that represents the number of decimal positions in the input data field. If the input data field contains a whole number, enter a 0 in this column. If the field contains data to be treated as alphameric data, leave this column blank. An entry must be made in Decimal Positions if a P has been specified in column 43 or if the field is to be used in arithmetic or editing operations. The number of decimal positions cannot exceed the field length described in columns 44-51.

In Figure 41, the fields named DIVSON, DEPT, EMPNO, and NAME are alphameric fields because no entry has been made in column 52. The fields FLDA through FLDC are numeric fields but they can contain only whole quantities without decimal fractions because a zero is entered in column 52 for these fields.

FIELD NAME (COLUMNS 53-58)

Each field defined must be given a field name. Once a name has been assigned to a field, it can be referred to in calculation and/or output-format specifications.

The field name must begin with an alphabetic character in column 53. The field name may be alphamericand may be from one to six characters in length. Special characters and embedded blanks must not be used.

Note: The same field name may be used for any number of fields of different records, provided that all these fields have the same length and the same number of decimal positions as the first field given this name. A field name is assigned only once by RPG. The same storage location is used for fields with identical names.

Field names beginning with PAGE can only be used for page numbering. Field names beginning with IN cannot be used in an RLABL statement (for a user written exit) as this is reserved for indicators. The word ALTSE must not be used as a field name, nor can field names start with the letters TAB.

Figure 42 shows a number of sample field names that are easy to read and suggest the function of the fields they represent.

Page (1-2): Ø4 Program Identification (75-80): REPORT

C/Z/D (40)	Character (41)	Stacker Select (42)	Packed (P) (43)	Field Location From (44-47)	Field Location To (48-51)	Decimal Positions (52)	Field Name (53-58)	Control Level (L1-L9) (59-60)	Matching Fields or Chaining Fields (61-62)	Field-Record Relation (63-64)	Field Indicators Plus (65-66)	Field Indicators Minus (67-68)	Field Indicators Zero or Blank (69-70)	Sterling Sign Position (71-74)
				1	4	Ø	DIVSON	L3						
				5	8		DEPT	L2						
				9	14		EMPNO	L1						
				15	28		NAME							
				29	31		ORDRNO							
				33	4Ø		BALNCE							
				41	43		PARTNO							
				49	55		FIELD1							
				56	6Ø		FIELD2							
				61	69		FIELDA							
				7Ø	74		CUSTNO							
				76	79	2	AMTDUE							

Figure 42. Example of Field Name and Control Fields

Defining the Same Data Field as Alphameric and Numeric

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

Using Input Data Fields for Constant Data

Constants are 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 method of supplying this information to the program is by defining a special record type and by assigning a field name that is not otherwise used. This technique also permits the entering of constants that are too large to be specified as literals in the source program.

When only disk files are being processed, specifications for an additional card file are required for the constant information. The file must be specified as follows:

1. In multifile programs using the matching technique, the additional card file must be designated as a secondary file without matching fields. Thus, the additional card file will be processed immediately as the cards are read.

2. In programs that do not use the matching technique (e.g., single-file programs or programs using the chaining technique) the additional card file must be designated as a primary file. For detailed information about matching and chaining techniques, refer to the appropriate sections.

CONTROL LEVEL (COLUMNS 59-60)

A change in a control field causes all processing indicated by this change to be initiated as well as the processing indicated by all lower control levels. 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 with RPG. These levels are designated by the indicators L1, L2, L3,...L9. level 1 is the lowest control level; level 9 is the highest. In RPG, a control break at one level forces control breaks for all lower levels. An indicator similar to the record identifying indicator is associated with each control-level designation. These control-level indicators are used to control functions specified on the Calculation and Output-Format Specifications forms. When the field specified in Field Location is a control field, its appropriate control level must be specified in columns 59-60. The example in Figure 42 shows the entries for three control fields.

The field located in positions 1-4 of the input record (first entry in the example) is the highest level of control.

Indicator L3 can be used to specify when the calculations for this control field are to take place, or when the totals for this field are to be printed or punched. More detailed information regarding the use of the control-level indicators is presented in the descriptions of the Calculation and Output-Format Specifications forms.

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.

The user has access only to the second area. This can be done by specifying the field name on the Calculation Specifications or the Output-Format Specifications form.

It may be required to use a field for controlling functions but without considering zone bits in the internal comparison of one record to the next record. If the zone bits are not to be used in an internal comparison of control fields, specify the field as numeric by making a decimal entry (0-9) in column 52. Decimal alignment is not performed for numeric control fields. If the zone bits are to be used in an internal comparison, leave column 52 blank.

In Figure 42, L3 in columns 59-60 causes the field located in positions 1-4 in the input record:

1. To be placed into storage in one area with the zone bits removed from all positions of the field (this permits the use of the field for control functions).

2. To be placed in another field named DIVSON in numeric format.

The RPG object program automatically performs the operations of storing the field twice and performing the appropriate zone elimination. No additional specifications are required.

Rules for Using Control Fields

The following points must be considered when using control-field specifications:

1. Within each control level, the length of the control fields or the overall length of the split control fields must be the same.

2. Split control fields: Several fields in an input record type can be specified as one control field to form a split control field.

Split control fields of any one level must be specified in subsequent field-description lines. All fields are placed according to the sequence in which they are defined on the Input Specifications form. For example, the first field defined in the input specifications is placed in the high-order position, and the last field is placed in the low-order position.

3. Control fields of one level can be specified as numeric or alphameric alternately. If a control field is specified once as a numeric field, all control fields of the same level will be treated as numeric control fields throughout, even if they are specified as alphameric in other input specifications.

4. The length of a packed numeric field is calculated as 2n-1, where n is the number of characters contained in the input field concerned. If control fields of one level are specified as packed and unpacked fields alternately, the unpacked fields must always have the same (odd) length as the packed numeric fields.

   If the length of the unpacked field is even, (i.e., the first digit of the packed field is not to be used by the object program), the programmer should specify (as split control fields) the first column of the packed field as unpacked numeric or alphameric and the remainder as packed numeric in two subsequent lines on the specifications form.

   Note: Packed data can occur only in disk records.

5. If multiple record types are specified in an OR-relationship, an indicator entered in Field-Record Relation can be used to relate control fields to the pertinent record types. The following rules apply:

   - The control-field specifications must be grouped by field-record relation indicators.
   - The control fields with no such indicators must be specified first within one record.
   - Fields other than control fields may be interspersed anywhere within the set of field descriptions.
   - Only a record identifying indicator from the main or an OR-relation record type can be used in field record relationship of a control level.

   The control fields that have no field-record relation indicators are assumed to be contained in all records specified in the OR-relation group described above. If control fields of one level are specified with and without field-record relation indicators, only the control fields conditioned by field-record relation indicators are used when the appropriate indicator is turned on. The control-field specification without a field-record relation indicator specified is then disregarded.

6. Control field specifications for a chained file are ignored by RPG.

7. Control levels may be defined in any sequence. That is, L2 may appear before L1 on the Input Specifications sheet.

Example of Using Split Control Fields

Figure 43 shows an example of using three levels of control fields contained in four different records in an OR-relationship.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350 Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 02

Program Identification: REPORT

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01		SPLTTCTL	AA			91	80		C	1																					
02		OR				92	80		C	2																					
03		OR				93	80		C	3																					
04		OR				94	80		C	4																					
05																					1	5		FLDA	L1						
06																					46	50		FLDB	L1						
07																					11	20		FLDC	L2						
08																					76	77		DAY							
09																					51	60		FLDD	L3						
10																					31	40		FLDE	L3						
11																					6	10		FLDA1	L1		91				
12																					46	50		FLDB1	L1		91				
13																					51	70		FLDF	L3		91				
14																					6	10		FLDA1	L1		92				
15																					46	50		FLDB1	L1		92				
16																					74	75		MONTH			92				
17																					61	70		FLDC2	L2		92				
18																					78	79		YEAR							

Figure 43. Example of Multiple Split Control-Field Specifications

record type	L3		L2	L1	
91	51-70		11-20	6-10	46-50
92	51-60	31-40	61-70	1-10	46-50
93	51-60	31-40	11-20	1-5	46-50
94	51-60	31-40	11-20	1-5	46-50

Figure 44. Example, Record Locations

Figure 44 shows the record locations from which the control fields will be taken when the different record types are read during execution of the object program.

Record type 91: The L1 and L3 control fields are specified and conditioned by the field-record relation indicator 91 in lines 02110-02130. The entries with indicators L1 and L3 that are not conditioned by field-record relation indicators are disregarded. Field-record relation indicator 91 is not entered for the L2 control field specified in line 02070. Therefore, this L2 control field is used by the object program for records of type 91.

Record type 92: L1 and L2 indicators are specified in lines 02140-02170. The specifications with the L3-indicators entry in lines 02090-02100 are used by the object program.

Record types 93 and 94: Control-field entries conditioned by field-record relation indicators are not specified for these records. The contents of the fields specified without a field-record relation indicator entry (lines 02050-02100) are obtained and processed by the object program for all three control levels.

False Control Levels

Normally, different record types will contain the same number of control fields. However, some applications require a dif-

ferent number of control fields in some cards. In Figure 44.1, the salesman card contains only the L2 control field while the item card contains both L2 and L1 control fields.

With normal RPG coding, a false control break will be created by the first item card following a salesman card. This will be recognized by an L1 control break immediately following the salesman card (see Figure 44.2). This can be corrected by the coding shown in Figure 44.3. Statement four on the Calculation Specifications sets on indicator 11 when the salesman card is read. When the next item card causes a control break (L1 will be turned on if the L1 hold area does not compare with the new L1 field), no total output will occur because indicator 11 is on (Output Specifications statement nine). Detail calculations are then processed for the item card and statement five sets indicator 11 off. This allows the normal L1 control breaks to occur. The correct output is shown in Figure 44.4.

```
01    SMITH
                    *
          100       3
          100       2
                    2
                    5 *
          101       4
                    4 *
                    9 **

02          JONES
                    *
          100       6
          100       2
                    8 *
          101       3
                    3 *
                   11 **

                   20
```

Figure 44.2. False Control Level Break (Output Example)

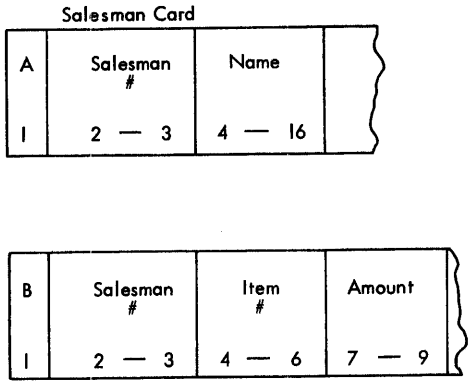


Figure 44.1 False Control Level Break (Salesman Card)

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	Sequence 15 16	Number (1-N) 17	Option (O) 18	Resulting Indicator 19 20	Record Identification Codes 1: Position 21–24	1: Not (N) 25	1: C/Z/D 26	1: Character 27	2: Position 28–31	2: Not (N) 32	2: C/Z/D 33	2: Character 34	3: Position 35–38	3: Not (N) 39	3: C/Z/D 40	3: Character 41	Stacker Select 42	Packed (P) 43	Field Location From 44–47	Field Location To 48–51	Decimal Positions 52	Field Name 53–58	Control Level (L1-L9) 59 60	Matching Fields or Chaining Fields 61 62	Field-Record Relation 63 64	Field Indicators Plus 65 66	Field Indicators Minus 67 68	Field Indicators Zero or Blank 69 70	Sterling Sign Position 71–74
0 1	I	CARD	AA			Ø1		1	C	A																					
0 2	I																				2	3		L2FLD	L2						
0 3	I																				4	16		NAME							
0 4	I		BB			Ø2		1	C	B																					
0 5	I																				2	3		L2FLD	L2						
0 6	I																				4	6		L1FLD	L1						
0 7	I																				7	9	Ø	AMT							
0 8	I																														

Figure 44.3. False Control Break- -Specification Forms (Part 1 of 2)

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

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 (LO-L9, LR)	Indicators: And / And — Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank — Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C											
02	C											
03	C											
04	C		01		SETON						11	
05	C		02								11	
06	C		02	AMT	ADD	L1TOT	L1TOT	50				
07	C	L1		L1TOT	ADD	L2TOT	L2TOT	50				
08	C	L2		L2TOT	ADD	LRTOT	LRTOT	50				
09	C											

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

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	Type (H/D/T)	Stacker Select	Space: Before / After	Skip: Before / After	Output Indicators: And / And — Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	PRINTER	D		1	1	01							
02	O							L2FLD			5			
03	O							NAME			25			
04	O													
05	O		D		1		02							
06	O							L1FLD			15			
07	O							AMT	Z		25			
08	O													
09	O		T		1		L1N11							
10	O							L1TOT	ZB		25			
11	O										27		'*'	
12	O		T		1		L2							
13	O							L2TOT	ZB		25			
14	O										28		'*'	
15	O		T		1		LR							
	O							LRTOT	ZB		25			
	O													

Figure 44.3. False Control Break- -Specification Forms (Part 2 of 2)

```
01     SMITH
       100        3
       100        2
                  5 *
       101        4
                  4 *
                  9 **

02     JONES
       100        6
       100        2
                  8 *
       101        3
                  3 *
                 11 **

                 20
```

Figure 44.4 False Control Level Break (Corrected Output)

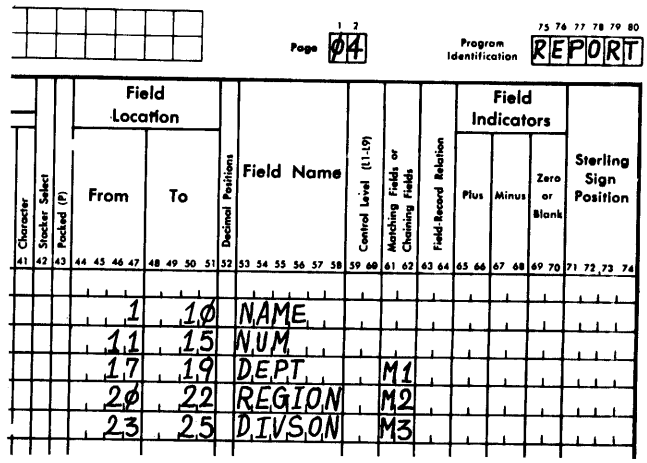


Figure 45. Example of Using Matching Fields in a Single Input File

MATCHING FIELDS OR CHAINING FIELDS (COLUMNS 61-62)

Matching Fields

Matching fields within a record are designated by entering M1 to M9 in columns 61-62 of the appropriate field-description-specifications line. These entries can be used for two purposes:

1. Matching fields used with one input file:

   If only one input file is specified with matching field entries, these entries are used to sequence-check fields within the file. The MR indicator will not be turned on in such a case.

   In Figure 45, three fields within the input file are to be sequence-checked. The compare operation between the matching fields of two subsequent records of the file is made on all three fields at the same time, (i.e., the three fields are treated as one). M3 is placed in the high-order position, M1 is placed in the low-order position.

   In column 18 of the associated File Description Specifications form, the programmer must indicate whether the file is in ascending or descending sequence. The example in Figure 45 is based on the assumption that the file has been specified to be in ascending sequence.

2. Matching fields used with several input files:

   In addition to the sequence checking described above, the matching fields are used to match records of the primary file with those of secondary files. This applies when more than one input file is specified with matching field entries. The entries in columns 61-62 determine which fields are to be matched. The comparison is made in the same manner as shown in Figure 46. For details, refer to the section Processing Multiple Input Files.

Rules for Using Matching Fields

The following rules must be observed when specifying matching fields.

1. There can be nine matching fields specified by indicators M1-M9.

2. The locations of the matching fields within a record type of a file must remain fixed. (They can vary if field record relation is used).

3. When there is more than one record type in an input file, the locations of the matching fields in the various types of records need not be the same.

4. The same number of matching fields must be specified on both the primary file and the secondary file. It is not allowed to specify M1, M2, and M3 in the primary file and only M1 and M2 on the secondary file.

5. Matching field specifications M1 to M9 must not be made for a chained file.

Data Contained in First Record

```
DEPT    008
REGION  051
DIVSON  005
```

Data Contained in Second Record

```
DEPT    003
REGION  025
DIVSON  005
```

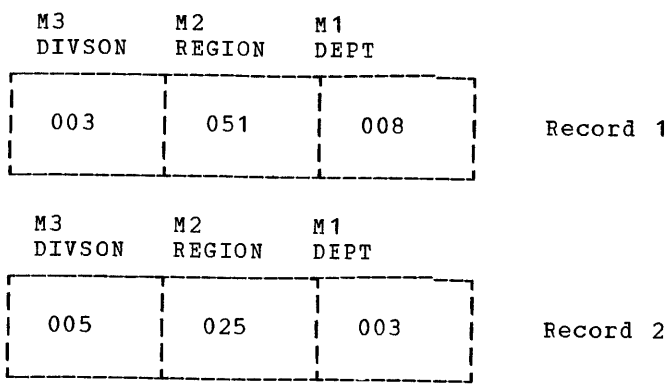


Figure 46. Comparing Matching Fields

They can be used for a file retrieved with an RA file.

6. If matching fields are specified for several records in a program, the same number and level of matching fields must be specified for each of these records. They must also have the same field length for the same level. Thus, it is not permitted to specify a record with only a matching field M1, and another record with only a matching field M2, or both M1 and M2. It is also not permitted to specify the same matching field, for example M1, with different lengths on two field description lines.

7. Split matching fields must not be specified. The object program places the fields defined with M1 to M9 in one holding area and compares them for matching and sequence checking at the same time. Thus, the matching fields are split fields themselves.

8. Matching fields of the same level can be specified as numeric in packed or unpacked decimal format alternately, or as alphameric fields. If a matching field is once specified as numeric, all matching fields of the same level will be treated as numeric matching fields throughout, even if they are specified as alphameric in other specifications (i.e., all zones are ignored by the internal compare operation for matching and/or sequence checking).

9. The length of a packed matching field is calculated as 2n-1, where n is the number of characters contained in the input field concerned. The calculated length is always an odd number. Packed format can occur only with disk records.

   If matching fields of one level are specified as unpacked and packed alternately, the lengths of the unpacked matching fields must be equal to the calculated lengths of the packed numeric matching fields.

   When matching fields have an even length in unpacked decimal format, the only way to specify these fields in packed and unpacked decimal format alternately is to define the first (high-order) column of both fields as unpacked numeric fields M2, and the remainder of the fields (i.e., the odd-length portion) alternately as the lower-level matching field M1.

10. Field-Record relations for matching fields must be specified with record identifying indicators from the main specification line or an OR-relation line to which the matching field refers. If multiple record types are specified in an OR-relationship, an indicator entered in Field-Record Relation can be used to relate matching fields to the pertinent record types. The following rules apply:

    - The matching field specifications must be grouped by field-record relation indicators.
    - Matching fields with no such indicators must be specified first within one record.
    - Fields other than matching fields may be interspersed anywhere within the set of field description.

    The matching fields that have no field-record relation indicators are assumed to be contained in all records specified in the OR-relation group described above. If matching fields of one level are specified with and without field-record relation indicators, only the matching fields conditioned by field-record relation indicators are used when the appropriate indicator is turned on. The matching field specification without a field-record relation indicator specified is then disregarded.

In the following examples the first entry is the match field and the second entry is the field-record relationship indicator. Assume that indicators 01 and 02 are record identifying indicators in an OR relationship.

VALID EXAMPLES

1.	2.	3.	4.	5.	6.
M1 01	M1 01	M1	M1	M1	M1
	M1 02	M1 01	M2	M2	M2
			M1 01	M1 01	M3
				M1 02	M1 01
					M2 01
					M1 02

INVALID EXAMPLES

A.	B.	C.	D.	E.	F.
M1 01	M1	M1	M1 01	M1	M1
M1	M2 01	M1 01	M2 01	M1 L1	M2
	M2 02	M2 01	M1 02		M1 01
			M2 02		M1 02
			M3		M2 01
					M2 02

Chaining Fields (Disk Only)

Two methods of chaining are allowed with 1130 RPG. The preferred method uses the CHAIN operation discussed in the section Calculation Specification. The following discussion concerns the alternative method of chaining and the required entries on the Input Specification sheet.

Up to three chaining fields are permitted in a record. To specify chaining, enter the code that identifies the chaining field C1 to C3 in columns 61-62.

A chaining field can be specified as an alphameric, or an unpacked numeric field. The format of the key field on disk can only be alphameric, therefore:

- Alphameric fields are moved unchanged into the appropriate chaining fields (C1 to C3).
- Unpacked numeric fields will be moved in unpacked format, but with 'F' zones, into the appropriate chaining fields.
- If a chaining field is specified alternately as numeric and alphameric it will always be handled as a numeric field. In other words the zone parts of the alphameric field will always be set to 'F'. Note that all positions of the key field on disk must also have 'F' zones if a chaining field is specified as numeric.

The use of file chaining is explained in the section Chaining. A chaining file can also be sequence-checked by entering M1-M9 in Chaining Fields (columns 61-62) on the line containing the specification for the appropriate field.

If the same field is used for chaining and sequence-checking it must be defined twice using two different field names.

Note: Matching field and control field specifications are not permitted on the chained file input records.

FIELD RECORD RELATION (COLUMNS 63-64)

This specification is used in conjunction with specifications for records in an OR-relationship.

Enter in these columns the record identifying indicator that determines which input record in an OR- relationship supplies the input field.

In addition, any resulting indicators can be used in conjunction with any field which is not a control level or matching field.

Control levels used in connection with field-record relation.. For details refer to the section Rules for Using Control Fields, item 5.

Using matching fields together with chaining field entries. See the section headed Rules for Using Control Fields, item 10.

Using Field-Record Relation together with Chaining field entries. An additional function of this specification is to selectively control chaining operations. This, however, is an alternative method and most selective chaining can best be accomplished with the chain operation on the calculation specifications. (See Chaining for a general explanation of chaining. In order to understand this function, readers should be familiar with chaining operations.)

If a record identifying indicator is placed in Field-Record Relation, the chained record will be obtained only if any one of the record types specified in an OR-relationship is present, and the record identifying indicator (in Field-Record Relation) is on.

However, if a chaining field is specified on a field description line and no entry is made in Field-Record Relation, the chained record will be obtained if any one of the record types specified in an OR-relationship is present. To condition chaining, any indicator may be used.

FIELD INDICATORS (COLUMNS 65-70)

An entry in Field Indicators causes a test of the associated field to determine if it contains a positive value (plus), or a negative value (minus), or no value (zero or blank). Each input field may be assigned one, two, or three field indicators in Plus, Minus, and/or Zero or Blank. Any indicator thus assigned can be used to condition calculation and/or output-format specification. The functions of field indicators and the available indicator codes are described as follows.

Plus (Columns 65-66)

Enter the resulting indicator that is to be turned on whenever the value in the associated input field is greater than zero. This entry is used with numeric fields only.

Minus (Columns 67-68)

Enter the resulting indicator that is to be turned on whenever the value of the associated input field is smaller than zero. This entry is used with numeric fields only.

Zero or Blank (Columns 69-70)

Enter the resulting indicator that is to be turned on whenever the associated input field contains no value. This condition occurs, if (a) a numeric input field is zero or blank, and (b) an alphameric field is blank. Fields that are all zeros turn on blank indicators even though a plus or minus sign is present.

Note: Alphameric input fields must not be assigned field indicators in Plus or Minus (i.e., Columns 65-68 must be blank). If the input field does not contain the appropriate value, the indicator is turned off.

Codes Available for Use as Field Indicators

Numeric Indicators. A 2-digit field indicator code is used for this specification. These codes, ranging from 01 through 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 resulting from the previous specification for it.

Note: Defining these indicators means specifying them in Field Indicators. This should not be confused with using these indicators. Using these indicators means specifying them in Indicators in the calculation specifications or in Output Indicators in the output-format specifications as many times as required. In the latter case, they are merely tested to determine their status but they are not reset by the test.

Halt Indicators. Nine additional indicator codes, referred to as halt indicators, can be used in the RPG program. For example, if H1 is placed in columns 69-70 (Zero or Blank), it is turned on whenever the associated field of the current input record is blank or contains zero(s) only. These indicators can also be used to control processing in calculation and output specifications.

If H1-H9 has been turned on during the processing of a record, the execution of the object program is discontinued on 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 by another specification before the program attempts to process the next input record.

How Field Indicators Are Turned On and Off

Field indicators entered in Plus, Minus, Zero, or Blank are turned on if the value in the tested input field is positive or negative, respectively. They are turned off if the associated input field of the next record of the same type contains a negative or positive value, respectively.

Each field indicator is related to one record type. Therefore, a field indicator cannot be turned off until the associated record type is read again or until it is defined in some other specification. One or more field indicators can be on at one time.

Zero or Blank indicators are initially turned off by the program and are not turned on if the associated field is "Blanked after" on the Output Specifications.

Figure 46.1, illustrates the setting of field indicators. Indicators 21-25 will be set on/off each time a record with a 1 in position one is read from the file DISKIN. If the next record read from the same file has a 2 or 3 in position one, the status of indicators 21-25 will remain unchanged. They will not be changed until another record with a 1 in position one is read.

Only numeric fields can be tested for a plus or minus setting. The same indicator can be used to test for either of two conditions (line 05). Different indicators can be used to test all of the possible conditions (line 08). Lines 10 and 11 show an improper coding technique. When a record with 3 in position one is read, indicator 29 will be set to reflect the status

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page

Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	DISKIN	AA			Ø1	1		C	1																					
02	I																				1		50	FLDA				21			
03	I																				6		80	FLDB					22		
04	I																				9	11		FLDC				24		23	
05	I																				12	14		FLDD						24	
06	I																				15	20		FLDE						25	
07	I		BB			Ø2	1		C	2																					
08	I																				1		50	FLDF				26	27	28	
09	I		CC			Ø3	1		C	3																					
10	I																				1		5	FLDG						29	
11	I																				6		9	FLDH						29	
12	I																														

Figure 46.1. Setting of Field Indicator

of FLDG and then reset to reflect the status of FLDH. Consequently, the status of FLDG is lost and indicator 29 will only reflect the status of FLDH.

STERLING SIGN POSITION (COLUMNS 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. Leave these columns blank if the field is not a sterling field.

FIELDS OF THE CALCULATION SPECIFICATIONS FORM

The Calculation Specifications form is used to specify the operations to be performed by the object program upon the input data and upon data obtained as a result of other calculations. Four general rules govern the writing of calculation specifications:

1. Each operation is specified on one line of the form.
2. Detail calculations must precede all total calculations.
3. Calculations are specified in the order they are to be performed.
4. Subroutines must follow all other calculations. Subroutines need not be specified in the order in which they are used.

The Calculation Specifications form is divided into three categories as illustrated in Figure 47. The circled numbers in Figure 47 refer to the item numbers in the following text.

1. Conditioning fields. Entries in columns 7-17 determine upon what conditions the calculations are to be performed.
2. Calculating fields. Entries in columns 18-53 define the kind of calculation to be performed, the data to be used in the calculation, and the result field of the calculation.
3. Testing fields. Entries in columns 54-59 cause the testing of result fields and the setting of indicators that can be used to condition subsequent calculation or output-format specifications.

CONDITIONING FIELDS (COLUMNS 7-17)

Control Level (Columns 7-8)

An entry, (L1-L9, L0, LR) in columns 7-8 indicates that calculations are to be performed at total time and determines the control level of the calculation specified in columns 18-59.

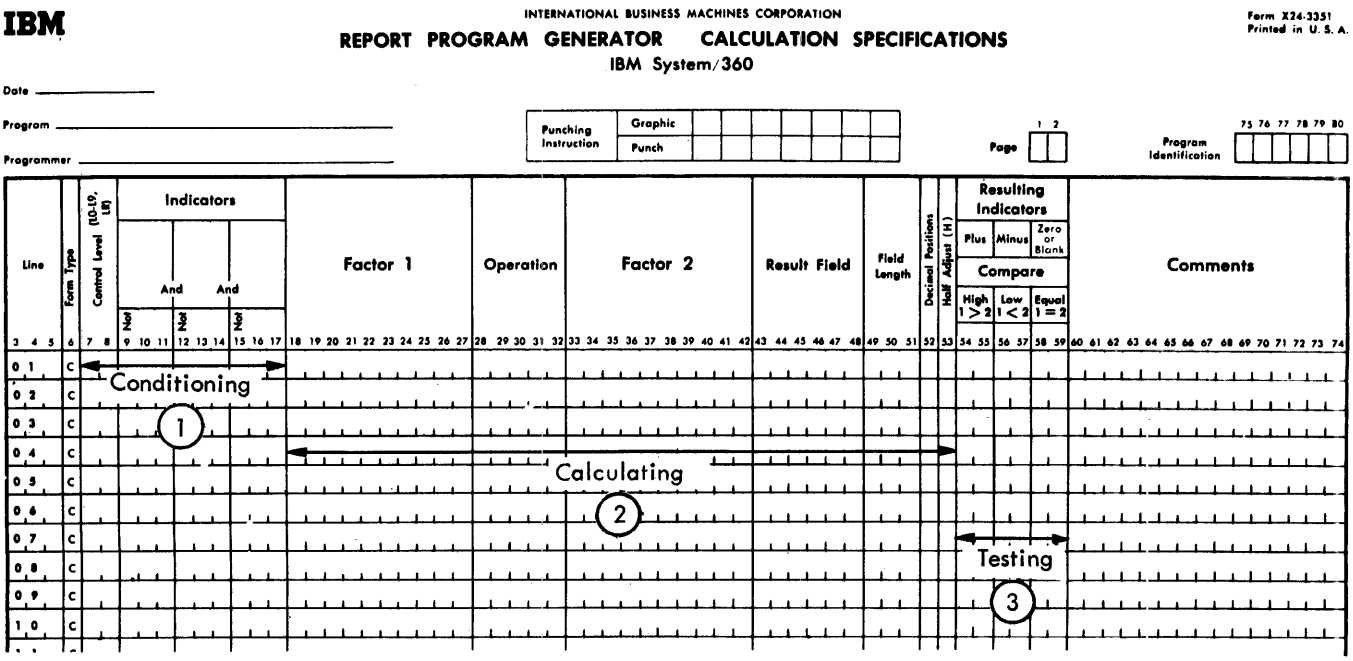


Figure 47. The RPG Calculation Specifications Form

The entries for these columns are the control-level indicators L1 through L9, as defined in Control Level (columns 59-60) of the Input Specifications form. In addition, the two control-level indicators L0 and LR, which are discussed below, may be specified. A control-level indicator is turned on by a control break. It is on during total time and it remains on for the following detail time, which includes both the calculating and the printing of the detail record.

A control break for a specific control level forces all lower control-level indicators to be turned on. For example, an L3 break causes L3, L2, and L1 indicators to be turned on.

When an input record has been read, a test is made to determine if a control break has occurred. Total calculations are performed when this test is completed and before the record that caused the control break is processed.

Total calculations and total output lines are processed at the beginning of a program run as follows:

1. If no control fields are specified, total calculations and total output lines are bypassed for the first record read.

2. If control fields are specified, total calculations and total output lines are bypassed for all records read until the first record that contains control field information has been processed. This applies also to the calculations specified with an L0 indicator.

3. The control fields are initialized by the program to hexadecimal zero (see the section Data Format). Therefore, if the first record has a blank or zero control field it will cause a control break.

4. If control fields are specified in the program in some records and not in others, at least one record with control fields should be processed. If not the LR indicator will not be turned on at the end of the job.

LR (LAST RECORD) INDICATOR. The indicator LR is turned on after the last input record has been read and calculated, and after the specified detail output records have been processed. At this time, the control-level indicators L1 through L9 are also turned on and final totals can be performed.

L0 (LEVEL ZERO) INDICATOR. The L0 indicator is turned on at the beginning of a job and before a record is read. It is not turned off by the RPG object program at any time. The L0 indicator cannot be turned off by the programmer. By specifying L0, total calculations can be performed even though a control break has not occurred. This operation is useful whenever totals are to be accumulated when no control break has occurred.

USING CONTROL LEVEL SPECIFICATIONS. Figure 48 illustrates six control-break specifications.

Line	Form Type	Control Level (L0-L9, LR)	Indicators			Factor 1	Operation	Facto
			And	And				
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
0 1 Ø	C	LØ	98			FIELDA	ADD	
0 2 Ø	C	L1				DEPTOT	SUB	
0 3 Ø	C	L1				FIELDB	ADD	
0 4 Ø	C	L2				FIELDC	ADD	
0 5 Ø	C	L2				FIELDN	ADD	
0 6 Ø	C	L2				FIELDR	MULT	
0 7	C							

Figure 48. Example of Using Control Level Indicators in Calculation Specifications

SR (SUBROUTINE IDENTIFICATION): The entry SR in these columns 7-8 identifies this line as a specification with a subroutine (see Subroutines). All RPG subroutines must have SR in columns 7-8. Control levels are not used in subroutines, as the subroutine may be invoked by a specification which is at either detail or total calculations.

Indicators (Columns 9-17)

The entries in columns 9-17 define the conditions (if any) that control the calculations specified in columns 18-59. From one to three indicator codes can be specified. If the calculation specified in columns 18-59 of a line is not governed by an indicator condition, leave columns 9-17 blank. If two or three indicators are specified in columns 9-17 on one line, these indicators are assumed to be in an AND-relationship. That is, if three indicator codes are specified, all three indicator conditions must be satisfied before the calculation can be executed. Enter in colums 10-11, 13-14, and 16-17 the appropriate indicators that must be checked before the associated calculation is performed. If it is required that an operation should only be executed if an indicator is off, enter an N in ei-

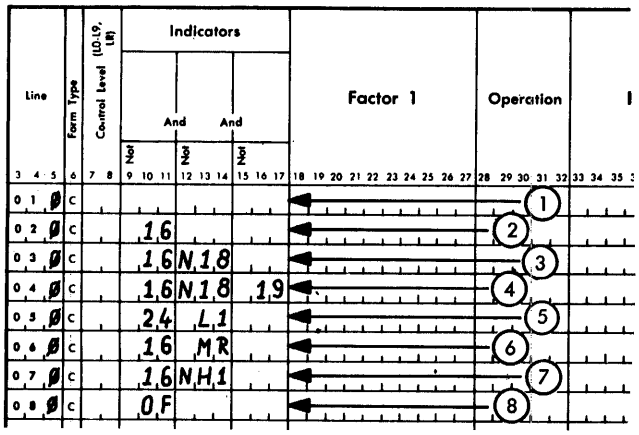


Figure 49. Example of Indicators in Calculation Specifications

ther column 9, 12, or 15 (whichever is appropriate).

The following considerations apply to the use of indicators in columns 9-17 of the Calculation Specifications form:

1. If columns 9-17 and columns 7-8 are blank, the calculation is performed each time a detail record is read.

2. If a record-identifying indicator that has been defined in columns 19-20 of the Input Specifications form is placed in Indicators on a particular calculation line, this calculation is performed only if a record of the type associated with the record identifying indicator is read.

3. A field-indicator code controls the calculation according to the status of the associated input data field.

4. A resulting indicator specified in columns 54-59 of a preceding calculation line controls the current calculation according to the condition that occurred in the result field of the preceding calculation. (This feature has not yet been discussed.)

5. A control-level indicator (L1-L9) that is used with a particular record identifying indicator defined on the Input Specifications form permits the calculation to be performed at detail time but only on the first record of the control group specified.

6. If the matching-record indicator (MR) is placed in Indicators, the calculation is performed only if there is a matching record in a secondary input file.

7. Halt indicators H1 through H9 are normally used to suppress a calculation when an error has been detected in the input data or during a previous calculation.

8. Overflow indicators OF and OV permit the calculation to be performed only if a page overflow has occurred.

9. If columns 9-17 of the specification line contain entries which are exactly identical to the preceding line, no repetitive test will be made. (The core storage requirements for testing indicators are also dropped.) Consequently, the programmer should group his calculations by identical conditioning indicators to reduce core and improve throughput.

Columns 9-17 may contain a combination of the types of indicators discussed in the preceding text.

Figure 49 shows some examples of indicators used in columns 9-17 of the calculation lines. The numbers to the right of the entries refer to the item numbers in the text below.

1. In this example, columns 7-17 are blank. This specifies that the calculation is to be performed on each detail record read.

2. In this example, the associated calculation is performed only if indicator 16 is on.

3. Assume that indicator 16 is a record-identifying indicator defined in columns 19-20 of the input specifications, and indicator 18 is a field indicator defined in Zero or Blank on the Input Specifications form. Then the associated calculation is performed only if record-type 16 is present and the field assigned to indicator 18 is not blank.

4. The calculation specified on this line (columns 18-59) is performed only if indicators 16 and 19 are on and indicator 18 is off.

5. This entry specifies that the calculation be performed at detail time whenever control-level indicator L1 and indicator 24 are on for the first card of a new group.

6. This entry specifies that the calculation be performed only if indicator 16 is on and there is a matching record condition. (For example, when fields from detail records are to be multiplied by a factor contained in a master

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And / Not 9 10 11 12 13 14 15 16 17	Factor 1 18-27	Operation 28-32	Factor 2 33-42	Result Field 43-48	Field Length 49-51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2 54 55 56 57 58 59	Comments 60-74
01	C		11 11 12 13 14 15 16 17		SETON						21	
02	C		11 14 21	A	ADD	B	C	40				
03	C				SETOF						21	
04	C											

Figure 49.1 Calculations Specifying an AND Condition

record, the program must have a means of ensuring that the detail record has been matched with the appropriate master record.)

7. The entry NH1 in this example prevents the object program from performing the associated calculation if an error condition has occurred that resulted in H1 being turned on. This application of halt indicators prevents the calculation of erroneous data.

8. This entry specifies that the calculation be performed during the detail cycle following page overflow and printing of the overflow lines.

Calculations in an AND/OR Relationship

There is no AND/OR ability on the Calculation Specifications as there is on Input and Output Specifications. The programmer must specify separate entries to code an OR condition. An AND condition can be coded by setting an intermediate indicator. Figure 49.1 shows an example of how this can be done when four indicators (11,12,13,14) must all be on before an addition is made. It is important to note that indicator 21 must be turned off each time or the addition will not be conditioned correctly.

Figure 49.1

CALCULATING FIELDS

This section describes the actual calculation specifications. These specifications (columns 18-53) are used to define the following:

1. The data fields to be used (Factors 1 and 2).

2. The type of operation to be performed (operation).

3. The result field (columns 43-53).

General Description of Data Fields

1. Field Name is to be used in calculations and must be defined in either columns 53-58 (field name) of the Input Specifications form, or columns 43-48 (result field) of the Calculations Specifications form. The first character of a field name must be alphabetic. The length must not exceed 6 characters. Special characters and embedded blanks must not be used.

2. Literals are the actual data to be operated on, rather than a name representing the location of data in storage. Literals may be numeric or alphameric and must be left-justified.

3. Label is the name of a subroutine begin or end statement, or a tag statement. The first character must be alphabetic.

The length must not exceed 6 characters. Special characters and embedded blanks must not be used. A label cannot have the same name as a field.

Numeric Literals.

A numeric literal may consist of any combination of the digits 0 through 9. One decimal symbol and/or one plus sign or one minus sign may also be used. Other separators (e.g., a comma between the thousands and hundreds positions) must not be used. Numeric literals may be up to 10 characters long. (The European format of commas to denote the decimal point may be used if the inverted print option is selected in the RPG control card)

Rules for Forming Numeric Literals

1. Blanks must not appear within a numeric literal.

2. The sign, if present, must be the leftmost character. If a literal is unsigned, it is treated as positive.

3. The decimal point may appear anywhere in the literal.

4. Numeric literals must not be enclosed in apostrophes.

Alphameric Literals.

Any set of characters enclosed in apostrophes is treated as an alphameric literal. Alphameric literals may be used for compare, move, and table-lookup operations, but they must not be used in arithmetic operations.

Rules for Forming Alphameric Literals

1. Alphameric literals must be enclosed in apostrophes.

2. Any characters of the EBCDIC character set may be used in an alphameric literal. Blanks in the body of the literal are treated as valid characters. The maximum length of alphameric literals is 8 characters, excluding the two enclosing apostrophes.

3. An apostrophe may be included in a literal by entering two consecutive apostrophes. For example, the literal o'clock would be coded as 'o''clock'.

Factor 1 (Columns 18-27)

The specifications in Factor 1 may be a field name or a literal. Both field names and literals must be left-justified. They may be defined as alphameric or numeric, depending on the calculation to be performed.

Figure 50 illustrates samples of Factor 1 entries.

The numbers in the right-hand margin of Figure 50 refer to the item numbers in the following text.

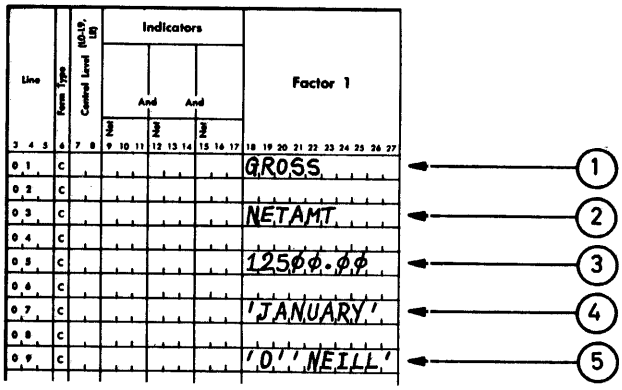


Figure 50. Sample Entries under Factor 1 on the Calculation Specification Form

1. GROSS is a field name that may have been defined, for instance, on the Input Specifications form.

2. NETAMT may be the name of a result field of another calculation, or of an input field defined on the Input Specifications form.

3. The numeric literal in the example could be used to determine if the contents of a specific input field are higher or lower than this quantity. The position of the decimal point must be indicated if the number is not an integer. No other function is permitted.

4. An alphameric literal, like the one in this example, can be compared against the contents of a data field, or moved into a data field, etc.

5. This example shows how to specify an apostrophe within an alphameric literal. The literal shown in this example would be printed as O'NEILL.

Factor 2 (Columns 33-42)

The description of Factor 1 also applies to Factor 2.

Operation (Columns 28-32)

Entries in these columns specify the operation to be performed using the entries in Factor 1, Factor 2, and Result Field. Each operation is specified by placing the operation code left-justified in Operation (columns 28-32). For detailed information refer to the section Entries in the Operation Field.

Result Field (Columns 43-48)

Entering a field name in Result Field causes the specified number of core storage positions to be reserved for calculation results. The name in Result Field must be alphameric and left-justified. It must not be more than six characters in length; it must not contain blanks, or special characters, and the first character must be alphabetic.

The same name can be used several times in different calculations if the length of the field and the number of decimal positions are the same for all calculations.

Figure 51 illustrates sample result-field entries. On the first line, GROSS is multiplied by DRATE, and the result is placed in DISCNT. The contents of DISCNT are then used as Factor 2 on the next line to calculate the net amount which is placed in NETAMT. The same field DISCNT is then used in Factor 1 on the next line to calculate total discount, which is placed in TOTDIS.

Field Length (Columns 49-51)

This entry specifies the length of the result field, (i.e., the number of positions to be reserved for the field specified in Result Field). The unpacked length must be specified. The maximum length of numeric fields is 14 digits. The maximum length of alphameric fields is 256 characters. In Figure 51 the field DISCNT is specified to have a total length of 8 digits.

Note: If the length of an arithmetic result exceeds the specified length of the associated result field, the leftmost digits are truncated. Any resulting indicators will be set according to the truncated field.

If the same result field name is used in more than one calculation, the field-length specification and the decimal position specification need not be specified more than once. If the length of and the number of decimal positions in a result field has been established on the Input Specifications form, entries in Field Length and Decimal Position on the Calculation Specifications form are not required. However, if such entries are made, they must not contradict the related entries on the Input Specifications form.

Note: If half-adjustment (rounding) is specified, the entries in Field Length and Decimal Position refer to the length of the result field after half-adjustment.

IBM

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 05

Program Identification

Line	Form Type	Control Level (L0-L9 LR)	Indicators: And / And (Not)	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C			GROSS	MULT	DRATE	DISCNT	8	2	H		
02	C			GROSS	SUB	DISCNT	NETAMT	8	2			
03	C			DISCNT	ADD	TOTDIS	TOTDIS	8	2			
0	C											

Figure 51. Sample Result Field Entries

Decimal Position (Column 52)

An entry (0-9) in Decimal Position identifies the associated result field as numeric and specifies the number of positions to the right of the decimal point in the result field. An entry must be made in this column for all arithmetic operations if the field is not specified elsewhere. If no decimal positions are to be retained in the result field, enter 0. Otherwise, enter one of the digits 1-9, as required. In Figure 51 each result field is specified to have two decimal positions.

Note: The number of decimal positions must not exceed the specified length of the result field.

If the result field is alphameric, leave this column blank.

Half-Adjust (Column 53)

An H in Half-Adjust specifies half-adjusting (or rounding) of the result.

If the number of decimal positions of the arithmetic result is less than or equal to the specified decimal positions of the result field, no half-adjusting is possible. If the arithmetic result has more decimal positions than the specified result field, an H in Half-Adjust causes the arithmetic result to be truncated (by dropping the rightmost digits) until the result is one digit longer than the specified result field. The absolute value of the low order digit of the (truncated) result is added to the absolute value of the (truncated) result. The rightmost digit is then dropped, the original sign (plus or minus) is restored, and the adjusted result is moved into the specified result field. Because of this, rounding always occurs away from zero for both positive and negative fields.

TESTING FIELDS

The last portion of the Calculation Specifications form is used to test the result of a calculation, to set indicators, etc.

Resulting Indicators (Columns 54-59)

This specification may be used to test the value of a result field after the completion of an operation. The indicators entered in columns 54-59 are turned on or off depending on the result of this test. They can be used to control subsequent calculation operations or to control output operations. The specification is used in six 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 that determines:

   a. If the table argument next-higher than the search argument is found.

   b. If the table argument next-lower than the search argument is found.

   c. If the table argument equal to the search argument is found.

   To place the indicator in the correct column, the programmer should think in terms of the table (Factor 2) rather than the search argument (Factor 1).

   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. (See the definition under TESTZ).

5. To define SETOF and SETON operations as to what indicators are to be turned off or on.

6. As an option to determine the results of a CHAIN operation. (See the definition under CHAIN)

The entries for this specification can be any of the indicator codes. 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 from a previous specification.

		Columns 54-55		Columns 56-57		Columns 58-59	
		PLUS	HIGH	MINUS	LOW	ZERO OR BLANK	EQUAL
Arithmetic Operation ------>	If the Result Field contains a:	Positive Value	-	Negative Value	-	Zero	-
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

Figure 52. Summary of Conditions that Turn on Indicators in Columns 54-59

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And Not 9 10 11	And Not 12 13 14	And Not 15 16 17	Factor 1 18 19 20 21 22 23 24 25 26 27	Operation 28 29 30 31 32	Factor 2 33 34 35 36 37 38 39 40 41 42	Result Field 43 44 45 46 47 48	Field Length 49 50 51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus 54 55	Minus 56 57	Zero or Blank 58 59	Comments 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
													Compare: High 1>2	Low 1<2	Equal 1=2	
0 1 Ø	C					GROSS	SUB	DISCNT	NETAMT	8	2	H		1Ø	15	
0 2 Ø	C					'JANUARY'	COMP	DATE							24	
0 3	C															

Figure 53. Example of Using Resulting Indicators

Note: Defining these indicators means specifying them as record identifying indicators or resulting field indicators in the input specifications, or as resulting indicators in the calculatien 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-format 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.

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

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; nor is the Zero or Blank indicator turned on by a Blank After specification. Figure 52 illustrates the various conditions that cause resulting indicators to be turned on.

On the first line in Figure 53 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 53 the literal JANUARY is compared against the contents of DATE. If the result is equal, indicator 24 is turned on.

Comments (Columns 60-74)

These columns may be used for comments and are not used by the program.

Type	Operation	Code
Arithmetic Operations	Add	ADD
	Zero and Add	Z-ADD
	Subtract	SUB
	Zero and Subtract	Z-SUB
	Multiply	MULT
	Divide	DIV
	Move Remainder	MVR
Move Operations	Move	MOVE
	Move-Leftmost-Characters	MOVEL
	Move Low-to-High Zone	MLHZO
	Move High-to-Low Zone	MHLZO
	Move High-to-High Zone	MHHZO
	Move Low-to-Low Zone	MLLZO
Testing or Compare Operations	Compare	COMP
	Test Zone	TESTZ
Turning Indicators ON or OFF	Set Indicators On	SETON
	Set Indicators Off	SETOF
Table Operations	Table Lcokup	LOKUP
Branching and Exit Operations	Branching (or Go To)	GOTO
	Providing a Label for GOTO	TAG
	Exit to Assembler Subroutine	EXIT
	RPG Label	RLABL
	Beginning of RPG Subroutine	BEGSR
	End of RPG Subroutine	ENDSR
	Branch to RPG Subroutine	EXSR
Random Retrieval from ISAM or Sequential Disk File	Chaining	CHAIN
Branch to Exception Output	Output Records During Calculations	EXCPT

Figure 54. Entries in the Operation Field

ENTRIES IN THE OPERATION FIELD

Figure 54 sumarizes the RPG operations.

ARITHMETIC OPERATIONS

The fields or literals involved in these operations must be numeric. All arithmetic operations are performed with automatic decimal alignment.

RPG cannot detect an arithmetic overflow so the user must ensure that the length of any field involved in arithmetic operations does not exceed 14 digits. This includes decimal alignment when necessary. The resulting field length after decimal alignment and/or half-adjusting must not exceed 14 digits. Resulting indicators (columns 54-59) may be used with all arithmetic operations.

Two of the fields specified for an arithmetic operation may be specified as the same field. For example, the literal 1 might be added to the field COUNT, and the resulting sum could replace the value previously in COUNT. In this case, COUNT is specified in both Factor 1 and Result.

Add (ADD)

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

Zero and Add (Z-ADD)

This operation causes the result field to be set to zeros and then causes the numeric

literal or the contents of the numeric field in Factor 2 to be placed in the result field. Factor 1 is not used in this operation. This operation is normally used to change the sign of a numeric field.

Subtract (SUB)

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

Zero and Subtract (Z-SUB)

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

Multiply (MULT)

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

Only the number of decimal positions specified in column 52 is retained. Any excess low-order (rightmost) decimal positions are dropped. If the arithmetic result is then longer than 14 digits, all high-order (leftmost) positions of the result in excess of 14 digits are truncated (dropped). The remaining digits of the arithmetic result are placed in the specified result field.

Divide (DIV)

This operation causes the literal or the contents of the field in Factor 1 to be divided by the literal or the contents of the field in Factor 2. The result of this operation (quotient) is placed in the specified result field. The literal or the contents of the field in Factor 2 must not be zero. Any remainder that results from this operation is lost, unless the move-remainder operation is specified as the next operation. If the move-remainder operation is used, half-adjusting (rounding) must not be specified for the divide operation.

The following formulas can be used to determine the highest number of positions (or decimal positions) available in any one of the fields. They can also be used to check in advance whether the pertinent field lengths are such that a given division can be executed.

$$L_2 + D_1 - D_2 - D_r \leq 14; \text{ and}$$

$$L_1 - D_1 + D_2 + D_r \leq 14; \text{ and}$$

in the case of half-adjustment,

$$L_1 - D_1 + D_2 + D_r \leq 13,$$

where

L_1 = Length of Factor 1 (dividend).

D_1 = Number of decimal positions in Factor 1.

L_2 = Length of Factor 2 (divisor).

D_2 = Number of decimal positions in Factor 2.

D_r = Specified number of decimal positions in the result field (quotient).

The number of decimal positions in each of the three elements of a division should satisfy the following condition:

$$A = D_r - D_1 + D_2 = 0$$

If this is not the case, either the dividend or the divisor will be adjusted (padded) by adding zeros to the right, depending on the following conditions:

$A > 0$ implies padding of dividend
$A < 0$ implies padding of divisor

The number of zeros added is equal to the absolute value of A.

Note: While a dividend may have the value of 0, a divisor of 0 will cause an error wait. The operator may terminate the job or continue with a 0 amount in the result field. If the programmer does not wish the operator to have this decision, then the divisor should be tested for 0 and appropriate steps taken. A dividend of zero will always result in a zero result field.

Move Remainder (MVR)

This operation is used to move the remainder of a divide operation to a separate field that has been set to zero by RPG. If MVR is used, it must immediately follow the divide operation. Figure 55 shows an example of using the MVR operation.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction	Graphic
	Punch

Page 05

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And / And / Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank	Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C		12	FIELD1	DIV	FIELD2	SAVE	8	0				
02	C		12		MVR		STORE	4	0				
03	C												

Figure 55. Example of Using the MVR Operation

The remainder is placed in a field named STORE. The field that contains the remainder must be specified in the result field. The value of the remainder can be determined by the following formula:

R = Dividend - Divisor x Quotient

The result field of an MVR operation may be assigned resulting indicators in columns 54-59. Factor 1 and Factor 2 must not be used in an MVR specification. Decimal alignment is performed by the RPG program.

If conditions (control level or indicators) are specified for a DIV operation, the same conditions must be established for the associated MVR operation. The MVR line must contain all the indicators which are specified in the DIV line. However, extra indicators are permitted in the MVR line. If a control level is specified for the DIV operation, the same or a higher control level must be entered in columns 7-8 of the MVR line. If the DIV operation is executed at detail time (columns 7-8 blank), the MVR operation must also be performed at detail time.

MOVE OPERATIONS

The MOVE and MOVEL operations instruct the RPG program to move data. The mnemonic MOVE or MOVEL must be placed in Operation, columns 28-32. The literal or the name of the field that is to be moved is placed in Factor 2. Columns 43-48 of the result field must contain the name of the field into which data is to be moved. If this result field is not defined elsewhere in the program, its length must be specified in columns 49-51. An entry in column 52 is required if this result field is numeric.

Factor 1, Half-Adjust and Resulting Indicators may not be used in a MOVE or MOVEL specification.

By means of 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, place the name of the numeric field in Factor 2 and use an alphameric result field. To change an alphameric field to a numeric field, place the name of the alphameric field in Factor 2 and use a numeric result field. No decimal alignment is performed when a move operation is used.

No sign checking will be performed on moves to a numeric field.

The MLHZO, MHLZO, MHHZO, and MLLZO move operations instruct the RPG program to move zones.

Move (MOVE)

The MOVE operation causes the literal or the contents of the field in Factor 2 to be placed in the specified result field. The field or literal in Factor 2 may be alphameric or numeric. Factor 1 is not used in this operation.

If Factor 2 is shorter than the specified result field, Factor 2 is moved to the rightmost positions of the result field as illustrated in Figure 56. The excess left-hand positions of the result field are not disturbed.

If Factor 2 is longer than the result field, the excess leftmost positions of Factor 2 are not moved, as illustrated in Figure 56.

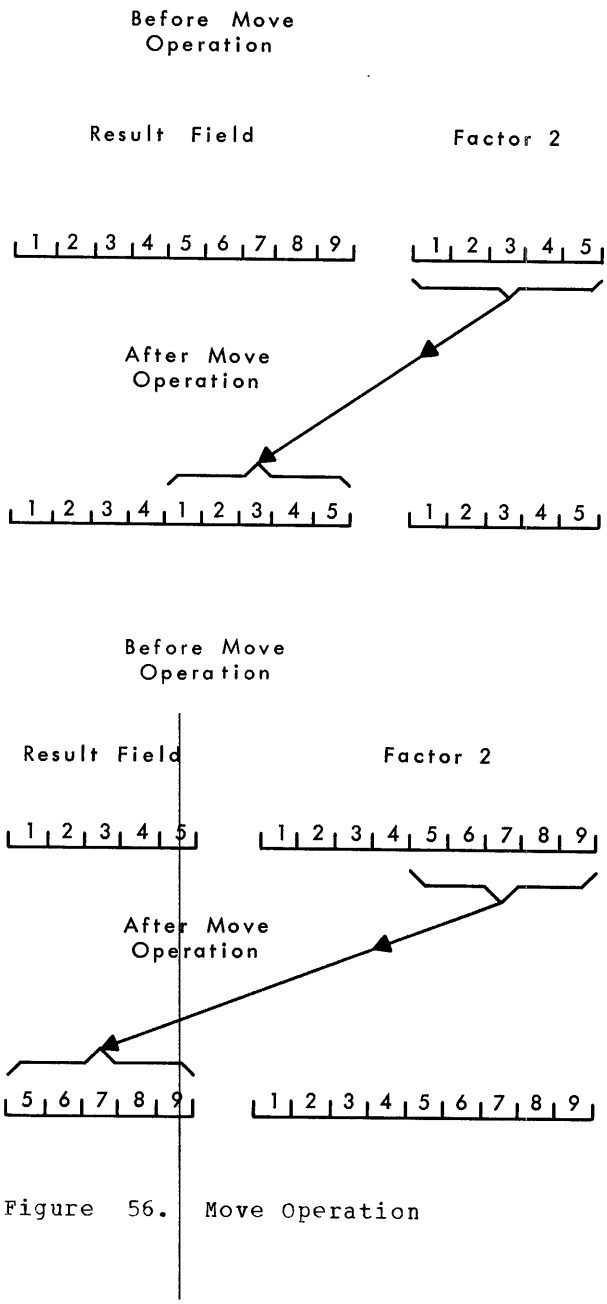


Figure 56. Move Operation

The MOVE operation can be performed on both numeric and alphameric fields. In addition, an alphameric field can be moved into a numeric field or vice versa. Resulting indicators must not be used in MOVE specifications. Decimal alignment or sign checking is not performed for this operation.

Move Left (MOVEL)

The MOVEL operation causes the literal or the contents of the field specified in Factor 2 -- beginning with the leftmost character -- to be moved left-justified into the specified result field. Alphameric data can be moved into numeric fields, and vice versa. Decimal alignment is not performed for this operation.

The format of the MOVEL operation is shown in Figure 57.

The data defined in Factor 2 is moved into the specified result field left-justified. If the result field is longer than Factor 2, the excess right-hand positions of the result field remain undisturbed. If the result field is shorter than Factor 2, the excess right-hand positions of Factor 2 are not moved.

If Factor 2 is shorter than the (numeric) result field, the sign of Factor 2 is not moved. If Factor 2 is equal to or longer than the (numeric) result field, the sign of Factor 2 is moved into the rightmost position of the result field.

If a numeric field is moved into an alphameric result field that is equal to or longer than Factor 2, the sign of the numeric field is moved into that position of the result field that contains the rightmost position of Factor 2. If the alphameric result field is shorter than the numeric Factor 2 field, the sign is not moved.

Three examples in Figure 58 illustrate the function of the MOVEL operation.

IBM

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

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And / And / Not / Not / Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
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
01	C											
02	C		optional	blank	MOVEL	FLDA	FLDB	n	n		blank	
03	C											
04	C											
05	C											

Figure 57. Format of MOVEL Operation

Operation	Factor 2	Result Field	Field Length	Decimal Positions	Factor 2	Result Field (after move)
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		
MOVEL	'DATA'	FLDA	6		DATA	DATAxx
MOVEL	FLDA	FLDB	81		98765	98765xxx
MOVEL	-13.57.65	FLDC	42		13576 5	135 7

Figure 58. Functions of MOVEL Operation

Move Zone

This operation has four variations. In each case, the zone portion of the specified position (L: Low-order, H: High-order) in the field in Factor 2 is moved to the specified position (L: Low-order, H: High-order) of the result field.

Factor 1 is not used in this operation. The field which is specified to use the high zone (H) can only be an alphameric field.

Move Low-to-High Zone (MLHZO). This operation moves the zone in the rightmost position of the field in Factor 2 to the leftmost position of the result field. Factor 2 can be numeric or alphameric, but the result field must be alphameric.

Move High-to-Low Zone (MHLZO). This operation moves the zone in the leftmost position of the field in Factor 2 to the rightmost position of the result field. The field in Factor 2 must be alphameric. If the field in Factor 2 contains only numeric data, this operation can still be performed, provided that the field is defined as an alphameric field. The result field may be numeric or alphameric.

Move High-to-High Zone (MHHZO). This operation moves the zone in the leftmost position of the field in Factor 2 into the leftmost position of the result field.

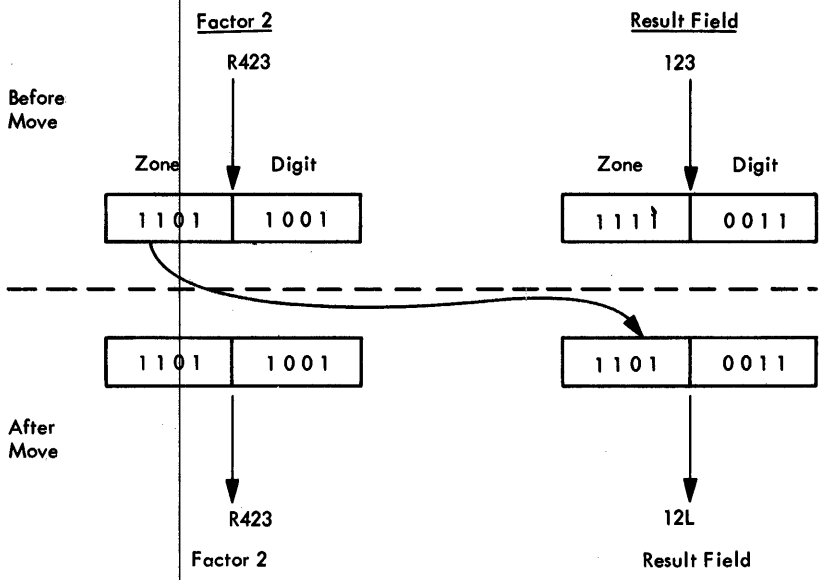


Figure 59. Function of the Move High-to-Low Zone (MHLZO) Operation

Factor 2 and the result field must be defined as alphameric.

Move Low-to-Low Zone (MLLZO). This operation moves the zone in the rightmost position of the field in Factor 2 into the rightmost position of the result field. Factor 2 and the result field may be alphameric or numeric. A result field specified as numeric always contains the correct sign after this operation.

Figure 60 illustrates the functions of the move-zone operations. The zones are moved from Factor 2 to the result field.

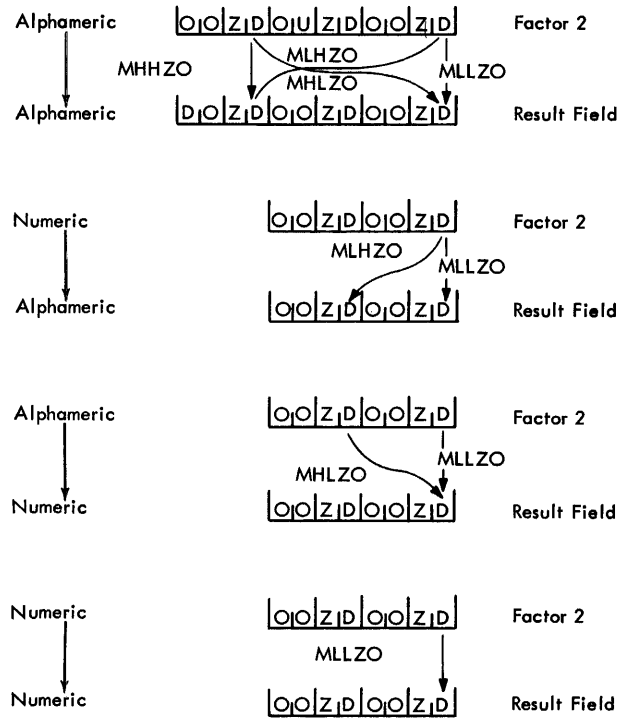


Figure 60. Function of Move Zone Operations

COMPARE AND TEST OPERATIONS

Compare (COMP)

This operation causes the literal or the contents of the field in Factor 1 to be compared against the literal or the contents of the field in Factor 2.

The compare operation turns on the appropriate indicator specified in columns 54-59. These resulting indicators can be used to condition calculation and/or output operations.

If the literal or the data in the field in Factor 1 is greater than the literal or the data in the field in Factor 2 (F1>F2), the indicator specified in High (columns 54-55) is turned on.

If the literal or the data in the field in Factor 1 is smaller than the literal or the data in the field in Factor 2 (F1<F2), the indicator specified in Low (columns 56-57) is turned on.

If the literal or the data in the field in Factor 1 equals the literal or the data in the field in Factor 2 (F1=F2), the indicator specified in Equal (columns 58-59) is turned on.

A result field must not be used in compare specifications.

The following considerations apply to the COMP operation:

1. Alignment of the Factor 1 and Factor 2 fields depends on whether they are numeric or alphameric.
   When numeric fields are compared, fields of unequal length are aligned at the implied decimal point. Excess positions in numeric fields are assumed to be filled with zeros.
   When alphameric fields are compared, fields of unequal length are aligned at their leftmost characters. The excess right-hand positions of the shorter field are assumed to be blank.
2. For numeric fields, the maximum length is 14 digits.
3. For alphameric fields, the maximum length is 40 characters.

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 61 shows the example of comparing the absolute value of a sum to a literal. The ADD operation in line 01 of the example places the arithmetic sum of the contents of FACT1 and FACT 2 into the result field FACT2. Indicator 21 is turned on if the sum is negative. In this case, line 02 contains the next operation to be performed: the negative of the contents of FACT2 is placed into the compare field CFLD. If FACT2 is positive (or zero), the next operation to be executed is specified in line 03. The actual value contained in FACT2 is moved into CFLD. Line 04 contains the COMP statement that compares CFLD with the literal 10000.0 and turns on one of the indicators 31, 32, or 33, depending on the result of the compare operation.

Test Zone (TESTZ)

The TESTZ operation is used to test the zone of the leftmost (high-order) position of the field whose name is entered in Result Field. This operation is restricted to alphameric fields and can be used to test the zone portion of any of the 256 EBCDIC characters. The TESTZ operation turns on one of the indicators specified in columns 54-59, depending on what zone it has encountered. The TESTZ operation is normally used to test for an 11 or 12 punch that was present in the high order position of a field from an input card file record.

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9 LR) 7 8	Indicators: And / And — Not 9 10 11	Not 12 13 14	Not 15 16 17	Factor 1 18 19 20 21 22 23 24 25 26 27	Operation 28 29 30 31 32	Factor 2 33 34 35 36 37 38 39 40 41 42	Result Field 43 44 45 46 47 48	Field Length 49 50 51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus / Minus / Zero or Blank — Compare: High 1>2 / Low 1<2 / Equal 1=2 54 55 / 56 57 / 58 59	Comments 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
0 1	C					FACT1	ADD	FACT2	FACT2	8	2		21	
0 2	C		21				Z-SUB	FACT2	CFLD	8	2			
0 3	C		N21				Z-ADD	FACT2	CFLD					
0 4	C					CFLD	COMP	10000.0					31 32 33	
0 5	C													

Figure 61. Example of an Absolute Compare Routine

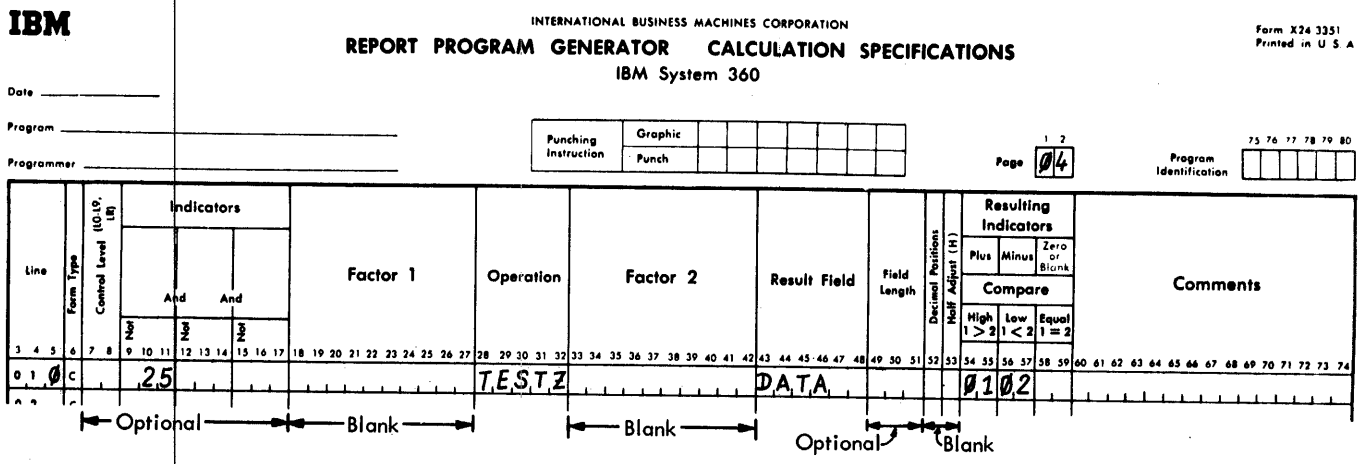


Figure 62. Example of Using the TESTZ Operation

If a 12-punch has been encountered (&, or any of the characters having the same zone as the letter A), the indicator in columns 54-55 is turned on. If an 11-punch has been encountered (-, or any of the characters having the same zone as the letter J), the indicator in columns 56-57 is turned on. Any other zone causes the indicator in columns 58-59 to be turned on.

Factor 1, Factor 2, and columns 52-53 are not used in test-zone operations.

Figure 62 shows an example of this operation. When indicator 25 is on, the field DATA is tested. If the high-order position has an 11-zone, indicator 02 is turned on. If the high-order position has a 12-zone, indicator 01 is turned on.

TURNING INDICATORS ON OR OFF

This section describes the SETON and SETOF operation. The column headings of Plus, Minus, and Zero and High, Low, and Equal have no meaning with regard to these operations, and should be ignored. In SETOF and SETON specifications, columns 54 through 59 are used only to record three indicator codes.

Set Indicators On (SETON)

This operation code causes the indicators specified in columns 54-59 to be turned on. One use of this specification is to turn on a halt indicator. All RPG indicators except L0, 1P, and MR can be turned on. Figure 63 (line 010) shows an example of this facility.

Indicator 01 is turned on whenever a specific record type is read. The L3 is a control-level indicator that is turned on when a level 3 control break has occurred.

If a level 3 control break is caused by a record type that is not associated with the record identifying indicator 01, halt indicator H1 is turned on.

Set Indicators Off (SETOF)

This operation code causes the indicators specified in columns 54-59 to be turned off. All RPG indicators except L0, 1P, and MR can be turned off. Figure 63 (line 020) shows an example of this facility. Indicator 11 is turned on at a specified time. L1 is a control level indicator that is turned on when a level 1 control break occurs. Whenever both the indicators L1 and 11 are on, the halt indicator H1 is turned off.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 01

Program Identification: REPORT

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And / And (Not)	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank	Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
010	C		L3N01		SETON							H1	
020	C		L1 11		SETOF							H1	
030	C												

Figure 63. Example of Using the SETON and SETOF Operations

Note:

1. If the LR indicator is turned on during total calculations by a SETON operation, processing is terminated after the output of the total lines.

2. If the OV or OF indicator is changed by a SETON or SETOF operation during total or detail calculations, the RPG object program resets the indicator at the end of the associated total detail, or exception output. Therefore, overflow printing will not be affected.

3. The indicators H1-H9, L1 through L9, and all record identifying indicatcrs that are defined in columns 19-20 of the Input Specifications form can be turned on by a SETON operation. However, they will all be turned off by the RPG object program following detail output.

4. Setting indicators L1 to L9 on or off will not automatically set the lower-level indicators on or off.

TABLE OPERATIONS

Table Look-Up (LOKUP)

The LOKUP operation causes the literal or the field name in Factor 1 to be used as a search argument in a look-up operation. Factor 2 contains the name of the table to be searched, and the result field may contain the name of the table from which the associated function is to be obtained. Decimal alignment is performed for this operation if the table is numeric. The result field may be left blank if the associated function is not to be retrieved. Indicator setting refers to the table argument, rather than the search argument. Refer to the section Using Tables in the Object Program for a complete discussion of tables.

CHAIN OPERATIONS

Chain (CHAIN)

The CHAIN operation causes a record to be retrieved from the file named in Factor 2, by using the contents of the field named in Factor 1 to identify the record. The CHAIN operation allows chaining to be performed on one file several times. The chaining field (Factor 1) may be either an input field or a calculation result field. The CHAIN operation may be executed at either detail or total time. (Refer to the section Chaining.)

The CHAIN statement consists of the entry CHAIN in Operation, and entries in Factor 1 and Factor 2. The entry in Factor 1 varies depending upon the type of file being chained to:

Indexed-Sequential Organization

Factor 1 contains the name of the chaining field. It must contain the key (e.g., customer number, item number, etc.) which is to be used to retrieve the record.

Sequential Disk Organization

Factor 1 contains the name of a field or a literal which contains the relative record number in the file. It must be a numeric field or literal. The number supplied (e.g., 151 for

the one hundred and fifty first record) must be supplied by the user.

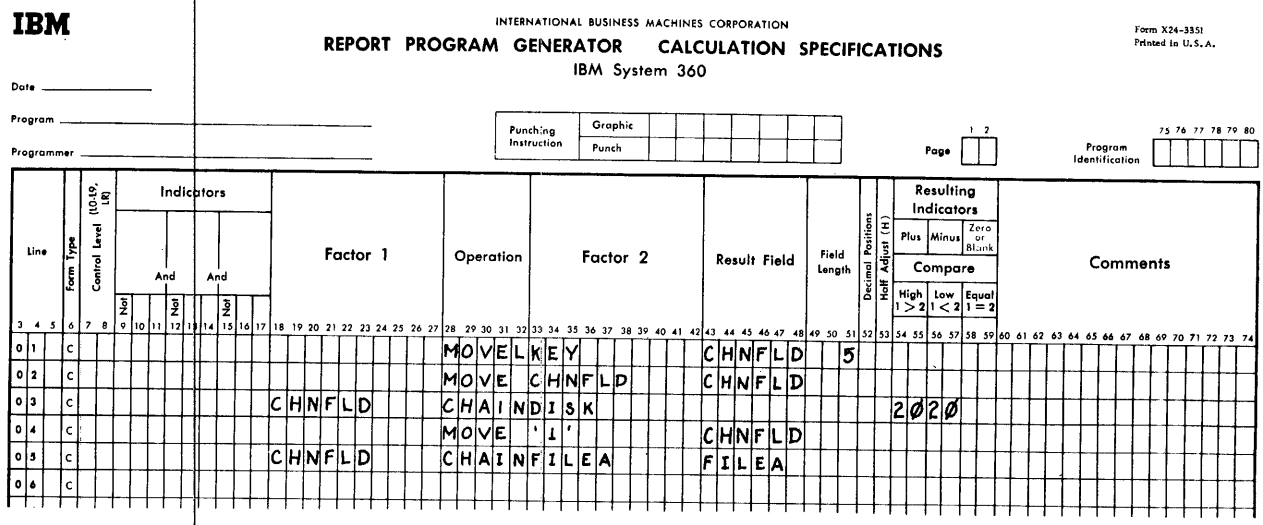
The entry in Factor 2 defines the chained file, and must be a Filename as specified on the File Description sheet. The CHAIN statement may be conditioned by control level in columns 7-8, and by indicators specified in columns 9-17.

Resulting indicators may be used to identify a not-found condition on the chained file. The same indicator should be specified in columns 54-55 and columns 56-57. Columns 58-59 should be blank. This indicator will be turned on whenever a chained record is not found. Not found is described as either: a. ISAM - no such key exists in the file or b. Sequential - the relative record number is either minus, 0, or larger than the largest record number of the file. See Figure 64 for an example of the CHAIN statement.

Note: If the indicator is not present, and a chained record is not found, the RPG program will notify the operator. The operator is given the option of canceling the job or bypassing the not-found condition and retrieving the next record in the RPG cycle. If the user does this, heading and detail output will be bypassed as well as the remaining calculations in the cycle.

The record indicator for any chained file is turned off following detail output. If the user chains to the same file more than once in the same cycle, there may be more than one record identifying indicator on for that file. This can be prevented by issuing a SETOF prior to chaining.

Example 1 The field CHNFLD is from an input record and is chained to the filename DISK. If a record is not found with a similar key as that of CHNFLD, indicator 20 is turned on. This indicator can then be used to bypass normal operations. In this example, indicator 20 is used to jump over normal calculations.


IBM

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

Program Identification

Line	Form Type	Control Level (L0,L9, LR)	Indicators: And / And / Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C				MOVELKEY		CHNFLD	5				
02	C				MOVE CHNFLD		CHNFLD					
03	C			CHNFLD	CHAINDISK						2020	
04	C				MOVE '1'		CHNFLD					
05	C			CHNFLD	CHAINFILEA		FILEA					
06	C											

Figure 64. Example 1 Using the CHAIN Operation

IBM

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

Program Identification

Line	Form Type	Control Level (L0,L9, LR)	Indicators: And / And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C				Z-ADD0		CHNFLD	60				
02	C				MOVELCUST							
03	C				TAG	LOOP						
04	C			CHNFLD	CHAIN	MASTCUST					202	
05	C		N20									
06	C		N20	1	ADD	CHNFLD	CHNFLD					
07	C		N20		GOTO	LOOP						
08	C											

Figure 64. Example 2 Using the CHAIN Operation

Example 2. In this example, the indexed sequentially organized file ISAM is made up of a variable number of records per customer. The customer record has a key field of six digits. The first 5 are the actual customer number and the units position is a digit 0 to 9 depending upon how many records the customer has. The field CHNFLD is set up for the first customer record with a units position of zero. The first customer record is chained to and indicator 20 is used for the not found condition. If 20 is not on, an exception output line occurs to print the information from the first record. Then a 1 is added to CHNFLD and GOTO is used. This process continues until no more records are found for this customer and indicator 20 is turned on. The program then continues to any other detail calculations. In this particular example the user would have to use the record identifying indicator of the chained file to determine whether a not found condition exists when the units position of the chaining field is zero.

Example 3. In this example, Figure 64, a sequential file is being retrieved by use of a relative record number. Since no index relates to a sequential file, the programmer must have some means of obtaining the relative record number based upon a customer number. This can normally be done by one of three methods:

a. The actual customer number is used as the relative record number (this method depends on the coding structure used).

b. A mathematical formula is used to convert the customer number to a relative record number.

c. A table lookup approach is used to locate the assigned relative record number.

In this example, the table look-up technique is used. Indicator 20 determines whether the customer number is in the table, and the appropriate processing that is necessary if it is not.

IBM

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 (L0-L9, LR)	Indicators: Not	And	Not	And	Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Minus	Zero or Blank	Compare: High 1>2	Low 1<2	Equal 1=2	Comments
01	C							CUST	LOKUPTABCUS		TABNUM									2Ø	
02	C		N2Ø						GOTO END												
03	C							TABNUM	CHAINDISK												
04	C																				
05	C																				
06	C																				
07	C							END	TAG												
08	C																				

Figure 64. Example 3 Using the CHAIN Operation

BRANCH AND EXIT OPERATIONS

Branch (GOTO)

The GOTO operation causes a branch from one point in the program to another. The point of origin of such a branch is the GOTO operation. The point-of-destination is a TAG specification that is assigned a name, which is entered in Factor 2 of the GOTO operation. The mnemonic GOTO is entered in Operation (columns 28-31). For example, a routine (a sequence of specifications) to calculate the employees' social security deductions might be named FICA. For a branch to this routine, Factor 2 of the GOTO specifications must contain the name FICA; and the first specification in the routine must be a TAG specification defining the name of the routine. (Refer to the description of the TAG operation code).

Branches may be forward (skipping subsequent specifications) or backward (returning by skipping preceding specifications).

Format of GOTO. GOTO statements consist of the entry GOTO in Operation and a name in Factor 2. GOTO statements may have a control-level indicator, and/or one, two, or three indicators in columns 9-17. All other columns of a GOTO statement must be blank. The GOTO statement is executed only if the conditions established in the indicator and control-level specifications are satisfied.

The name in Factor 2 may have a maximum length of six characters and it must be left-justified. Its first character must be alphabetic, the remaining characters may be alphabetic and/or numeric. Special characters and embedded blanks must not be used. The name must be unique. That is, there may be only one TAG statement with the same name. However, more than one GOTO statment may refer to this name.

The name of the TAG statement cannot be the same as one which is used for a field or file name.

A GOTO from outside of a subroutine cannot be issued to a tag which is within a subroutine.

A GOTO may be issued from detail to total calculations, but this is normally not needed due to the EXCPT operation code.

Providing a Name for a GOTO (TAG)

The TAG specification marks the point to which the program can branch. Enter a name in Factor 1 and the code TAG in Operation (columns 28-32). The same name must be used in Factor 2 of the associated GOTO operation. A tag statement may appear in detail or total calculations or within a closed subroutine. Columns 7-8 may have any valid entry. Columns 9-17 and 33-59 must be blank for a TAG statement.

EXAMPLE. Figure 65 illustrates the use of GOTO and TAG specifications.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

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 (LO-L9, LR)	Indicators: And / And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank	Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C		N10		GOTO	ADDR01							
02	C		06 05		GOTO	ADDR02							
03	C			ADDR03	TAG								
04	C												Some sequence of specifications
05	C												
06	C			ADDR01	TAG								
07	C												Some sequence of specifications
08	C												
09	C		05		GOTO	ADDR03							
10	C	L0		ADDR02	TAG								TAG for total calculations
11	C												Some sequence of specifications
12	C												
13	C	L0	06		GOTO	ADDR02							
14	C		05		GOTO	ADDR03							

Figure 65. Example of Using the GOTO Operation

The GOTO statement in line 01 causes control to be transferred to ADDR01 on line 05 if indicator 10 is not on. If indicator 05 and 06 are on, the GOTO statement in line 02 causes control to be transferred to ADDR02 on line 08. If indicators 05 and 06 are not on, the next statement in sequence is performed.

The GOTO statement in line 11 causes control to be transferred to ADDR03 on line 15, if the indicators L1 and 08 are on. Both the GOTO and the TAG are at L1 time.

Note: A literal must not be entered in the factor fields of a GOTO or TAG operation.

Transfer to an RPG Subroutine (EXSR)

The EXSR operation causes a branch from any point in the program to a subroutine in the program. The subroutine referenced must be an internal subroutine, written in RPG as a part of this program. (Refer to the section Subroutines.) After the subroutine has been performed, a branch occurs back to the next calculation specification following the EXSR operation.

Format of EXSR. EXSR statements consist of the entry EXSR in Operation and a name in Factor 2. This name is that of the subroutine being referenced. EXSR statements may have a control-level indicator, and/or one, two, or three indicators in columns 9-17. All other columns of a EXSR statement must be blank. The EXSR operation is executed only if the conditions established in the indicator and control-level specifications are satisfied.

The EXSR operation may itself be part of an RPG subroutine and refer to another subroutine. In this case, columns 7-8 would contain SR. However, the name in Factor 2 must not be the name of the subroutine of which the EXSR operation is a part. (A subroutine may not call itself.) See Figure 66 for an example of the EXSR statement.

Begin RPG Subroutine (BEGSR)

The BEGSR specification appears as the first statement of an RPG subroutine. Such a subroutine is written in RPG and is a part of the program which uses it. (Refer to the section Subroutines.) The BEGSR specification defines the logical beginning

of the subroutine. Also, it defines the name in Factor 1 as the entry point of the subroutine.

BEGSR statements consist of the entry BEGSR in Operation and a name in Factor 1. This name is the name referenced by EXSR statements. Columns 7-8 must contain SR. All other columns must be blank. See Figure 66 for an example of the BEGSR statement.

End RPG Subroutine (ENDSR)

The ENDSR specification appears as the last statement of an RPG subroutine. The ENDSR serves to delimit the RPG subroutine and to identify the exit point from the subroutine. A branch occurs at this point to return to the specification immediately following the EXSR statement that invoked this subroutine.

ENDSR statements consist of the entry ENDSR in Operation. Factor 1 may be used to provide a name for a GOTO within the subroutine. Columns 7-8 must contain SR. All other columns must be blank. See Figure 66 for an example of the ENDSR statement.

In Figure 66, two RPG subroutines are shown, SUBR1 and SUBR2. Lines 01 and 03 show conditional calls to the subroutines at detail time. If the associated indicator is on, the subroutine will be executed. Line 05 shows a call to SUBR1 at total time. The subroutines perform the desired function and return to the statement following the EXSR associated with the subroutine. Line 13 shows a conditional GOTO to the ENDSR statement. If indicator 17 is set on, the GOTO will transfer to the ENDSR statement which will return to the calling EXSR.

Exit to an External Subroutine (EXIT)

The EXIT operation causes control to be transferred from the RPG program to an external subroutine written by the user. Factor 2 contains the name of this subroutine. (Refer to the section Subroutines.)

IBM

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 (L0-L9, LR)	Indicators: And Not	And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Minus	Zero or Blank	Compare: High 1>2	Low 1<2	Equal 1=2	Comments
01	C		05			EXSR	SUBR1											
02	C					•												
03	C		22			EXSR	SUBR2											
04	C					•												
05	C	L1				EXSR	SUBR1											
06	C					•												
07	C	SR			SUBR1	BEGSR												
08	C	SR				•												
09	C	SR				•												
10	C	SR				ENDSR												
11	C	SR			SUBR2	BEGSR												
12	C	SR				•												
13	C	SR				GOTO	END2											
14	C	SR				•												
15	C	SR			END2	ENDSR												
	C																	

Figure 66. Example Using the BEGSR, ENDSR, and EXSR Operations

RPG Label (RLABL)

RLABL operation is used in RPG programs to define fields or indicators that are to be referred to in external user's subroutines. (Refer to the section Subroutines.)

Place the code RLABL in Operation, and the name of the field that is to be defined in Result Field. To define an indicator, place the code IN followed by the indicator code in Result field. The entry in Result Field must be left-justified. Factor 1 and Factor 2 must not be used in RLABL specifications.

RLABL statements must be specified following the EXIT statement which refers to the subroutine that uses the field or indicator defined by the RLABL.

Therefore, if the need exists to exit to the same subroutine from more than one place in the program, it can best be coded in an RPG subroutine.

Output Records During Calculations (EXCPT)

Occasionally it is necessary for the programmer to vary the fixed logic flow of RPG. A typical example is when the programmer requires several lines of output (fixed or variable number) that are identical (or almost identical). A fixed number of output lines can be created by RPG's normal logic, but this may entail duplication of coding.

To eliminate duplicate coding and to provide for unusual processing situations, RPG allows for exception output.

The EXCPT operation causes exception records (E in column 15 of the Output-Format Specification) to be produced during calculations. This allows producing a variable number of output lines which are identical (such as Tub File Replenishment), without repetitively coding the output specification for the records. Similar output records can also be produced without repeating the entire record, by changing selected fields before placing the subsequent record in the output file.

An EXCPT statement consists of the entry EXCPT in Operation. The statement may be conditioned by control levels (column 7-8), or indicators (columns 9-17). The remaining columns should be blank. Execution of this statement causes all exception records whose indicator tests are met to be placed in the output file at this time. These exception records are produced in the order of their specification on the Output-Format Specification sheet.

Example 1. In this example, Figure 67, the EXCPT operation is used to create a variable number of Tub File Replenishment cards. The master card is read and contains data to be reproduced in columns 1-77. The variable number of cards to be punched is contained in a field called VARFLD. The Calculation Specifications show a series of steps that form a loop. A field called COUNT is created and set to zeros. It is then compared against the variable count field. If not equal, the exception output lines are requested by the EXCPT operation.

Exception Output (E in column 15) lines are tested for indicator conditions which are met. In this case, a card is to be punched with the contents of the field DATA. After all exception output lines are tested (only one in this case), control is transferred to the calculation specifications at the step immediately following EXCPT (line 05). A one is added to COUNT and a GOTO (branch) returns to the TAG operation labeled LOOP. The process is repeated again until the field COUNT is equal to the field VARFLD. Each time it is not equal, exception output is called for and another card is punched.

Example 2. In Figure 67.1 exception output is called from two places in the program and under different conditions. Line 04 sets on indicator 12 which is used to condition the file called PRINTER. Line 06 sets off indicator 12 to prevent this same output from occurring at the wrong time. Lines 12 and 14 use indicator 13 to control a file called PUNCH in the same manner. The two EXCPT statements (lines 05 and 13) both call for exception output to occur. Line 05 is performed at detail calculations and line 13 at total calculations on a L1 control break.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
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-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	CARD	AA			01																									
02	I																				L	77		DATA							
03	I																				78	80		VARFLD							
04	I																														

IBM

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 (LO-L9, LR)	Indicators: And Not	Indicators: And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1 > 2	Compare: Low 1 < 2	Compare: Equal 1 = 2	Comments
01	C					Z-ADD0		COUNT	30									
02	C				LOOP	TAG												
03	C				VARFLD	COMP	COUNT										21	
04	C		N21			EXCPT												
05	C		N21		1	ADD	COUNT	COUNT										
06	C		N21			GOTO	LOOP											
07	C																	

IBM

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	Type (H/D/T)	Space: Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And Not	Output Indicators: And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	PUNCH	E						01N21								
02	O										DATA			77			
03	O																

Figure 67. Example Using the EXCPT Operation

IBM

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 (L0-L9, LR)	Indicators: And	Indicators: And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1 > 2	Compare: Low 1 < 2	Compare: Equal 1 = 2	Comments
01	C																	
02	C																	
03	C																	
04	C					SETON									12			
05	C					EXCPT												
06	C					SETOF									12			
07	C																	
08	C																	
09	C																	
10	C																	
11	C																	
12	C	L1				SETON									13			
13	C	L1				EXCPT												
14	C	L1				SETOF									13			
15	C																	

IBM

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	Type (H/D/T)	Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And	Output Indicators: And	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	PRINTER	E		1				12								
02	O																
03	O																
04	O																
05	O	PUNCH	E						13								
06	O																
07	O																
08	O																
09	O																

Figure 67.1 EXCPT Operation - Example 2

Figure 68 is a summary of the RPG operation codes and associated entries.

Operation Codes Operation	Control Level	Conditioning Indicators	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust	Resulting Indicators
DECIMAL										
ADD	O	O	N	ADD	N	N	O	O	O	O
ZERO AND ADD	O	O		Z-ADD	N	N	O	O	O	O
SUBTRACT	O	O	N	SUB	N	N	O	O	O	O
ZERO AND SUBTRACT	O	O		Z-SUB	N	N	O	O	O	O
MULTIPLY	O	O	N	MULT	N	N	O	O	O	O
DIVIDE	O	O	N	DIV	N	N	O	O	O	O
MOVE REMAINDER	O	O		MVR		N	O	O		O
MOVES										
MOVE	O	O		MOVE	E	E	O	O		
MOVE LEFT	O	O		MOVEL	E	E	O	O		
MOVE HIGH-TO-LOW ZONE	O	O		MHLZO	A	E	O	O		
MOVE LOW-TO-HIGH ZONE	O	O		MLHZO	E	A	O			
MOVE HIGH-TO-HIGH ZONE	O	O		MHHZO	A	A	O			
MOVE LOW-TO-LOW ZONE	O	O		MLLZO	E	E	O	O		
LOGICAL										
COMPARE	O	O	E	COMP	S					R
TEST ZONE	O	O		TESTZ		A	O			R
TABLE LOOKUP	O	O	E	LOKUP	S	O	O	O		R
SET INDICATORS										
SET INDICATORS ON	O	O		SETON						R
SET INDICATORS OFF	O	O		SETOF						R
FILE										
CHAIN TO A FILE RANDOMLY	O	O	E	CHAIN	I					O
BRANCH TO EXCEPTION OUTPUT LINE	O	O		EXCPT						
BRANCHING										
BRANCHING OR GOTO	O	O		GOTO	L					
PROVIDING LABEL FOR GOTO	O		L	TAG						
BRANCH TO CLOSED SUBROUT	O	O		EXSR	L					
BEGIN CLOSED SUBROUT	SR		L	BEGSR						
END CLOSED SUBROUT	SR		OL	ENDSR						
EXIT TO A SUBROUT	O	O		EXIT	L					
RPG LABEL	O			RLABL		E	O	O		

KEY TO OPERATION CODE CHART

IF THE ENTRY IS BLANK IT CANNOT BE FILLED IN

E = REQUIRED FIELD MUST BE EITHER ALPHAMERIC OR NUMERIC
A = REQUIRED FIELD MUST BE ALPHAMERIC
I = REQUIRED FILE NAME
L = REQUIRED LABEL NAME
N = REQUIRED FIELD MUST BE NUMERIC
O = OPTIONAL ENTRY
OL= OPTIONAL LABEL ENTRY
R = REQUIRED ENTRY
S = REQUIRED FIELD MUST BE OF THE SAME TYPE (ALPHAMERIC OR NUMERIC) AS FACTOR 1
SR= REQUIRED ENTRY IN CONTROL LEVEL COLUMNS

Figure 68. Summary of RPG Operation Codes

IBM

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 03

Program Identification

Line	Form Type	Control Level (L0,L9, LR)	Indicators: And Not	And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Minus	Zero or Blank	Compare: High 1>2	Low 1<2	Equal 1=2	Comments
01 Ø	C				HOURS	COMP	Ø										15	
02 Ø	C				HOURS	COMP	40								20			
03 Ø	C		20		HOURS	SUB	40	OVERHR	31									
04 Ø	C		20		RATE	MULT	OVERHR	SAVE	62	H								
05 Ø	C		20		SAVE	MULT	1.5	SAVE		H								
06 Ø	C		N15		RATE	MULT	40	GROSS	62	H								
07 Ø	C		20		SAVE	ADD	GROSS	GROSS										
08 Ø	C		N15			GOTO	FICA											
09 Ø	C				(Additional Calculations)													
10 Ø	C				FICA	TAG												
11 Ø	C				GROSS	MULT	.0485	DFICA	62	H								
12 Ø	C				YDFICA	ADD	DFICA	HOLD	62	H								
13 Ø	C				HOLD	COMP	37,4 .40								21	21		
14 Ø	C		21			GOTO	ADFICA											
15 Ø	C				37,4 .40	SUB	YDFICA	DFICA										
16 Ø	C				ADFICA	TAG												
17 Ø	C				YDFICA	ADD	DFICA	YDFICA										
18 Ø	C				(Additional Calculations)													
19 Ø	C				WHTAX	TAG												

Figure 69. Example of Using the Calculation Specifications Form

USING THE CALCULATION SPECIFICATIONS FORM

Figure 69 illustrates entries on the Calculation Specifications form that are used in part of a payroll application. The entries of 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.

Page-Line Explanation

0301 The program compares 0 to the number of hours worked by the employee. If the result is equal, indicator 15 is turned on.

0302 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.

0303-0305 If indicator 20 is on, three calculations are performed. The literal 40 is subtracted from the number of hours worked. Then the overtime hours are placed in the field OVERHR. OVERHR is a three sposition field with one decimal position. RATE is multiplied by OVERHR and the result is placed in the field SAVE. SAVE is a six-position field with two decimal positions. SAVE is multiplied by the literal 1.5 (the overtime premium rate). The result is half-adjusted and is placed back in SAVE.

0306 RATE is multiplied by 40, and the result is stored in GROSS. This operation is performed whether or not the employee worked any overtime. This operation is not performed if

the employee did not work any hours during the pay period.

0307 GROSS is added to SAVE.

0308 The program branches to the label FICA if indicator 15 is not on.

0309 Additional calculations not used in this example.

0310 The operation code TAG provides the label FICA. The RPG program branches to this label from 0308.

0311 GROSS is multiplied by the literal .048 and the result is placed in the field DFICA. DFICA is a six-position field with two decimal positions. The result is half-adjusted.

0312 DFICA is added to YDFICA and the result is placed in the field HOLD.

0313 The contents of HOLD are compared with the literal 374.40. If HOLD is less than, or equal to, 374.40, indicator 21 is turned on.

0314 If indicator 21 is on, the program branches to the label ADFICA.

0315 YDFICA is subtracted from the literal 374.40. The result is placed in DFICA.

0316 The operation TAG provides the label ADFICA to which the program can branch (either from the specification on line 0314 or sequentially from the specification on line 0315).

0317 DFICA is added to YDFICA and the result is stored in YDFICA.

0318 Additional calculations not used in this example.

0319 The operation code TAG provides the label WHTAX to which the RPG program may branch.

This form specifies the kind of output files to be produced and the location of the specific data fields in the output reports and records.

The specifications for this form are divided into two categories as illustrated in Figure 70: file identification and control (columns 7-31), and field description (columns 23-74, where columns 71-74 are used in conjunction with sterling currency processing).

File Identification and Control. These specifications identify the output files (disk, printer and/or punched card files), and records to be added to the file. They direct cards to the appropriate stackers and provide for correct spacing on printed reports. They determine under what conditions and at what time (detail, total, exception or overflow time) the records are to be produced.

Field Description. These specifications indicate where and when the individual fields of the output record are to be punched, printed, or written on disk. The entries for these specifications are written on the lines below the file identification entries. Each field is described on a separate line.

The reader should note that the specification Output Indicators is used for both file identification and field description. The facility of controlling the printing and punching of each specific field of the record provides great program flexibility in RPG.

Sequence of Specifications

The sequence of specification should be as follows:

1. Heading and/or Detail lines.
2. Total lines.
3. Exception lines.

The sequence of the output operations is equal to the sequence of the definitions on the Output-Format Specifications form. The sequence for the output specifications depends upon the order of the events to be recorded. In a printed report, for example, the specifications for a level 1 total would be listed first followed by the specifications for the level 2 totals, etc. If this data is to be both printed and punched, specifications for punching and printing lines should be listed alternately. This enables the object program to overlap the punching and printing operations.

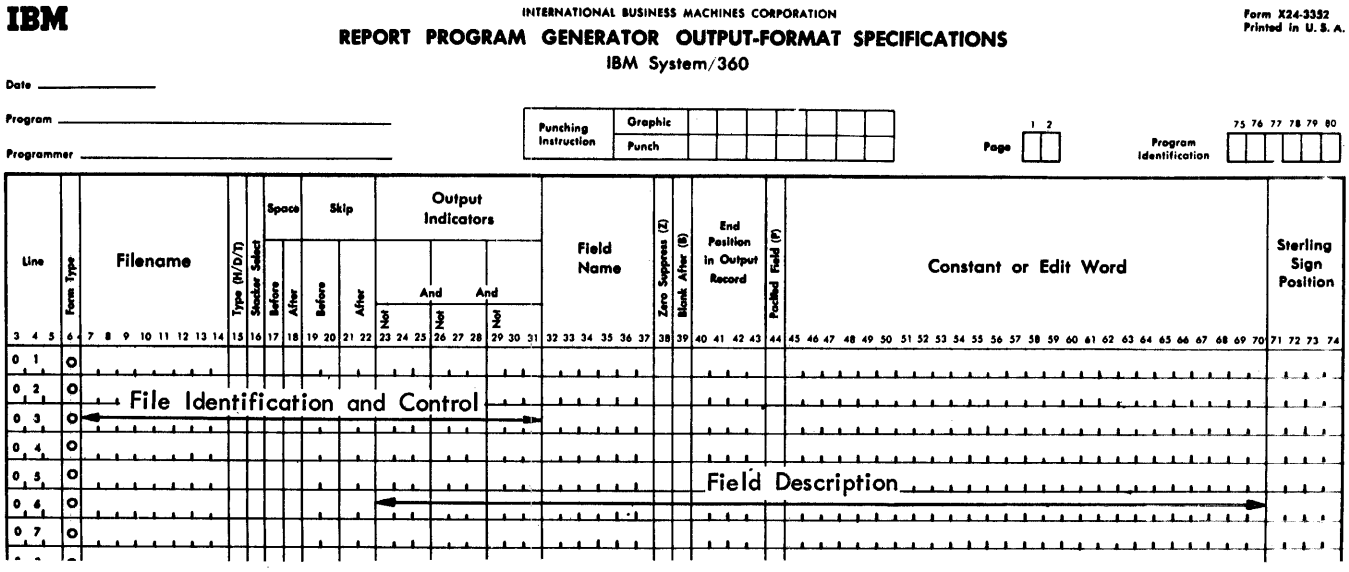


Figure 70. The RPG Output-Format Specification Form

If a detail card is to be punched, and selected into a specified stacker, all of the specifications for these operations must be given as part of one file identification line.

Specifying Output Units

When writing the output format specifications, it is not necessary to indicate the specific output units used in the program. The input/output unit used for each file is specified in the File Description Specifications form. Each file name, therefore, is related to a specific output unit. By merely writing the file name on the output form, the output unit has, in effect, been specified.

Writing the specifications in the proper sequence should not be difficult if (1) the layout of a printed report has been correctly made on a Printer Spacing Chart or proportional record layout form, and (2) reference is made to the spacing chart or layout form as the specifications are written.

FILE IDENTIFICATION AND CONTROL

File Name (Columns 7-14)

A filename must be assigned to each output file. The name may contain up to eight alphabetic and/or numeric characters and must be left-justified. The first character must be alphabetic, and the name must not contain embedded blanks or special characters. 1130 RPG utilizes only the first 5 characters.

In file maintenance operations where an input record is updated on the basis of newly calculated results (update or combined files) the same filename must be used for both input and output specifications.

When writing the specifications for output records, the filename must be given only on the first line for the file. On subsequent lines for the same file, the filename need not be specified. However, the filename must be repeated each time another output file is specified on the preceding lines. Figure 71 illustrates this point. (Field specifications are not shown.) The first and second specification lines cause the printing of the overflow heading and detail lines. The third specification line causes the punching of a summary card; this line causes a new file to be created and must therefore be given a separate filename. The remaining specification lines cause the control level 1 and final total lines to be printed.

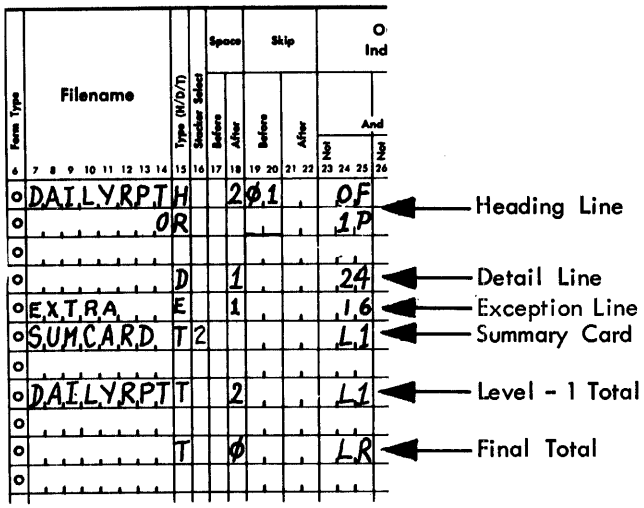


Figure 71. Sample of File Name and Type H/D/T/E Specifications

Type H/D/T/E (Column 15)

This specification identifies the type of record being specified. The following four entries are used for this specification:

H Heading Record.
D Detail Record.
T Total Record.
E Exception Record.

Heading Records. The records usually contain constant information, but 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 records. Most of the data in a detail line 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, an input record that causes a control field change cannot contribute data to total records that result from that control change.

Exception Records. These records are produced whenever the EXCPT operation is executed. This can occur at detail or total time, depending on the conditions imposed on the Calculation Specification line which contains the EXCPT operation.

Figure 71 contains examples of entries for the specification Type H/D/T/E.

Note: Heading and detail records are distinguished only for the user's convenience. The RPG program treats both specification types in the same way.

Sequence of Specifications

The sequence must be as follows:

H/D	Heading and Detail Lines
T	Total Lines
E	Exception Lines

ADDING RECORDS TO AN INDEXED SEQUENTIAL ORGANIZED FILE (COLUMNS 16-18)

Enter the characters ADD in columns 16-18, if the output record is to be added to an indexed sequential file. These columns must be left blank if the contents of the output record are to be used to update or to load an indexed sequential file. Adding to a file cannot occur in the same program in which the file is being retrieved or updated. It is possible to retrieve or update one file while a second file is being added to in the same program.

In 1130 RPG the key field for any indexed-sequential file must begin in position 1 of each record. This entry along with the length of the key field is described on the File Description Specifications. If the programmer describes the key field as being 5 positions long, the RPG program will use the first 5 positions of the record to be added as the new key field. In 1130 RPG, the key field can be either alphameric or numeric, but it cannot be packed.

Note: Column 66 of the related file-description specifications must contain an A if a record is to be added.

For card and/or printer specifications use these columns as described below.

Stacker Select (Column 16)

This specification causes card records to be selected into the stackers of the output units.

It is used when an input/output unit with more than one stacker is attached to the system. It is used only for output files or combined files. (Selecting cards from an input file is specified on the Input Specifications form.) New stackers may be specified in subsequent OR-lines. If in an OR-line column 16 (Stacker Select) is left blank, the card is directed to the normal pocket. If all OR-cards should go into the same stacker, the same specification must be made for each card.

The number of the stacker into which the card record is to be selected is written in this specification. The specific card record to be selected is identified in the specification Output Indicators. Figure 72 illustrates two examples of Stacker Select entries. Field entries are not shown.

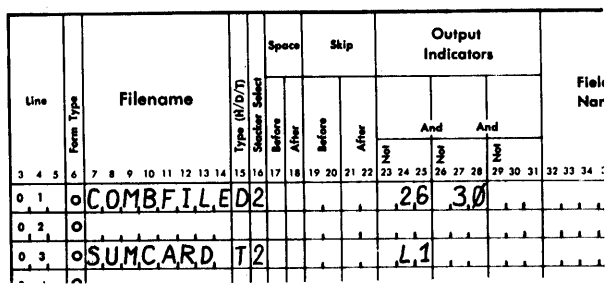


Figure 72. Sample of Stacker Select Specifications

The second specification causes the summary card created at control level-1 to be selected into stacker 2. The first specification causes a card record identified by indicators 26 and 30 to be selected into stacker 2. (Indicator 26 could be the record-identifying indicator for a particular card record. Indicator 30 could be a field indicator turned on by some special condition such as a zero or minus status in a field of the record.)

If no entry is made in Stacker Select, all cards from output and combined files are directed to a specific stacker, depending on the input/output unit attached to the system.

With the IBM 1442 Card Read-Punch, a blank or 1 in column 16 will cause cards to be directed to the normal stacker, while a 2 will result in the selection of stacker 2.

SPACE (COLUMNS 17-18)

This specification and the next specification (Skip, columns 19-22) are used to provide for the proper spacing of printed reports. Note that in OR-relationships, the Space entries may differ. If the record is to be printed, at least one entry is required in columns 17-22.

Space Before (Column 17). Zero, one, two, or three spaces before printing can be spe-

cified by placing the entries 0, 1, 2, or 3 in column 17.

Space After (Column 18). Zero, one, two, or three spaces after printing can be specified by placing the entries 0, 1, 2, or 3 in column 18.

Skip (Columns 19-22)

This specification provides for the proper spacing of printed reports. It is related directly to the operation of the 1132 or 1403 printer carriage control tape. Note that in OR-relationships, the Skip specification in the various OR lines may differ.

Skip Before (Columns 19-20). The entries 01 through 12 cause the printer carriage to skip to channels 1 through 12 of the carriage tape, respectively, before the line is printed.

Skip After (Columns 21-22). The entries 01 through 12 cause the printer carriage to skip to channels 1 through 12 of the carriage control tape, respectively, after the line is printed.

Note 1 At least one entry should be made in columns 17-22 if the record is to be printed.

Note 2. If Space After and Skip After are specified on the same specification line, only Skip After will be executed.

Note 3. A skip may not be specified for the console printer, or for channels 7, 8, 10, and 11 with the 1132 printer.

Note 4. A space before or space after of 0 may not be specified for the console printer.

OVERFLOW INDICATOR (OF OR OV)

Carriage overflow, indicated by a punch in channel 12 of the control tape, causes the overflow indicator specified on the File Description Specifications to be turned on. It remains on for one complete cycle of processing (thus it is turned off after the heading and detail lines of the next record are printed). Because the overflow indicator is on during calculations, it can be used to control calculation specifications.

A test for carriage overflow is made by the object program immediately before each line of the report is printed (but after any Space or Skip Before specifications are executed). Two conditions occur:

1. If a carriage overflow occurs at detail time, the remainder of the detail lines are printed. Then the overflow indicator is turned on and the next record is read. If the appropriate conditions are satisfied, total calculations and total output are performed. Then the overflow lines are printed or an automatic skip to the next page is executed if no overflow lines are specified. When the next detail calculations and detail output have been performed, the overflow indicator is turned off.

2. If a carriage overflow occurs at total time, the remainder of the total lines are printed. Then the overflow indicator is turned on. Overflow lines are printed if specified. Otherwise, an automatic skip to the next page is executed. When the next detail calculations and detail output have been performed, the overflow indicator is turned off.

Printing Overflow Lines. If an output line is coded as shown in the top half of Figure 73 the line will be printed whenever the OF or L1 indicator is on. If L1 and OF are on at the same time during processing, the line will be printed twice. This might cause an error in the printed report.

IBM

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 3 4 5	Form Type 6	Filename 7 8 9 10 11 12 13 14	Type (H/D/T) 15	Space: Stacker Select 16	Space: Before 17	Space: After 18	Skip: Before 19 20	Skip: After 21 22	Output Indicators: Not 23 24	And: 25 26	And: Not 27 28	And: Not 29 30 31	Field Name 32 33 34 35 36 37	Zero Suppress (Z) 38	Blank After (B) 39	End Position in Output Record 40 41 42 43	Packed Field (P) 44	Constant or Edit Word 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	Sterling Sign Position 71 72 73 74
0 1	O	OUTPUT	H				Ø1		OF										
0 2	O	OR							L1										
0 3	O															1ØØ		'ACCOUNT'	
0 4																			
0 5	O	OUTPUT	H				Ø1		OF	NL	1								
0 6	O	OR							L1										
0 7	O															1ØØ		'ACCOUNT'	
0 8	O																		

Figure 73. Example of Overflow Printing Specifications

To prevent this, the line is coded as shown in the bottom half of Figure 73. By making the conditions mutually exclusive, the line will not be printed twice.

Indicators OF and OV cannot be specified for an Exception Output line. The object program prints overflow lines in the following order:

1. Total lines conditioned by overflow are printed.

2. Heading and detail lines conditioned by overflow are printed.

Overflow Lines. An overflow line is any line conditioned by a positive overflow indicator. A line conditioned by OF (or OV) with or without any other indicator is an overflow line. It will be printed during overflow output time in the object flow. A line conditioned by NOF (or NOV) is not an overflow line. It is printed at either heading/detail, total or exception time.

Fields Conditioned by Overflow

A field description may also be conditioned by an overflow indicator. This field will be treated as any other field conditioned by an indicator. It will print only if printing is specified and the field indicator is on.

Note: If overflow lines are not specified on the report, a punch in channel 12 of the control tape causes the carriage to skip automatically to a punch in channel 1. If at least one overflow line has been specified, automatic skipping to a punch in channel 1 is not performed.

Multiple Printers. It is possible to use two printer files for each RPG object program. A unique overflow indicator must be specified for the second printer.

Output Indicators (Columns 23-31)

This specification may be used either for file and record identification (in this case at least one indicator is required) or field description. A maximum of three indicators may be specified. These indicators control:

1. When the record is to become output, and

2. When a particular field is to be printed, punched or written.

If more than one indicator is specified for one line, they are considered to be in an AND-relationship. Hence all conditions specified must be satisfied before the specified output operation can be executed.

If the object program requires more than three indicators in an AND-relationship, AND is entered in columns 14-16 of the following line, and the additional indicators are specified on that line. If one of these indicators is over-

flow (OF or OV), it must not appear on the same line as the AND.

If the output condition is executed in an OR-relationship, OR is entered in columns 14-15 of the following specification lines, and the OR indicators are specified on that line.

Additional specification lines can specify as many output indicators, in either an AND- or an OR-relationship, as required by the object program. Each additional line must begin with AND or OR in column 14. AND and OR can only be specified to condition records, and not fields.

Permitted Entries for Output Indicators, Columns 24-25, 27-28, and/or 30-31

Enter in these columns:

1. Record Identifying Indicators: A record identifying indicator codespecifies the particular input record type on which the output operation is to be performed. These indicators refer to entries in columns 19-20 of the Input Specification form.

2. Resulting Indicators: A resulting indicator code controls theoutput operation by conditions that have occurred during calculations. These indicators refer to entries in columns 54-59 of the Calculation Specifications form.

3. Field Indicators: A field indicator code controls theoutput operation by the status of the input field. These indicators refer to entries in columns 65-70 of the Input Specifications form.

4. Control Level Indicators: Control level indicators (L0...L9, LR) cause the output operation to be performed only when the affected control break has occurred.

5. Matching Record Indicator: The MR indicator code causes the output operation to be performed only if there is a matching record in a secondary file.

6. Halt Indicators: The halt indicators H1 through H9 arenormally used to suppress the output operation when an error has been detected in the input data or during calculation.

7. Overflow Indicators: The overflow indicators OF and OV cause the output operation to take place only if a page overflow has occurred. Note that these indicators do not apply to the first page of a report. The indicators must be defined on the File Description Specifications.

8. First Page Indicator: The 1P indicator enables headings to be printed on the first page of a report. This indicator is turned on only at the beginning of the processing before any input records have been read. The purpose of this indicator is to cause the printing of a cover sheet or page heading lines on the first sheet of a report.

If the user intends to print identical heading lines on each page, an overflow indicator (OF or OV) and the first page indicator (1P) must be specified in an OR-relationship because the overflow condition does not occur until after the first page has been printed.

If any of the output indicators must be off, enter N in columns 23, 26, or 29, whichever is appropriate.

At the beginning of executing an object program, the following indicators are always on:

- First page indicator 1P
- Level-zero indicator L0

Examples of Output Indicators

Figure 74 shows six examples of output indicators used with the output file. The numbers to the right of the figure refer to the item numbers in the following text.

1. The carriage is skipped to channel 02 before printing. 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 or record identifying indicators from the input specifications, or 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 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 program from printing a line if an error condition has occurred. Although the program does not stop until all processing for the record has been completed, printing of erroneous data is prevented.

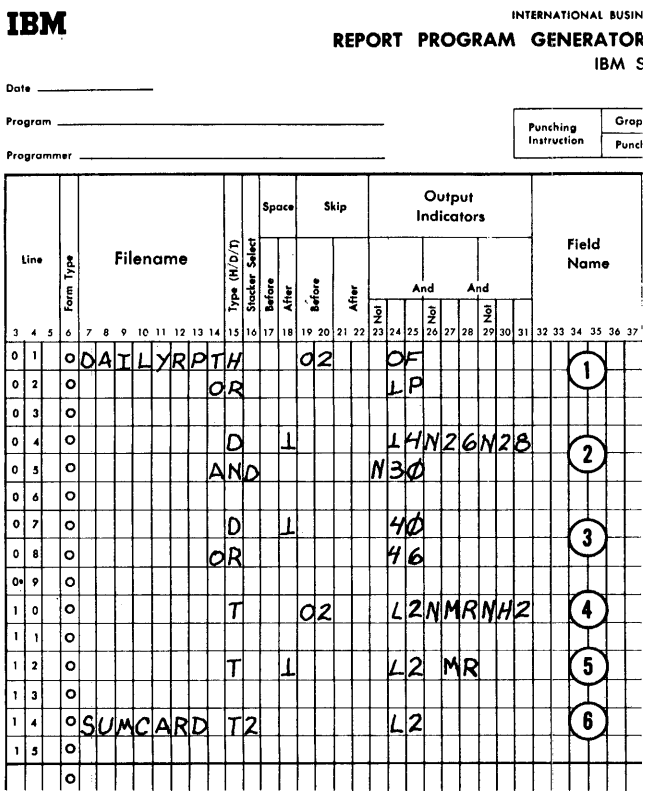


Figure 74. Example of Specifying Output Indicators for Output Lines

5. The summary record is printed at level 2 control time, but the MR (matching records) indicator must also be on.

6. A summary card is punched at level 2 control time, and it is selected into stacker 2.

FIELD DESCRIPTION

These entries include specifications for:

1. The control of the individual fields of a record.

2. The output format of individual fields of a record.

The fields of a 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.

In general, the rightmost position of a numeric field contains the sign. A 12-punch or a digit punch designates a positive value, an 11-punch a negative value. If a numeric field with a negative value is printed, the rightmost position in the output field will contain one of the letters J - R, or the special character -(Minus). This can be avoided by means of an appropriate edit word, or edit code in the output specifications.

Output Indicators (Columns 23-31)

The same types of indicators used for record identification can be used for field description. The maximum number of indicators that can be considered to be in an AND-relationship is three for each field description line.

Output indicators must not be specified in an OR-relationship on a field description line.

Figure 75 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:

IBM

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 06

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And/Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
010	O	PRINT	D		1				44							
020	O									INVOIC			16			
030	O									AMOUNT			26		(1)	
040	O									CUSTR			40			
050	O								L1	SALESM			59			
070	O	PRINT	D		1				24							
080	O									NAME			40			
090	O									DEPT			48		(2)	
100	O								N16	AMOUNT			59			
120	O	PRINT	H		1				04							
130	O	OR					03		OF							
140	O								NOF	ORDER			16			
150	O								NOF	DATE			26		(3)	
160	O								NOF	SALESM			39			
170	O									CUST			48			
180	O									INVOIC			61			
	O															
	O															

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 06

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And/Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
200	O	TOTAL	T		1				12 MR							
210	O								16 NH2 NL3	DIVSON			26			
220	O								26	AMOUNT			48		(4)	
230	O								28	AMOUNT			59			
	O															

Figure 75. Example of Specifying Output Indicators for Fields

1. Four fields are printed from a detail record identified by indicator 44: INVOIC, AMOUNT, CUSTR, and SALESM. The entry L1 causes the contents of the field SALESM to be printed only for the first detail record of each control group. Only the field named SALESM has group indication. (Note that control level indicators remain on during the following detail calculation and output cycle.)

2. The second example illustrates how to prevent the printing of just one field

of a record. The contents of the field AMOUNT is printed only if indicator 16 is turned off. (This indicator might be a resulting indicator used to determine if the calculated contents of the field AMOUNT are zero.)

3. This example shows selective printing of constant headings contained in cards for an invoice form. The 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 named ORDER, DATE, and SALESM when an overflow condition occurs. In this example, indicators OF and 04 are never on at the same time.

4. The last example shows how the printing of a field can be controlled by an AND-relationship and an OR-relationship. The printing of the field named DIVSON is controlled by three AND indicators: 16, NH2, and NL3. The field named 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. OR cannot be specified in columns 14-15 of a field-description line.

Field Name (Columns 32-37)

This specification identifies each field of the record to be printed. The fields may 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 the name of the field which is to become output in columns 32-37. The field name must have been previously defined on

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System 360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 04

Program Identification: RPT001

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	ADDRFILE	WS			33	1		C	1									1												
02	I																				2	50		PAGE							
03	I		01	1		01	1		C	1									4												
04	I																				2	80		CUST							
05	I																				9	28		NAME							
06	I																														
07	I																														
08	I																														
09	I		02	W		04	1		C	4									3												
10	I																				2	80		CUST							
11	I																				9	120		QUAND							
12	I																				48	530		PARTNO							
13	I																				69	740		ORDNO							
14	I																														
15	I																														

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System 360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 06

Program Identification: RPT001

Line	Form Type	Filename	Type (H/D/T)	Space: Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And Not	Output Indicators: And	Output Indicators: And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
010	O	REPORT	D	1					16									
020	O											PARTNO			8			
030	O											ORDNO			19			
040	O											ORDQTYZ			50			
050	O											SCRAPQZ		B	88			
060	O											RECPTQZ		B	108			
090	O	REPORT	H	1					OF									
100	O								L3			PAGE	Z		100			
130	O	CARDF	D	1														
160	O											AMT			40	P		

Figure 76. Example of Page Numbering

IBM

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 06

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
0 1	O	LOWERPRTD					02		20									
0 2	O	OR							OF									
0 3	O											PAGE1	Z		100			
0 4	O	UPPERPRTD					01		OV									
0 5	O											PAGE	Z		50			
0 6	O																	

Figure 77. Example of Page Numbering with Two Counters

	Negative Balance Indicator		
	None	CR	-
Print with commas, print zero balance	1	A	J
Print with commas suppress zero balance	2	B	K
Print without commas print zero balance	3	C	L
Print without commas suppress zero balance	4	D	M

Figure 78. Summary of Edit Codes

either the Input Specifications or Calculation Specifications form. If a constant is to be written, it is specified in Constant or Edit Word (columns 45-70), and Field Name is left blank.

Page or Record-Sequence Numbering

Automatic page numbering is another feature of RPG. By placing the word PAGE in columns 32-37 (Field Name) of the output specifications, automatic page numbering will be obtained on the printed report. The page entry in Field Name of Figure 76 (line 06100) causes each page of the output to be numbered consecutively in print-positions 97-100. The number is always four positions long. However, an edit code or edit word should be specified by the user. The page number is increased by one before it is printed.

Page numbering normally begins with the number 1, but page numbering can be started with any number by preparing a record type that contains the starting page, less 1.

In this case the record must be defined on the Input Specifications form with a field labeled PAGE. Figure 76 shows an example of this specification. In this example page numbering is defined on line 04010 and 04020 of the Input Specifications form. On the output form in Figure 76 (line 06100), it is defined as a separate field that will print in position 100. If the page numbering were to begin with the number 500, 0499 would be punched in columns 2-5 of the input record that contains the character - (minus) in position 1.

A page number can be reset to zero and a new series of page numbers may be started during the processing of the object program. When PAGE appears in the field name of the Output Specifications form, the output indicators are used to reset the page to number 1. In this case, the indicator must be placed in Output Indicators on the same line as the field name specification PAGE (see the L3 indicator in Figure 76).

Field length and digits	1769532	02	00	000	041345
Specified as	Positive number; two decimal positions	Negative number; two decimal positions	Zero; two decimal positions	Zero; no decimal positions	Positive number; three decimal positions
Edit Codes 1	17,695.32	.02	.00	0	41.345
2	17,695.32	.02			41.345
3	17695.32	.02	.00	0	41.345
4	17695.32	.02			41.345
A	17,695.32	.02CR	.00	0	41.345
B	17,695.32	.02CR			41.345
C	17695.32	.02CR	.00	0	41.345
D	17695.32	.02CR			41.345
J	17,695.32	.02-	.00	0	41.345
K	17,695.32	.02-			41.345
L	17695.32	.02-	.00	0	41.345
M	17695.32	.02-			41.345
Y	Must be used with a 3 to 6 digit field.				4/13/45
Z	1769532	2			41345

Figure 79. Example of Edit Code Usage

Any output indicators specified in a PAGE line are checked before printing. If the conditions of the indicators are satisfied, the page counter is reset to 1, instead of being incremented by 1.

Note: Any field name which begins with PAGE__ will be treated as a page numbering field. Therefore, the user could specify several page numbering fields for one or more files. A second page numbering field might be called PAGE1. Figure 77 illustrates the use of two page counters.

Edit Codes (Column 38)

Numeric fields can be automatically edited by entering an edit code in column 38 of the Output-Format Specification. This code can provide comma insertion, zero suppression, sign removal, negative indication with either - or CR, blanking of zero fields, date field editing, and insertion of the decimal point.

Zero suppression means that zeros to the left (leading zeros), of the significant digits, and the sign on the units position,do not appear in the output record.

Simple Edit Codes: The use of these codes causes no punctuation of amount fields.

X The positive sign is removed from the units position of a numeric field. No zero suppression occurs (this code has no effect upon 1130 RPG since the valid plus sign is a hexadecimal F. Therefore, all positive fields will be punched or printed with only a digit in the units position).

Y Date field is edited, as follows:

3 digits	(n)n/n
4 digits	(n)n/nn
5 digits	(n)n/nn/n
6 digits	(n)n/nn/nn

The first digit is zero suppressed.

Note: If inverted print is specified on the RPG Header Control Card, a decimal point (.) is printed instead of a slash (/).

Z Leading zeros are suppressed. Sign, if any, is removed. Decimal positions are not considered.

Combination Edit Codes: These codes cause punctuation of an amount field. If the field has decimal positions, the decimal point is inserted and printed (except when a zero balance is suppressed). When a zero balance is to print, the decimal positions print as zero, and positions to the left of the decimal point print as blanks. In the case of a numeric field without decimal positions, printing a zero balance results in only a zero in the units position of the field. When a zero balance is to be suppressed, the field prints as blanks. Leading zeros are always suppressed. Figure 78 summarizes these edit codes.

Note 1: When inverted print is specified (I in column 21 of the RPG Header Control Card), the use of comma (,) and period (.) in punctuating amount fields is reversed.

Note 2: When '*' is entered in columns 45-47, the asterisk fills the suppressed positions. Each leading zero is replaced by * when zero suppression is used.

Note 3: When '$' is entered in columns 45-47, the dollar sign appears immediately to the left of the high-order significant (non-zero) digit.

Note 4: This column (38) should be left blank when an edit word is used (columns 45-57).

Figure 79 illustrates the use of edit codes when printing numeric fields.

Blank After (Column 39)

If a B is entered in this column the output field will be reset to blanks or zeros immediately after execution of the transfer operation. Alphameric fields are set to blanks, and numeric fields are set to zeros.

Note: A specification in Blank After also affects any constant (columns 45-70) that may be contained in the line. 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 any following operation. If a field is tested for indicators in the plus, minus or zero columns (Input or Calculation Specifications), a blank after specification will not reset these indicators.

End Position in Output Record (Columns 40-43)

This specification indicates the exact 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 42-43 would be 30. Columns 40-41 are left blank, since leading zeros may be omitted.

When edit words or edit codes are being used, the end position must allow for the punctuation characters introduced by the editing process.

Packed Field (Column 44)

This field may be used only with disk files. Enter a P in this column if the output field is to appear in the packed decimal format. Otherwise, leave this column blank. If the output is a constant or is to be interpreted or printed, leave this column blank. The length of an output field in characters of packed format is

$$\frac{n+B}{2}$$

where n is the number of digits used, and B=1 if n is odd, B=2 if n is even.

Figure 76 shows examples of Field Name (columns 32-37), Edit Codes (column 38, Blank After (column 39), End Position in Output Record (columns 40-43), and Packed Field (column 44).

Constant or Edit Word (Columns 45-70)

Columns 45-70 are used to:

1. Include constants (literals) in output records.
2. Specify the edit word for proper output format.
3. Modify the function of the edit code (column 38). Permit editing of numeric fields.

Constants

An alphameric literal of up to 24 characters may be placed in columns 45-70. The literal must begin in column 45, and it must be enclosed in apostrophes. The literal 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 EBCDIC character set may be used in an alphameric literal. Blanks are treated as valid characters.
2. Alphameric literals must be enclosed in apostrophes. An apostrophe (') is represented in a literal by two consecutive apostrophes. For example, the literal 5 o'clock would be entered as '5 o''clock' in columns 45-56 of the Output-Format Specifications form.
3. Field Name (columns 32-37) must be left blank when an alphameric literal is defined on the line.

Edit Code Options

The characters '*' and '$' may be entered in columns 45-47 to modify the edit code entered in column 38.

'*' indicates that asterisk fill is used with zero suppression. All leading zeros are replaced by *.

'$' indicates that a floating dollar sign is to be used. The character

$ prints immediately left of the high-order significant digit.

Edit Words

For typical fields, the use of edit codes will provide proper editing. Edit words may be used where unusual editing requirements are needed.

An edit word provides for the punctuation of amount fields, including the printing of dollar signs, commas, periods, and sign status. Edit words can also be used to suppress leading zeros, and to provide for asterisk protection up to a specific position as indicated (see items 3 and 4 under the next section). A sample edit operation is shown in Figure 80.

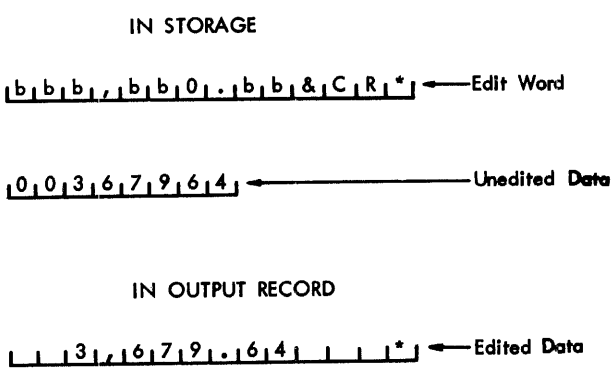


Figure 80. Functions of the Edit Operation

When an amount field is to be edited, its edit word is placed in columns 45-70 of the same output specification line where the field to be edited is specified.

An edit word consists of three parts (the body, the status, and the expansion) as shown in Figure 81.

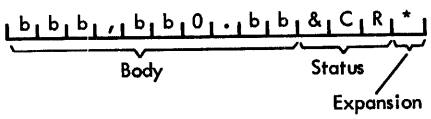


Figure 81. Body, Status, and Expansion of an Edit Word

The body of an edit word governs the transfer of the data field to the output record. The body portion begins at the leftmost character of the edit word and contains the same number of blanks (one zero or an asterisk) as the number of digits of the data field.

The function of the edit-word status position is to display the status (positive or negative) of the data field. It is the portion continuing to the right from the body to the CR (credit) or - (minus) symbol. Edit words that contain no CR or - symbol have no status portion.

The expansion portion of an edit word remains unchanged. It is the portion continuing to the right from the status portion (or body portion if there is no status portion) and ending with the rightmost character of the edit word.

Rules for Forming an Edit Word

1. An edit word must be enclosed in apostrophes.
2. A blank in the body portion of the edit word is replaced with the character from the corresponding position of the data field specified in Field Name.
3. An ampersand in the body or status portion causes a blank in the edited field. It remains unchanged in the expansion portion.
4. A zero is used for zero-suppression. It is placed in the rightmost position where zero suppression is to stop. It is replaced with the character from the corresponding position of the data field, unless that character is a zero. Column 38 (edit codes) must be left blank.
5. If leading zeros are desired, the edit word must contain one more position than the field to be edited, and a zero must be placed in the high order position of the edit word.
6. An asterisk in the body of the edit word is used for asterisk protection and zero suppression. It is placed in the rightmost position where zero suppression is to stop. It is replaced with the non-zero character from the corresponding position of the data field. Each suppressed zero is replaced by an asterisk. An asterisk preceding a CR symbol, a minus symbol, or a zero, is interpreted as representing asterisk protection.
7. A dollar sign in the body of the edit word written immediately to the left of the zero-suppression code causes the insertion of a dollar sign in the position to the left of the first significant digit. This is the floating dollar sign. A dollar sign that is

entered immediately after the initial single-quote mark is fixed (printed in the same location each time). This is the fixed dollar sign.

8. The decimal and commas are printed in the same relative positions they were written in the edit word. If they are to the left of significant digits, they are blanked out or replaced by an asterisk.

9. 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. If asterisk protection is used, they are replaced by an asterisk.

10. 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, or if the value of the field is minus zero, CR and - are blanked out.

11. Characters to the right of the status portion of the edit word are undisturbed.

12. The edit word may be larger than the field to be edited.

Figure 82 illustrates the use of constants and edit words. The numbers to the left refer to the item numbers in the following text.

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 must be blank.

3. This example illustrates zero suppression to the left of significant digits. The letters CR are written because the amount field can contain a negative value.

4. In this example, the floating dollar-sign protection enters the $ to the left of the first significant digit.

5. 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.

Sterling (Columns 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. If the sterling specification is not required, leave this column blank.

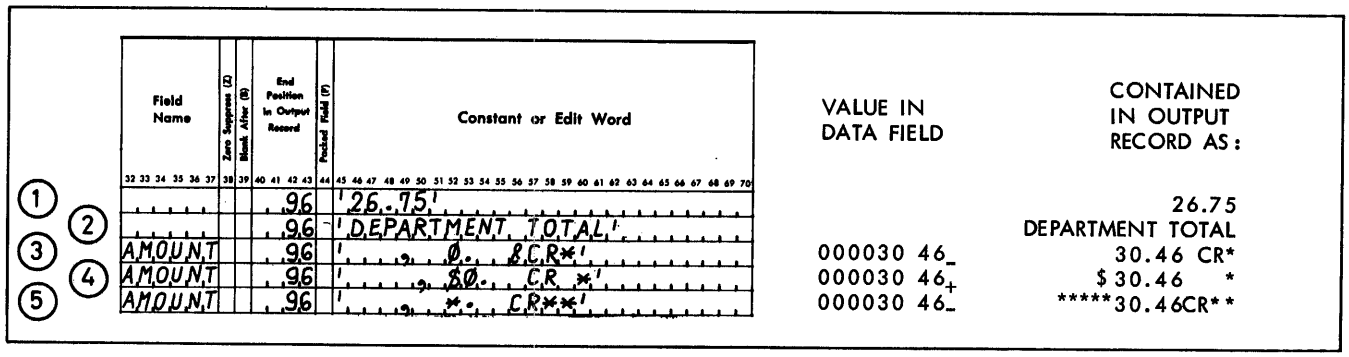


Figure 82. Example of Using Constants and Edit Words

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

1. The input files, specified on the Input Specifications form, from which the object program obtains data records.
2. Input table files, and RA files.
3. The output files, specified on the Output-Format Specifications form on which the object program writes data records.

The maximum number of files that will be supported by the RPG object program is 10. Figure 83 defines the maximum number of files for the various types of files that can be used. Any combination of these is permitted, up to a total of 10. A table file is defined as one line on the extension specifications. It is possible to have eight table files loaded, each of which contains an alternating entry. Therefore, up to 16 different table names are possible.

Type of File	Max. No. of Files
Primary	1
Secondary with/without matching fields	8
Record Address File	1
Chained	9
C1-C3	3
Chain Operation	9
Table	8
Output	
Printer	2
Other Output Units	9

Figure 83. Maximum Number of Files

File Name (Columns 7-14)

Each file used in the program is identified by writing the name of the file in columns 7-14. Filenames must be alphameric and left-justified (the filename must begin in column 7). The filename entered in these columns must also be entered on the Input Specifications form (for input data files), on the Output-Format Specifications form (for output data files), or on the Extension Specifications form (for tables, RA files, chaining files).

Note: The filename may be up to eight characters long, but RPG recognizes only the first five characters of a file name. Therefore, it is important that the first five characters constitute a unique file name. Special characters or embedded blanks may not be used.

File Type (Column 15)

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

Entry	Type of File	Explanation
I	Input	Identifies the file as an input file (it may be a table file, RA file, or a file containing input data records which are read only).
O	Output	Identifies the file as an output file. It may be an output data file or an updated table file to be printed, punched, or written.
U	Update	Identifies the file as an update file (DASD only). An update file is both an input and an output file. It must be specified similarly to an input file on the Input Specifications form. 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 without changing: 1. The nature of the data. 2. The length of the field in which the data is found. 3. The location of the field. A chained file may be updated at detail time or at total time. All other disk files can be updated only at detail time.

C Combined Identifies the file as a combined card file.

A combined file is a card file containing cards that are read into the system and cards used for punching. Reading and then punching into the same file is accomplished in two different ways:

1. Punching into cards that contain input data.

2. Punching into a blank trailer card in the same file.

For those cards of a combined file that are read only, Stacker Select entries should be specified on the Input Specifications form. This will reduce the time required to run the object program. For combined-file cards that are punched, Stacker Select entries must not be specified on the Input Specifications form.

A combined file differs from an update file in that the update file input fields are revised or updated. A combined file can only refer to cards, and an update file can only refer to disk.

Figure 84 illustrates two examples of File Name and File Type specifications.

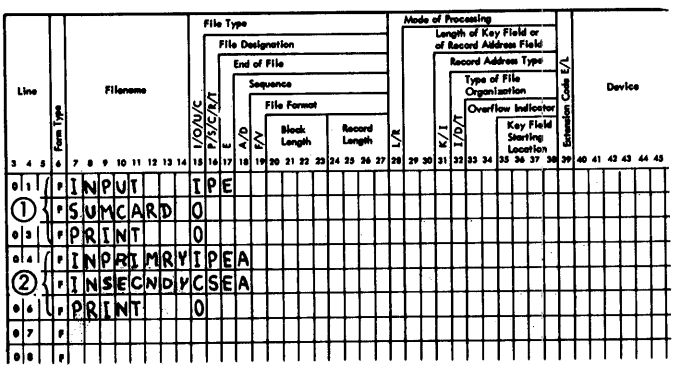


Figure 84. Example of Specifying File Name and File Type

1. Detail cards are read into the system in one file. Summary cards are punched into a separate file which, in this example, contains blank cards only. The printed report is a third file.

2. There are two input files in this example. The second file (INSECNDY) also contains cards that will be used for punching output data; therefore it is specified with file type C.

File Designation (Column 16)

Column 16 is used to designate the type of input file being defined on this line of the File Description form. The five entries permitted in this column are:

Entry Explanation

P The file defined is a primary file. Only one primary file may be defined. A primary file is an input file.

S The file is a secondary file.

T The file defined is an input table file.

R The file defined is an RA file which relates to a direct access storage file. A record address file is specified in the File Description and Extension Specifications forms. Entries for RA files are not permitted in the input and output specifications.

C The file defined is a chained file, as defined under Chaining . Records and fields for chained files are specified similarly to those for primary and secondary files on the Input Specifications form. The following exceptions should be noted: matching field specifications, control field specifications, and numeric sequence records are not permitted for chained files. A C should be entered for either method of chaining (CHAIN Operation or C1-C3).

If the file is an output file identified by O in column 15, leave this column blank; an entry must be made if the file is an input, update, or combined file.

Detailed explanations of chained files, table files, RA files, primary files, and secondary files may be found in the sections Using Tables in the Object Program and Processing Multiple Input Files.

The priority of multiple secondary files is determined by the sequence in which they are recorded on the File Description form. For further information refer to the section Order of Processing Records Using the Matching Technique.

End of File (Column 17)

Enter an E in column 17 if the input file is a primary, secondary, or an RA file and the programmer wishes to determine an end of job after the end-of-file condition for this file has been reached. For multiple input files, the end-of-job condition (LR) will occur when all input files for which

an E is specified in column 17 have been processed.

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

Note 1: An E should not be entered in column 17 if the file is processed in random mode or if the file is either an output or a table file.

Note 2: If a matching records job has an E on the primary file and no E's on the secondary files, any matched secondaries will be processed before the end of job occurs.

Sequence (Column 18)

This entry is made if the matching-fields specification (columns 61-62 of the Input Specifications form) 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.

If the matching field specification is not used, this column is left blank.

File Format (Column 19)

Enter an F in this column. Records may only be fixed-length.

Block Length and Record Length (Columns 20-27)

Enter right-justified the length of one record in Record Length (columns 24-27). Leave Block Length (columns 20-23) blank. Disk files will be automatically blocked so the maximum number of records will fit on a disk sector. Card and Printer Files cannot be blocked.

Maximum record lengths are as follows:

2501 and 1442 I/O units	80 characters
1132, 1403, or console printers	120 characters
sequential or direct disk files	640 characters
indexed-sequential disk files	636 characters

Note: If the record length entry is missing, a record length of 80 is assumed.

Columns 28-38 must be left blank for card- or printer-file specifications.

Mode of File Processing (Column 28)

This column is used to indicate the method or mode by which disk files are processed. Acceptable entries are listed here.

Entry	Explanation
Blank	Leave this column blank if the disk file is to be processed sequentially or created. If an indexed sequentially organized file is to be created, column 15 of the File Description Specifications form must contain O, column 32 must contain an I, and column 66 must be left blank. If an indexed sequential file is to be added to, this column must be blank.
R	Enter an R in this column if the user's records are to be processed randomly. In this case, the records of an indexed sequentially organized file to be processed will be obtained by the CHAIN operation (or C1-C3) using key fields. The records of a sequentially organized disk file will be processed randomly by using the CHAIN operation (or RA file) and relative record numbers.
L	Enter an L if a segment of an indexed sequential file is to be processed using limits specified by an RA file. Only an indexed sequential file can be processed within limits.

Length of Key Field or of Record Address Field (Columns 29-30)

If the file defined on this line is a Record Address file (RA file), enter the number of positions that each entry in the RA file occupies.

Enter the length of the key, if the file defined on this line is a file with an indexed-sequential organization (I in column 32). The maximum key length is 50 characters.

Type of Record Address (Column 31)

K	Enter a K in this column if the file is an indexed sequentially organized file. The K indicates that the file defined on this line will be processed by use of the record key.
blank	For sequential file

Type of File Organization (Column 32)

I Enter an I in this column if the file to be created or processed has an indexed sequential organization. (There must also be a K in column 31 and 1 in column 38).

blank Leave this column blank if the file is organized sequentially.

2 Enter the digit 2 to signify two I/O areas to be assigned to a card reader, punch, or printer file. Stacker select (Input Specification, column 42) should not be specified. (Any entry 1-9 may be used but the digit 2 is preferred.)

RULES FOR SPECIFYING MODE OF PROCESSING, TYPE OF RECORD ADDRESS, 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. For sequential file organization, columns 28, 31, 32 are also left blank.

3. For indexed sequential file organization (I in column 32), column 31 must contain a K and column 28 can contain either R, L, or blank.

4. For direct file organization (D in column 32), column 31 must contain an N.

Figure 85 illustrates the code combinations possible for these three specifications.

Overflow Indicator (Columns 33-34)

If the file defined on the line is a 1132 or 1403 printer file and overflow indicators are used, enter the overflow indicator associated with the file. The permissible indicators are OF and OV. Do not specify an overflow indicator for the console printer.

Key Field Starting Location (Columns 35-38)

Enter a 1 in column 38 if the file specified has indexed sequential organization. With 1130 indexed sequential organization, the key field always begins in position 1.

Type of File Organization (Col. 32)	Type of Record Addresses (Col. 31)	Mode of Processing (Col. 28)
Indexed-Sequential *(I)	Record Key (K)	The entire file will be processed. (blank)
		A segment of the file will be processed. The limits to be processed are supplied by a Record Address File (RAF). (L)
		The records will be processed randomly. (R)
		The keys are supplied: (a) by Factor 1 of the CHAIN Operation (b) by a chaining field (C1-C3) or (c) by a record address file (RAF). The records will be processed randomly - the record numbers are supplied by Factor 1 of the CHAIN operation or by a RA file. (R)
Sequential	Not applicable (blank)	The entire file will be processed. (blank)
Direct (D)	Record Number (N)	The records will be processed randomly. (R)

Figure 85. Processing Direct Access Storage Files

The key field can be either an alphameric or numeric field. It cannot be packed.

Extension Code (Column 39)

If the file defined on the line is a chaining file (using C1-C3 codes), RA file, or a table file, enter an E in column 39. The E specifies that additional information about the file will be coded on the Extension Specifications form.

Device (Columns 40-46)

These columns are used to indicate the input/output units used by each file. The following entries can be used:

Entry	Input/Output Unit
PRINTER	IBM 1403 printer
PRINT32	IBM 1132 printer
PUNCH42	IBM 1442 punch
READ42	IBM 1442 reader/punch
READ01	IBM 2501 reader
CONSOLE	Console printer
DISK	IBM 2310 disk

Symbolic Device (Columns 47-52)

These columns are used in 1130 RPG only when the file described is indexed sequential and is to be loaded. Enter left-justified the maximum number of records in the file. This number may not exceed 99999. A message will be printed during compilation which describes how many disk sectors are needed for a file of this size.

Columns 53-65

Columns 53 through 65 are not used by 1130 RPG and must be blank.

File Addition (Column 66)

Enter an A in this column if the file defined on this line is an indexed sequential organized file and new records are to be added. The actual record or records to be added are described on the Output Specifications with an entry of the characters ADD in Columns 16-18.

For all other file description specifications, this column must be left blank.

Figure 86 illustrates the code combination for columns 15 and 66 of the File Description Specifications form if the file defined is an indexed sequential organized file (I in column 32).

Column 15	Column 66	Mode of Processing
O	blank	Creating an IS file (load).
O	A	Adding records to an IS file.
I	blank	Processing of an IS file without updating and without additions of new records.
I	A	Processing of an IS file and adding new records.
U	blank	Processing and updating records of an IS file.
U	A	Processing and updating records of an IS file and adding new records to the file.

Figure 86. Summary of Code Combinations for an Indexed Sequential File

Columns 67-74

Columns 67 through 74 are not used. Leave blank.

ENTRIES ON THE FILE DESCRIPTION SPECIFICATIONS FORM

Figure 87 shows several examples of entries on the File Description Specifications form. The numbers to the right correspond to the explanation that follows.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7 8 9 10 11 12 13 14	File Type: I/O/U/C 15	File Designation: P/S/C/R/T 16	End of File: E 17	Sequence: A/D 18	File Format: F/V 19	Block Length 20 21 22 23	Record Length 24 25 26 27	L/R 28	Mode of Processing: Length of Key Field or of Record Address Field 29 30	Record Address Type: K/I 31	Type of File Organization: I/D/T 32	Overflow Indicator 33 34	Key Field Starting Location 35 36 37 38	Extension Code E/L 39	Device 40 41 42 43 44 45 46	Symbolic Device 47 48 49 50 51 52	Labels (S, N, or E) 53	Name of Label Exit 54 55 56 57 58 59	Extent Exit for DAM 60 61 62 63 64 65	A 66	File Addition: No. Tracks for Cylinder Overflows 67	No. of Extents 68 69	Tape Rewind: N/U 70	71 72 73 74
0 1 Ø	F	INPUT	I	P	E	A	F		8Ø								READ42			1						
0 2 Ø	F	RAFFILE	I	R			F		8Ø		8						EREADØL				2					
0 3 Ø	F	TABFILE	I	T			F		1ØØ								EDISK			3						
0 4 Ø	F	MASTCUST	I	C			F		1ØØ	R	8	K	I			1	DISK				4					
0 5 Ø	F	DISKUPDT	U	C			F		2ØØ	R	6	K	I			1	DISK			5						
0 6 Ø	F	CARDREC	C	S		A	F		8Ø								READ42				6					
0 7 Ø	F	OUTPUT	O				F		12Ø					OF			PRINTER			7						
0 8 Ø	F	ISAM	O				F		75		6	K	I			1	DISK	5ØØØ			8					
0 9 Ø	F	ISAM	O				F		75		6	K	I			1	DISK			9		A				
1 0 Ø	F	SEQDISK	I	P			F		6Ø								DISK				10					
1 1 Ø	F	SEQDISK	U	C			F		6Ø	R							DISK			11						
1 2 Ø	F	ISAMLIMT	I	S		A	F		13Ø	L	3	K	I			1	DISK				12					
1 3	F																									
1 4	F																									

Figure 87. Example of Using the File Description Specifications Form

1. The P in column 16 indicates that the input file INPUT is the primary file. The E in column 17 indicates that the end-of-job condition will occur when the end of the file is reached. The file is ascending (A in column 18). Each record is 80 characters long. This file is read in on an IBM 1442. The device code is READ42.

2. The RA file defined on this line has a fixed format (column 19). Each record is 80 characters long and the length of each record address field is 8. The E in column 39 indicates that the Extension Specifications form will be used. The RA file is read into the program by an IBM 2501 Card Reader. The device code is READ01.

3. The third file defined on this form is a table file indicated by a T in cclumn 16. It has a fixed format and a record length of 100. Column 39 (E) indicates that it will be further described on the Extension Specifications form. The file is read in from magnetic disk; therefore, the Device code is DISK.

4. MASTCUST is an input file that will be processed under the control of chaining. It is a chained file (C in column 16) that will be processed randomly (R in column 28). The record addresses that will be referred to by the chaining file are keys (K in column 31), and the file has an indexed-sequential organization (I in column 32). This file is located on a direct access storage device and the Device code is DISK.

5. The update file DISKUPDT (U in column 15) will be used for input, and it will be updated after the 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. The record length is 200. This file is located on a direct access storage device and the Device code is DISK. The key field starting location (columns 37-38) must be position 1 within the data record.

6. The file CARDREC is a combined file (column 15). The file will be used as input, and additional information will

be punched in the input cards during processing. It is a secondary file (column 16). This combined file is read in and punched out on an IBM 1442. The specified device code is READ42.

7. The file OUTPUT is to be a printed report on the 1403. The length of the record is 120 characters. The printed report has a Device code of PRINTER. The overflow indicator OF has been assigned to this printer.

8. The file ISAM is an indexed sequential file (I in column 32) with a record length of 75 characters. The file is to be initially loaded (O in column 15). The key field is six characters long (columns 29-30) and begins in position 1 (columns 35-38). The system should allocate space for 5000 records to be loaded in this file (columns 47-52). RPG will automatically print the number of sectors that will be required for this file. The index will be organized using the first six positions of the output record as the key field.

9. The same file that was created in line 08 is being added to in line 09. The entries are identical except that the number of records to be allowed is not specified. There is an A in column 66 to denote the add function. The letters ADD are required on the Output Format Specifications (columns 16-18) for the record to be added. When adding records, no retrieval or updating is allowed on the file during the addition operation.

10. The file SEQDISK is a sequentially organized disk file (blank in column 32) of 60 characters. No sequence (blank in column 18) is specified since it is the only input file for the job and no sequence checking is to be done.

11. The same sequential file described in line 10 is now being processed randomly (R in column 28) as a chained file (C in column 16). A relative record number will be used to randomly retrieve records from this file.

12. The file ISAMLIMT is an indexed sequential organized file being processed by limits. It will be treated as an input file (I in column 15) and is a secondary file (S in column 16) being matched against a primary file. The match fields are in ascending sequence (A in column 18). The records are 130 characters long (130 in columns 24-27). The file is limited (L in column 28) by an RA file which will supply the limits. The key is three bytes long (3 in columns 29-30) and must begin in position 1 (1 in columns 35-38). The entries of K-I (columns 31-32) are required for an ISAM file and an ISAM file can only reside on disk (DISK in columns 40-46).

Entries made on the Extension Specifications form provide information to RPG about chaining files (except when processed with the CHAIN operation), tables used in the object program, and RA files. In the sections Using Tables in the Object Program and Processing Multiple Input Files, detailed information and examples show how to use these functions.

RECORD SEQUENCE (COLUMNS 7-8)

These columns are used when chaining by the alternate method described in the section Input Specifications Form.

If the file defined on this line is a chaining file, enter the sequence of the file from columns 15-16 of the Input specification sheet. The file name of the chaining file is taken from columns 7-14 of the input specification sheet.

Leave these columns blank if the file described is a RA file.

NUMBER OF THE CHAINING FIELD (COLUMNS 9-10)

These columns are used when chaining by the alternate method described in the section Input Specifications Form. Leave these columns blank if the file is not a chaining file.

Enter the identifying number of the chaining field (C1-C3) in columns 9-10. This number must be the one entered in columns 61-62 (Chaining Fields) of the Input Specifications form.

FROM FILE NAME (COLUMNS 11-18)

Enter in these columns the name of the chaining file, the RA file, or the table file.

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 may provide the name of an input table file and the name of the table file that is to be output at end of job. Both From Filename and To Filename are taken from Filename (columns 7-14) of the related entry on the File Description form.

Figure 88 illustrates the entries for these two specifications.

TO FILE NAME (COLUMNS 27-32)

Refer to Figure 88.

Type of File	* From File Name (11-18)	* To File Name (19-26)
Chaining Files	Name of Chaining File. 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 Specifications form.	Name of Chaining File. 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 specifications form for the chained file.
Record Address File	The name of the RA 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 is being defined (columns 27-57) enter the name of the file that contains the table data.	If the table being defined will be printed, punched, or written after it is updated, enter the name of the table output file. If it is not being printed or written, leave the field blank.

Figure 88. From and To File Names

TABLE NAME (COLUMNS 27-32)

Columns 27-32 contain the name of either the table that contains arguments or the table that contains functions. If the appropriate table records contain entries of only one table, the name of the table is entered in these columns. A table name must consist of 4, 5, or 6 characters. The first three must be TAB; the remaining characters must be alphameric. The entry must be left-justified.

NUMBER OF TABLE ENTRIES PER RECORD (COLUMNS 33-35)

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

NUMBER OF TABLE ENTRIES PER TABLE (COLUMNS 36-39)

In these columns, enter the number of table entries contained in the table. This entry must be right-justified.

LENGTH OF TABLE ENTRY (COLUMNS 40-42)

Enter in columns 40-42 the length of each table entry. The maximum lengths are 248 alphameric characters or 14 numeric digits. For a packed field enter the number of digits of the field. The entry must be right-justified.

PACKED (COLUMN 43)

If the data in the table is in the packed-decimal format, enter P in this column. Otherwise, leave this cclumn blank. (Packed data may only be from disk files.)

DECIMAL POSITIONS (COLUMN 44)

This column is used only if the entries contain numeric data. Enter a zero if there are no decimal positions. Enter the number of decimal positions (1-9) in this column if the data contains decimal positions. If the field is alphameric, leave this column blank. The number of decimal positions cannot exceed the length of the table entry defined in cclumns 40-42.

SEQUENCE (ASCENDING OR DESCENDING, COLUMN 45)

A If the data contained in the table is in ascending sequence, enter an A in this column.

D If the data contained in the table is in descending sequence, enter a D in this column.

blank 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.

The table must be defined as either ascending or descending if it is used in a LOKUP operation as Factor 2 (argument table) and high or low indicators are used.

COLUMNS 46-57 (SECOND TABLE)

Columns 46-57 are used if a table file consists of alternating arguments and functions.

TABLE NAME (COLUMNS 46-51)

If two table names are used, enter the second table name in these columns. The entry must be left-justified.

LENGTH OF TABLE ENTRY (COLUMNS 52-54)

Enter in these columns the length of each table entry. The maximum length of a table entry is 248 alphameric characters or 14 numeric digits. For a packed field enter the number of digits of the field. The entry must be right-justified.

PACKED (COLUMN 55)

Enter a P if the data in the table is in the packed-decimal format. Otherwise, leave this column blank. (Packed data can only be from disk files.)

NUMERIC DECIMAL POSITIONS (COLUMN 56)

If the data contained in the table is numeric, enter a zero if there are no decimal positions. Enter the number of decimal positions (1-9) in this column if the data contains decimal positions. If the data field is alphameric, leave this column blank. The number of decimal postions cannot exceed the length of the table entry defined in columns 52-54.

SEQUENCE (ASCENDING OR DESCENDING, COLUMN 57)

A If the data contained in the table is in ascending sequence, enter an A in this column.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348 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	Number of the Chaining Field	From Filename	To Filename	Table Name	Number of Table Entries Per Record	Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed(P)	Decimal Pos.	Sequence (A/D)	Comments
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
01	E	AA	C1	INPUT	MASTCUST													①
02	E			RAFFILE	DISKUPDT													②
03	E			TABFIL		TABARG	10	150	10			A	TABFUN	10				③
04	E			OLDTAB	NEWTAB	TABREC	5	10	12									④
05	E																	

Figure 89. Sample Entries on the Extension Specifications Form

D If the data contained in the table is in descending sequence, enter a D in this column.

blank 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.

The table must be defined as either ascending or descending if it is used in a LOKUP operation as Factor 2 (argument table) and high or low indicators are used. If the table is used in the Result Field of a LOKUP operation it does not have to be specified as ascending or descending.

COMMENTS (COLUMNS 58-74)

Columns 58-74 may contain comments.

ENTRIES ON THE EXTENSION SPECIFICATIONS FORM

Figure 89 shows four examples of Extension Specifications form entries. The numbers at the right of the example correspond to the explanation that follows.

1. In this example, the alternate method of chaining is described. 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, (i.e., it is the file that links or chains to another file). The field contained in INPUT, which is used to link the two files, is the chaining field. The AA entry in columns 7-8 denotes the record sequence entry of the chaining file record. C1 is the number of the chaining field. Thus, INPUT is chained to MASTCUST by using a field defined on the Input Specifications form that contains C1 in columns 61-62. A complete discussion of chaining may be found in the section Processing Multiple Input Files.

2. In this example, RAFFILE is an RA file that supplies the record number of the records to be processed in the file DISKUPDT. The DISKUPDT file in this example is a direct access organized file. The same entries are made if it is an indexed sequential file. In this case the RA file would have to supply the actual keys to be processed or the limits of the file to be processed.

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 named 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 (column 44). The A in column 45 denotes the table to be in ascending sequence. The table lookup operation

can then specify whether it should be an equal, high, or low search.

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. The table is organized in 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 or punched on an output unit.

OLDTAB is the name of the input table file. NEWTAB is the name given to the output table file.

The arguments in the file are contained in the table named TABREC. The argument table is described in columns 33-45. Five table entries are in each record. The number of table entries in the table is 10, and each table entry is 12 characters long. The data is numeric. TABREC is the name used on the Calculation Specifications form when specifying a table look-up or update operation.

A table is an arrangement of data that is searched and used by the object program. Tables are loaded into storage by the RPG object program before any files are processed.

A table may consist of two parts: arguments and functions. In Figure 90 the table consists of part numbers (arguments) and prices (functions). The card file in Figure 90 contains part numbers that have been ordered. The cards do not contain the prices of the parts. The part 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. 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.

Entries in a table may be (Figure 91):

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

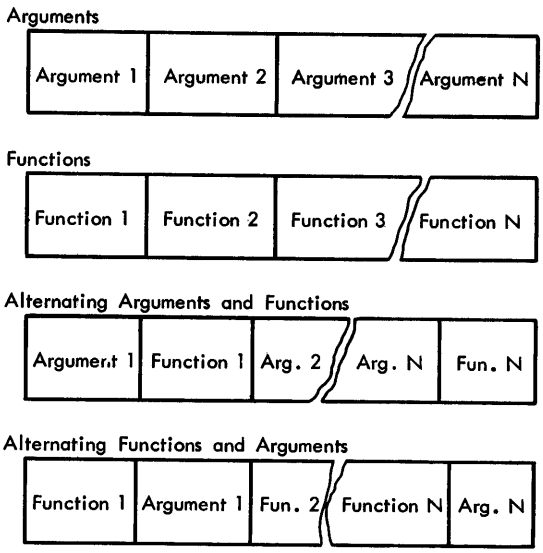


Figure 91. The Four Types of Tables

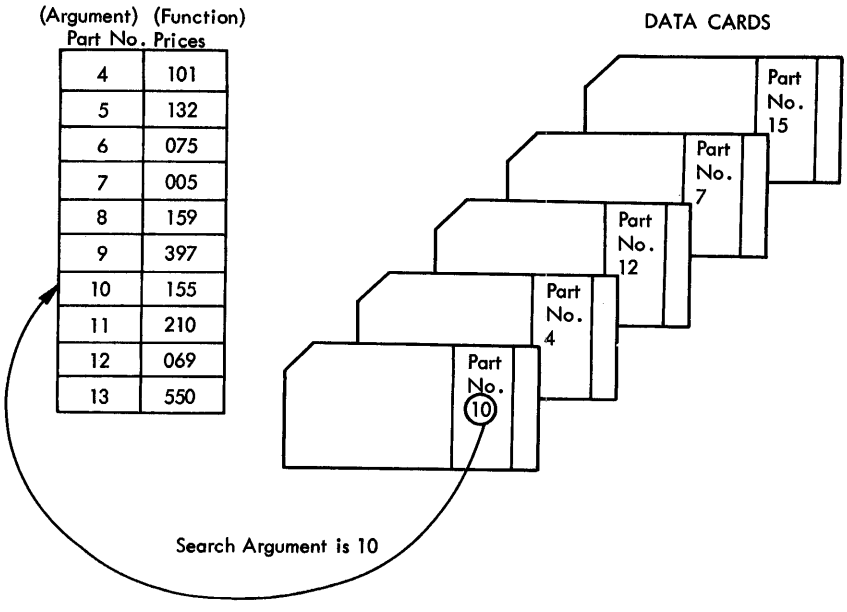


Figure 90. Example of Using a Table

RULES FOR FORMING TABLES

1. For tables consisting of arguments only, or functions only, each unit of table data is called a table entry. For tables consisting of alternating arguments and functions, one argument and one function are called one table entry.

2. The collection of all arguments is assigned a name. The collection of all function entries is assigned a name. The table name may consist of 4, 5, or 6 characters. The first three must be TAB, the rest may be any alphameric characters. In Figure 95 the alternating table file called RATETABL is split into a collection of arguments (TABNUM) and functions (TABRAT). RATETABL is the name of the file containing the tables.

3. Although all of the tables may be loaded from the same device, a unique file name must be assigned in columns 7-14 for each line entry on the File Description Specifications form and repeated in columns 11-18 of the Extension Specifications form. The tables will be loaded into storage before the object program is processed, and each line entry on the Extension Specifications form must have:
   a. A unique filename (columns 11-18)
   b. Entries in columns 27-45 if the table contains only arguments or functions.
   c. Entries in columns 27-45 and 46-57 if the table consists of alternating arguments and functions.

Rules for Creating Records Containing Table Data

1. The first table entry for each record must begin in column 1.

2. All records must have the same number of table entries, except the last (see Figure 92).

3. All entries must be continuous in every record. In Figure 92, the first entry begins in position 1 and the second entry begins in position 10. No blanks may be contained between the table entries.

4. All function or argument entries belonging to a table must have the same length. In Figure 92, each argument is three positions long, and each function is six positions long.

5. When alternating tables are used, each record must begin with an entry of the same type, i.e., each record must

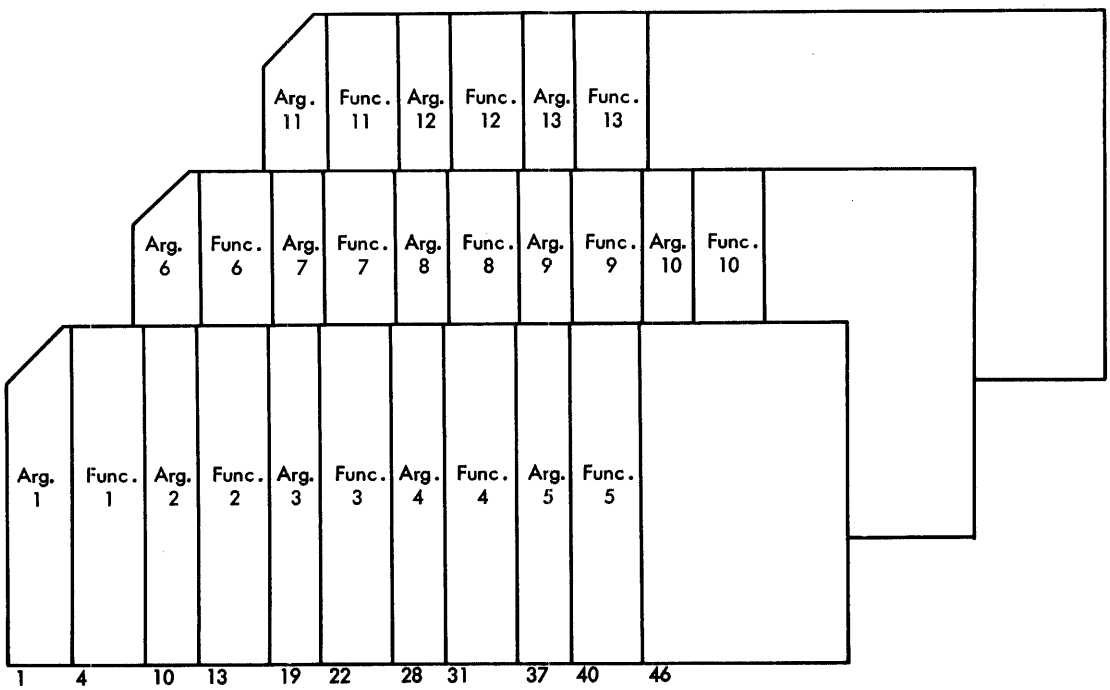


Figure 92. Sample Table File Containing Arguments and Functions

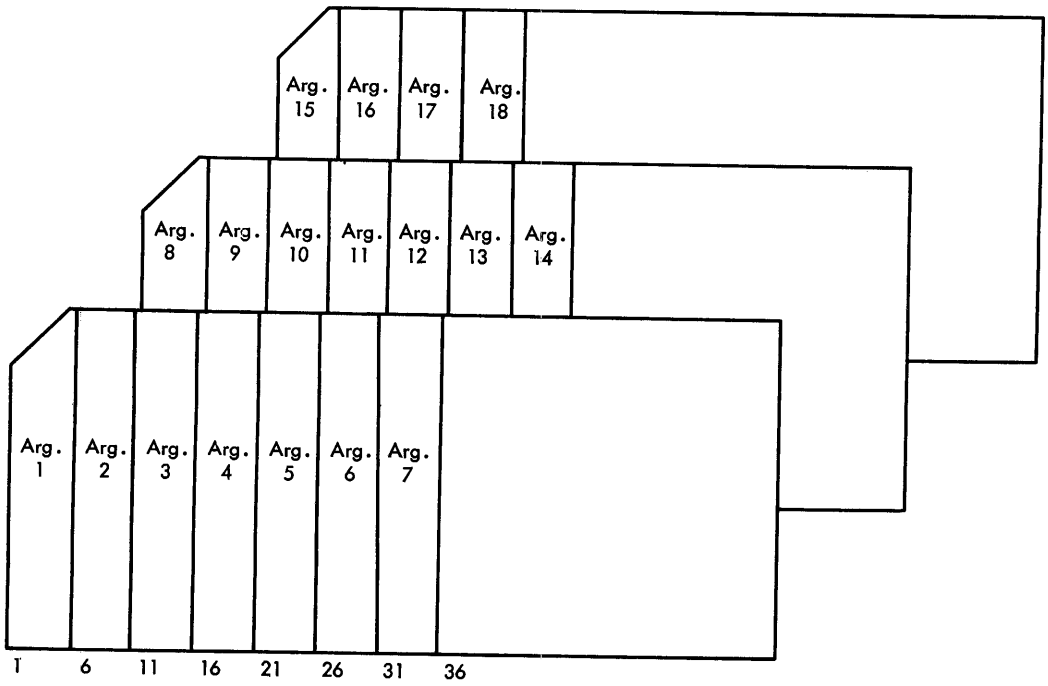


Figure 93. Sample Table File Containing Arguments Only

always begin with an argument, or each record must always begin with a function as shown in Figure 92.

6. When alternating tables are used, the table entries in each record must not be split. Function 3, for example, must be in the same record as argument 3. It is not permissible for a function to appear in a different record from its corresponding argument.

7. If a table consists of all arguments or all functions, an argument or a function must not be split. Assume that argument one, argument two, argument three, and argument four are contained in the first record. No part of argument four could overflow into the second record. Figure 93 illustrates the correct way to specify records containing arguments or functions.

8. The table may be ascending, descending, or in no sequence. If the LOKUP operation is to be performed on this table, and high or low indicators are specified, the table must be either ascending or descending. If an alternating table is used, normally the argument table will be in ascending sequence (A) and the function table will have no sequence (blank).

9. The maximum length of alphameric entries is 248 characters. Numeric entries must not exceed 14 digits in length.

10. The records of a table must be on a sequentially organized file.

11. The table file to be loaded must contain the exact number of table entries as specified on the Extension Specifications form.

12. The record format for a table must be fixed-length. However, there may be more than one table entry per record.

13. When tables are read from or written onto disk, each table is regarded as one file.

14. If multiple tables are loaded from a card reader, the same device name is used on the File Description Specifications (however, the filenames must differ). The sequence of loading tables is based upon the sequence assigned on the Extension Specifications.

15. 1130 RPG has a limit of eight table files. This means that no more than eight lines for tables may be coded on the Extension Specifications. If each

table has an alternating table, it is possible to have 16 different table names.

METHODS OF PROCESSING TABLES

The operation code LOKUP entered on the Calculation Specifications form 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. Factor 2 contains the name of the table which contains the arguments.

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. Decimal alignment will be performed.

The result field contains the name of the table from which an associated function is to be located, if arguments and functions are used. The result field may be left blank if the user wants to determine if an argument is present in the table, but a corresponding function is not required.

Resulting indicators (columns 54-59) must always have an entry when the table lookup operation is performed. The indicators indicate the type of lookup to be performed, and the indicators are turned on whenever the condition is satisfied. The program may search for the table argument next higher than the search argument (resulting indicator in columns 54-55) or it may search for the table argument next lower than the search argument (resulting indicator in columns 56-57) or it may search for the table argument equal to the search argument (resulting indicator in columns 58-59). An entry must be made somewhere in columns 54-59. A high-equal or low-equal search may be specified by placing indicators in the appropriate two of the three fields (columns 54-59). It is not possible to specify indicators in both the high and low columns.

Note: Indicators must not be placed in High or Low if the table is not in ascending or descending sequence.

The compare operations are logical for alphameric arguments and algebraic for numeric arguments. The search arguments must have the same format as the table entries compared with them, i.e., they must be numeric for numeric table entries, etc. Decimal alignment is performed if numeric search arguments and table entries have differing decimal lengths.

The lookup operation is performed in this way:

1. The object program 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 the 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 the result field) is located. Then, argument and function are placed in special holding areas for the function and argument tables. If the proper table argument is not found, the indicators in columns 54-59 are not turned on.

Other operations may be performed using the data just found by the table-lookup operation which is stored in special holding areas for the function and argument tables. It can be retrieved by using the name of the function table in either Factor 1 or Factor 2 of an operation.

Updating Tables

In addition to using a function obtained by means of a LOKUP operation, calculations can be performed to change the argument or the function in the appropriate holding area as well as the table. This can be achieved by entering the defined name of the function or argument table in the Result Field and by entering an arithmetic operation code (such as ADD, SUB, MULT, DIV) or a move operation code (such as MOVE, MOVEL, MHHZO, etc) in Operation. This procedure changes the contents of the function or argument holding area and the table. If a Blank After specification is entered on a table name, both the holding area and the specific table entry will be blanked or zeroed out.

The RPG program returns the updated contents of the appropriate holding area to the location in core storage from where it was retrieved by the LOKUP operation. Updating of data in a table can be performed only on an argument or the related function obtained by means of the last preceding LOKUP operation that refers to this table. The programmer must ensure that an entry was found that satisfies the table search condition (specified by the resulting indicators) before updating the table. If no entry is found and updating occurs, the last found entry will be updated.

Figure 94 illustrates several ways in which the data found by the operation can be used. The numbers on the figure refer to this discussion.

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 06

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And	Indicators: And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High (1 > 2)	Compare: Low (1 < 2)	Compare: Equal (1 = 2)	Comments
01	C	1			PERCNT	LOKUP	TABCST	TABAMT									15	
02	2		15		TABAMT	LOKUP	TABARG	TABFUN									10	
03	C	3	10			MOVE	TABFUN	WITHTX	5	2								
04	4				+25	LOKUP	TABFIL	TABLIT									20	
05	C		20			MOVE	+500	TABLIT										
06	C		20			MOVE	+30	TABFIL										
07	C	5			SEARCH	LOKUP	TABNUM										30	
08	C		N30			SETON								H1				
09	6		01		000	LOKUP	TABARG	TABFUN									35	
10	C		35	01		Z-ADD	NEWARG	TABARG										
11	C		35	01		MOVE	NEWFUN	TABFUN										
12	C	7																
13	C																	

Figure 94. Example Using the LOKUP Operation Code

1. Factor 1 contains the name of a field. The field PERCNT contains the search argument. The name of the table that contains the arguments is TABCST. The name of the table that contains the corresponding function is TABAMT. The program will search for the value in the argument table that is equal to the search argument because of the entry in columns 58-59.

2. The function found in TABAMT from the previous operation is used as the search argument in this example. (TABAMT is the name of the table; however, Factor 1 is the function stored in the special holding area of the table TABAMT.) The program searches for the value in the table TABARG that is equal (columns 58-59) to the search argument.

3. The data obtained from the function table TABFUN from the previous lookup operation is moved to a field called WITHTX. It will be used for additional calculations.

4. The data found in the function and argument tables is updated. The literal +25 is the search argument. The table TABFIL is searched for +25 (indicated by the entry in columns 58-59). A new entry for the corresponding function of +25 is entered in TABLIT. The new function is +500; the new argument is +30. When updating a table, both the special holding area and the table entry are changed.

5. In example 5, a lookup with only one argument table turns on indicator 30 if SEARCH is equal to an argument in TABNUM. If 30 is not on (N30), H1 is turned on by the SETON operation.

6. This example illustrates how entries are added to the table. The LOKUP operation is conditioned by indicator 01. (Indicator 01 is turned on when the input file contains records with additional table information. Each record contains the two fields, NEWARG and NEWFUN.) To determine the first vacant argument, a field of zeros is used as the search argument. Zeros are used if the argument field is numeric, whereas blanks are used if the argument field is alphabetic. However, these zeros or blanks must be loaded as part of the table at the beginning of the job. If there is an equal compare, indicator 35 is turned on. Because the argument field of the table is vacant, the corresponding function field is also vacant. The new argument (NEWARG)

is inserted in the TABARG field, and the function is inserted in the TABFUN field.

Figure 94.1 shows the actual contents of four tables. The Extension Specifications for entering them into a program are shown in Figure 94.2. TABLEC and TABLED are entered in an alternating format.

The Calculation Specifications show various LOKUP possibilities with various indicator settings. (See Figure 94.3)

Figure 94.4 then describes the results of each of the LOKUP statements. The indicator is only set when the search is satisfied in the table. If the search is nct satisfied, the table holding areas will contain the value of the last satisfied search.

	First Entry	Second Entry	Third Entry	Fourth Entry	Fifth Entry
Table A	01	05	08	32	96
Table B	05.13	02.12	47.15	28.70	15.16
Table C	WWW	NNN	LLL	GGG	AAA
Table D	7	8	3	2	5

Figure 94.1 Content of Four Tables

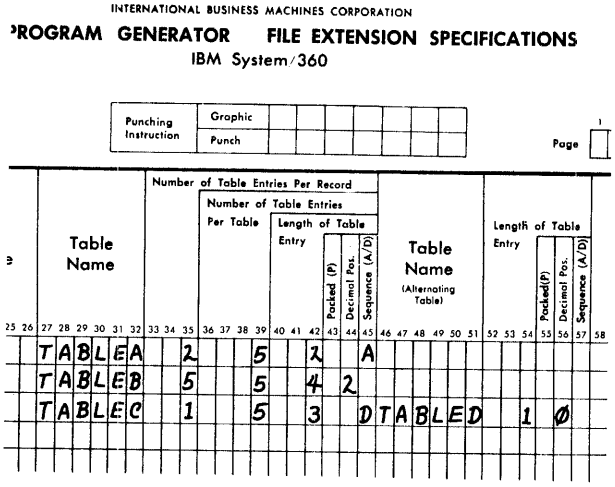


Figure 94.2 Extension Specifications for Table Entries

RETRIEVING TABLES

After a table has been updated, the table may be written or punched out for later use. On the File Description Specifications form, the programmer enters the name of the file that will ccntain the updated table. The file must be defined as a sequentially organized output file. On the Extension Specifications form, the programmer enters the name of the file on which the updated table will be written under To Filename. The name of the table is entered in columns 27-32. If two tables are to be put out, enter the name of the second table in columns 46-51. The updated table files will be put onto the output file after the program has reached the end-of-job condition (LR condition). This output file must have the same format and size as the input table file. No editing is possible for printing of tables. If editing is necessary, two approaches are possible (see Example of Retrieving Tables):

1. The table is written onto disk and then read back in and edited in a second job.
2. The table is put out at LR time using the EXCPT operation and a loop on the Calculation Specifications.

EXAMPLE OF USING TABLES

Figures 95 and 96 illustrate an input data file, the way a table might appear, and the entries necessary on the RPG specification forms for a program that uses tables. 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 employee's rate has been determined, the rate is multiplied by the number of hours worked by the employee. The result of this operation is the amount earned for each employee.

IBM

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 (L0-L9, LR)	Indicators: And / And / Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Minus	Zero or Blank	Compare: High 1>2	Low 1<2	Equal 1=2	Comments
01	C			'08'	LOKUPTABLEA		TABLEB							01			
02	C			'08'	LOKUPTABLEA		TABLEB								02		
03	C			'08'	LOKUPTABLEA		TABLEB									03	
04	C			'08'	LOKUPTABLEA									04		05	
05	C			'08'	LOKUPTABLEA										06	07	
06	C			'00'	LOKUPTABLEA										08		
07	C			'97'	LOKUPTABLEA									09			
08	C			'09'	LOKUPTABLEA		TABLEC							10		11	
09	C			'09'	LOKUPTABLEA		TABLEC								12	13	
10	C			'LLL'	LOKUPTABLEC		TABLED							14			
11	C			'JJJ'	LOKUPTABLEC										15		
12	C			'JJJ'	LOKUPTABLEC											16	
13	C			'JJJ'	LOKUPTABLEC		TABLED							17			
14	C																
15	C																

Figure 94.3 Table Lookup (LOKUP Operations)

Specification Line Number	Entry Found	Indicator On	Table Item Satisfying search Ccndition	Table Item Used From Related Table
01	yes	01	32	28.70
02	yes	02	05	02.12
03	yes	03	08	47.15
04	yes	05	08	
05	yes	06	05	
06	no			
07	no			
08	yes	10	32	GGG
09	yes	12	08	LLL
10	yes	14	NNN	8
11	yes	15	GGG	
12	no			
13	yes	17	LLL	3

Figure 94.4 Table Lookup (Results of LOKUP Operations)

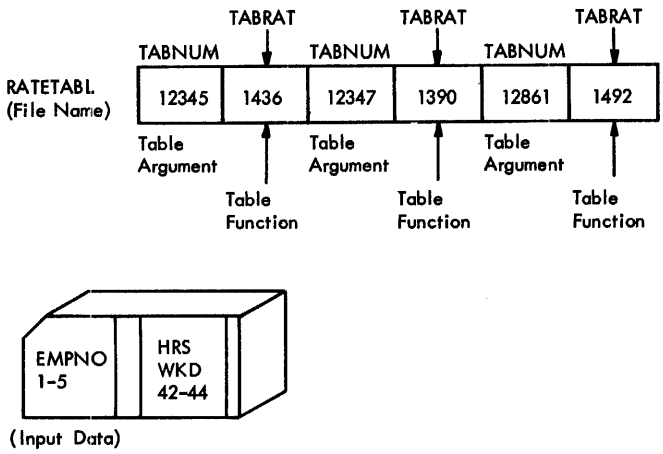


Figure 95. Example of Using Alternating Arguments and Functions

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 01

Program Identification: RPT

Line	Form Type	Filename	File Type	File Designation	End of File	Sequence	File Format	Block Length	Record Length	Mode of Processing	Length of Key Field or of Record Address Field	Record Address Type	Type of File Organization	Overflow Indicator	Key Field Starting Location	Extension Code (E/A)	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	File Addition	No. Tracks for Cylinder Overflow	No. of Extents	Tape Rewind
			I/O/U/C	P/S/C/R/T	E	A/D	F/V			L/R		K/I	I/O/T												N/U
01	P	TIMECARDIPE					F		80								READ01								
02	P	RATETABLIT					F		80								READ42								

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 02

Program Identification: RPT

Line	Form Type	Record Sequence of the Chaining File	Number of the Chaining Field	From Filename	To Filename	Table Name	Number of Table Entries Per Record	Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Comments
01	E			RATETABL		TABNUM	8	500	5		0	A	TABRAT	4		3		

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 03

Program Identification: RPT

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators Plus	Field Indicators Minus	Field Indicators Zero or Blank	Sterling Sign Position
01	I	TIMECARD	AA			01																									
02	I																				1	5	0	EMPNUM							
03	I																				42	44	1	HRSWKD							

IBM

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 04

Program Identification: RPT

Line	Form Type	Control Level (L0,L1-L9)	Indicators And Not	And Not	And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus	Minus	Zero or Blank	Compare High 1>2	Low 1<2	Equal 1=2	Comments
01	C					EMPNUM	LOKUP	TABNUM	TABRAT									03	
02	C		03			TABRAT	MULT	HRSWKD	EARNS	5	2	H							
03	C		N03				MOVE	+000.00	EARNS										

Figure 96. Coding Forms for Table Example

In this example, the table consists of alternating arguments and functions. The way the table data might appear is shown in Figure 95. 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 forms follow.

File Description Specifications Form

The two files are defined on the File Description Specifications form. The file containing the input card records is called TIMECARD. It is an input file (I in column 15) and a primary file (P in column 16). When this file is depleted, processing is terminated (E in column 17). Each record is 80 characters long (80 in columns 26-27). This file is read in on an IBM 2501 Card Reader.

The table file is defined on the line below the card-input file. The name of the file (RATETABL) is entered in Filename (columns 7-14). It is an input table file (I in column 15, T in column 16). The file has a record length of 80 characters. The E in column 39 indicates that additional information about the file is coded on the Extension Specifications form. This file is read in on the IBM 1442 Card Reader.

Extension Specifications Form

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 eight arguments per record (columns 34-35) and 500 entries 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).

The collection of functions is described in 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).

Input Specifications Form

The input file (TIMECARD) is described on the Input Specifications form in columns 7-14. The file is assigned a sequence of AA (columns 15-16), and record identifying indicator 01 is turned on whenever an input record is present for processing. No record identification codes are specified because record identification codes need not be entered if there is only one record type.

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, 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 are found in columns 42-44 of the input record, as specified by the entries in field location. The name HRSWKD is assigned to the field containing the number of hours worked by each employee.

Calculation Specifications Form

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 columns 28-32 causes the lookup operation to be performed. Factor 2 contains the name of the collection of arguments (TABNUM) which is searched by the search argument. The result field contains the name of the collection of functions (TABRAT). Thus, this operation causes the employee number (EMPNUM) to be used as the search argument for the data contained in TABNUM. The 03 entered in columns 58-59 indicates that indicator 03 will be turned on when the program 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 4 positions long, 3 decimal positions) that has been located is multiplied by the number of hours worked (HRSWKD 3 positions long, 1 decimal position) and the result is stored (EARNS 5 positions long, 2 decimal positions). The result 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 EARNS, specifying that the employee does not have an entry in the table.

EXAMPLE OF RETRIEVING TABLES

Occasionally, it is necessary to update a table during a program and to output the table once (e.g., at LR time) or several times (e.g., at L1 time). If output is required only at LR time, it can be accomplished by letting RPG put out the table and specifying a second program to print the results.

If the table must be printed and formatted during the update program, a solution is possible with RPG.

The following example shows a table being put out during a processing run. The same technique can be used at LR time. A card file shows salesman records to be processed and totals accumulated for the items sold. The cards are in sequence by salesman but the items within each salesman are not in sequence. Consequently, a table-updating program has been chosen. The same job could be accomplished by sorting the cards by item within salesman and using control levels (Figure 96.1).

The File Description entries show two table files on disk and a card and printer file. The Extension Specifications shows TABL1 as being an alternating table containing TABCOD (the code numbers of the items sold) and TABAMT (initially loaded as zeros and will be updated during the program). There are 20 entries in TABL1 (one for each of the items).

TABL2 also has twenty entries and is made up of consecutive numbers 01 through 20. This table will be used to assist in the output of the other two tables at the end of each salesman. The table entries are shown in Figure 96.1.

The Input Specifications describe the item card and provide a level 1 control break on salesman number.

The Calculation Specifications show three detail calculations. Line 01 does a lookup of the field CODE of the item card against TABCOD to find the same item in the table. The result field of TABAMT denotes that the corresponding amount is also made available.

Indicator 21 is set on if the search produces an equal entry. Line 02 sets on H1 if the entry is not found. Line 03 adds the field AMOUNT from the input card to the table TABAMT if indicator 21 is on.

This process continues until all of the salesman's cards have been read and a level 1 control break occurs. At this point, TABAMT contains the correct amounts and needs to be put out.

The total calculations start at line 06 where indicator 30 is set on. This will be used to produce a heading line prior to the output of the table. Line 07 sets up the field COUNT to be equal to 1 so it can be used to search for the first entry in TABCOD.

Line 09 does a lookup of TABNUM and produces the first entry of TABCOD (031) since the first entry in TABNUM is 01. Line 10 moves the first entry in TABCOD to a saved area. Line 11 takes the saved area and looks for the same entry in TABCOD. The result field of TABAMT produces the corresponding entry.

The programmer now has the data needed to print the first summary line. Line 12 adds the amount in TABAMT to a total field. Line 13 makes a comparison to determine if the field COUNT has reached 20. When it reaches 20, the last item in the table (472) is ready to be printed.

Line 14 causes a branch to exception output. Line 15 sets off indicator 30 to prevent repeated printing of the heading line.

Lines 16 and 17 will be performed if the field COUNT has not yet reached 20. The branch to LOOP will start the process over again. The second time through the field COUNT will be equal to 2. This will cause the second item number (032) to be retrieved from the table and its amount to be printed.

The Output Specifications show a typical detail output of the input card in lines 01-04. The SLSMAN field is printed on the first line following a L1 control break.

Lines 05-08 show the heading line which is to be printed just prior to printing the first item. Indicator 30 was set on to allow this and then set off after the first exception output was called for.

Lines 09-11 show the output required for each item. TABCOD and TABAMT are printed to provide the proper information. TABAMT is blanked after (zeroed out) to allow the proper total to be accumulated for the next salesman.

Lines 12-14 provide for the total amount of each salesman to be printed and a double asterisk to denote it. The field TOTAL is also blanked after to allow for the proper accumulation for the next salesman.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	I/O/U/C 15	P/S/C/R/T 16	E 17	A/D 18	F/V 19	Block Length 20–23	Record Length 24–27	L/R 28	K/I 31	I/D/T 32	Key Field Starting Location 35–38	Extension Code E/L 39	Device 40–46	Symbolic Device 47–52	Labels (S, N, or E) 53	Name of Label Exit 54–59	Extent Exit for DAM 60–65	A 66	No. Tracks for Cylinder Overflows 67–68	No. of Extents 69	Tape Rewind N/U 70
0 1	F	CARD	I	P			F		80						READ42								
0 2	F	TABL1	I	T			F		9					E	DISK								
0 3	F	TABL2	I	T			F		2					E	DISK								
0 4	F	PRINTER	O				F		120						PRINTER								
0 5	F																						

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System 360

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

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 02

Program Identification RPT

Line 3 4 5	Form Type 6	Record Sequence of the Chaining File 7–8	Number of the Chaining Field 9–10	From Filename 11–18	To Filename 19–26	Table Name 27–32	Number of Table Entries Per Record 33–35	Number of Table Entries Per Table 36–39	Length of Table Entry 40–42	Packed (P) 43	Decimal Pos. 44	Sequence (A/D) 45	Table Name (Alternating Table) 46–51	Length of Table Entry 52–54	Packed (P) 55	Decimal Pos. 56	Sequence (A/D) 57	Comments 58–74
0 1	E			TABL1		TABCOD	1	20	3				TABAMT		6	2		
0 2	E			TABL2		TABNUM	1	20	2		0							
0 3	E																	

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 03

Program Identification RPT

Line 3 4 5	Form Type 6	Filename 7–14	Sequence 15–16	Number (1-N) 17	Option (O) 18	Resulting Indicator 19–20	Record Identification Codes 1: Position 21–24	1: Not (N) 25	1: C/Z/D 26	1: Character 27	2: Position 28–31	2: Not (N) 32	2: C/Z/D 33	2: Character 34	3: Position 35–38	3: Not (N) 39	3: C/Z/D 40	3: Character 41	Stacker Select 42	Packed (P) 43	Field Location From 44–47	Field Location To 48–51	Decimal Positions 52	Field Name 53–58	Control Level (L1-L9) 59–60	Matching Fields or Chaining Fields 61–62	Field-Record Relation 63–64	Field Indicators Plus 65–66	Minus 67–68	Zero or Blank 69–70	Sterling Sign Position 71–74
0 1	0	CARD	A	A		01																									
0 2	0																				1	2		SLSMANL	1						
0 3	0																				3	5	0	CODEHM							
0 4																					6	9	2	AMOUNT							
0 5																															

Figure 96.1. Retrieving an Updated Table (Part 1 of 2)

IBM

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 04

Program Identification: RPT

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And / And	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank	Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C			CODE	LOKUPTABCOD		TABAMT					21	
02	C		N21		SETON						H1		
03	C		21	AMOUNT	ADD	TABAMT	TABAMT						
04	C				}								
05	C				}								
06	C	L1			SETON						30		
07	C	L1			Z-ADD1		COUNT	20					
08	C	L1		LOOP	TAG								
09	C	L1		COUNT	LOKUPTABNUM		TABCOD					22	
10	C	L1			MOVE	TABCOD	SAVE	3					
11	C	L1		SAVE	LOKUPTABCOD		TABAMT					23	
12	C	L1		TABAMT	ADD	TOTAL	TOTAL	82					
13	C	L1		20	COMP	COUNT						24	
14	C	L1			EXCPT								
15	C	L1			SETOF						30		
16	C	L1	N24	1	ADD	COUNT	COUNT						
17	C	L1	N24		GOTO	LOOP							
	C												

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

IBM System/360

Form X24
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And / And	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	PRINTER	D		1				01NH1							
02	O									LISLSMAN			2			
03	O									CODE			10			
04	O									AMOUNTA			25			
05	O		E		22				30							
06	O												20		'SALESMAN'	
07	O									SLSMAN			23			
08	O												30		'TOTALS'	
09	O		E		1				L1							
10	O									TABCOD			10			
11	O									TABAMTAB			25			
12	O		E		11				L1 24							
13	O									TOTAL AB			25			
14	O												28		'**'	

Figure 96.1. Retrieving an Updated Table (Part 2 of 2)

An RPG program may use subroutines to avoid repetitious coding of common or frequently used procedures these subroutines are of two types:

1. External: written in the Assembler language. This type of subroutine must be assembled and stored in the System Subroutine Library prior to its use. This subroutine is invoked from an RPG program with the operation EXIT, and data references are made with the operation RLABL.

2. Internal: written in the RPG language. This type of subroutine is generated and executed as a part of the RPG program. It is defined with the operations BEGSR and ENDSR, and invcked with the operation EXSR. Since the subroutine is a part of the RPG program, all field names defined in the main program are available to the subroutine and vice-versa.

EXTERNAL SUBROUTINES

The EXIT operation code on the Calculation Specifications form causes RPG to transfer control from the RPG object program to a subroutine that has been coded in the Assembler language. A subroutine might be a standard routine (such as a state withholding tax routine) or it might be a routine that performs a function not easily accomplished using RPG (such as a square-root computation). The subroutine, written in Assembler language, is coded by the user. Entries made on the Calculation Specifications form enable the programmer to:

1. Exit from the RPG program to a subroutine.

2. Execute the subroutine.

3. Return to the main program after the subroutine has been executed.

The use of the EXIT operation and the understanding of this section requires thorough familiarity with the 1130 Assembler language. Information on this subject is contained in the SRL publication IBM 1130 Assembler Language, Form C26-5927.

A control-level entry and indicators (columns 7 through 17 of the Calculation Specifications form) can be used with the EXIT operation. When this is done, the EXIT operation will be executed only if the conditions established by the control-level entry and/or indicators are satisfied. An EXIT operation whose execution depends on a control-level entry or on indicators is called a conditional EXIT operation. (An EXIT can also be used within an internal RPG subroutine)

EXIT Format: The format of the EXIT operation is shown in Figure 97.

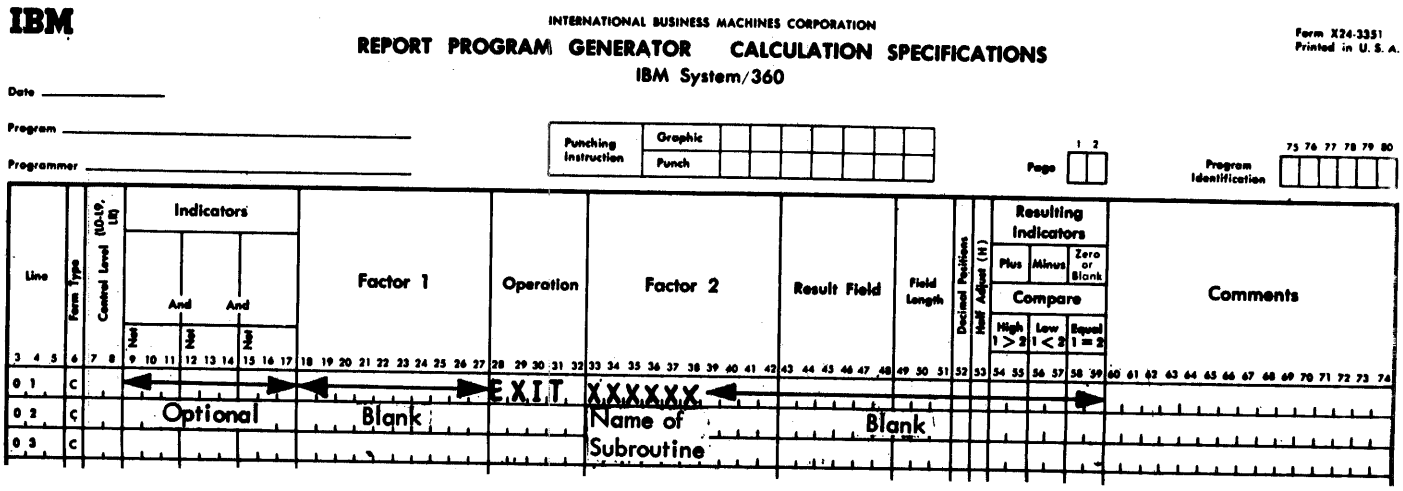


Figure 97. Format of the EXIT Operation

Columns 7 through 17 are used cnly with conditional EXIT operations. The mnemonic EXIT must be entered in columns 28-31. The name of the subroutine to which control is to be transferred must be specified in columns 33-38. The name of a subroutine is restricted to five characters. The first character must be alphabetic; the other characters may be alphabetic or numeric. Special characters or embedded blanks must not be used. The subrcutine must be present in the System Subroutine Library prior to execution of the RPG program.

Position: The EXIT operation may be used anywhere in the Calculation Specifications. However, the user should be aware of the following considerations regarding four specific positions of the EXIT operation in the program:

1. When the EXIT operation is the first detail calculation cf the program, control will be transferred upon completion of the input routine, i.e., as soon as the pertinent input data are available for processing.

2. When the EXIT operation is the last detail calculation cf the prcgram, control will be transferred to the designated subroutine immediately before heading and detail records are printed or punched.

3. When the EXIT operation is the first total calculation of the program, the exit to the subroutine will be made after the record type has been determined and the contrcl field has been tested.

4. When the EXIT operation is the last total calculation, control will be transferred immediately before the totals are printed or punched.

Field Definiticn: Any fields that are used in both the subroutine and the RPG program must be defined in the RPG program. This is achieved by RLABL statements. The use of the RLABL statement is described as follows..

1. If the field is nct used as a result field elsewhere in the program, or if it has not been defined in the Input Specifications form, it must be defined on the Calculaticn Specificaticns form (Figure 98, item 1).

   The symbol RLABL must be placed in columns 28-32. The name of the field is placed in the Result Field (columns 43-48). The length of a field name used in an RLABL statement must not exceed 6 characters, and the first character must be alphabetic. Field

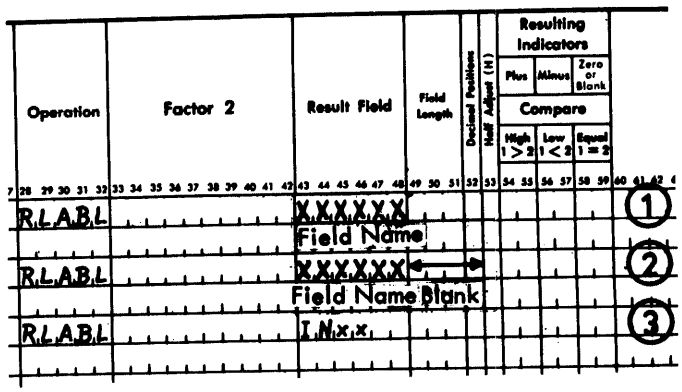


Figure 98. Format of the RLABL Operation

   length (mmm) and number of decimal positions (n) are entered in the appropriate columns.

2. If an RPG field that is to be used in a subroutine has been defined in the Input Specifications form or in the result field of another Calculation Specification form the symbol RLABL must be placed in Operation and the name of the field in Result Field. Field Length and Decimal Positions should be left blank (Figure 98, item 2).

A field whose name has been defined by means of a TAG specification must not be used in a subroutine. Consequently, the name of such a field cannot be used in an RLABL specification. RLABL specifications must immediately follow the EXIT specification with which they are associated.

Indicator Definitiqn. The user may want to test RPG indicators in a subroutine. In this case, each indicator required must be defined on the Calculation Specifications form (Figure 98, item 3).

The entry in columns 43-46 of the Result Field consists of the letters IN followed by the symbol for the indicator required. If, for example, the condition of the Matching Record indicator is to be tested in the subroutine, the entry in the Result Field portion of the appropriate RLABL statement is INMR. This entry indicates to the RPG program that MR will be tested in a subroutine. In the subroutine, the indicator may be referred to as the data located at INxx.

Note: The hexadecimal representation of indicator conditions is 0001 for ON or 0000 for OFF.

If, in the subroutine, the user sets on, sets off, or tests resulting indica-

tors, he must observe the following rules when he codes his subrcutine:

1. To set on a resulting indicator, set the data located in INxx to hexadecimal 0001.

2. To set off a resulting indicator, set the data located at INxx to hexadecimal 0000.

3. To test resulting indicators:
   a. If on, the data at INxx will be hexadecimal 0001.
   b. If off, the data at INxx will be hexadecimal 0000.

RLABL statements must be specified immediately following the EXIT statement which refers to the subroutine in which the desired fields are to be referenced.

If the same routine must be exited to at various points within the RPG source statements, it is desirable to place the EXIT and RLABL statements in an RPG internal subroutine.

The EXIT statement will cause an assembler CALL statement to be generated. Each RLABL statement will cause the compiler to generate the address of the referenced field or indicator.

Coding of Subroutines. All subroutines that are to be incorporated into RPG programs by means of the EXIT operation must be coded in Assembler language. A subroutine that is to be linked by an RPG program may not contain input/output operations. The following points must be observed:

1. RPG passes control to a subroutine via the assembler CALL statement. The Core Load Builder changes the CALL to a BSI indirect to the subroutines. Refer to IBM 1130 Assembler Language, Form C26-5927.

2. The CALL statement will place the address following the CALL statement in the first word at the subroutine entry point. This address will point to a list of RLABL referenced field addresses. The subroutine can reference the desired field by using the supplied address.

3. The subroutine must return to the address following the CALL statement plus the number of RLABL statements. For example, if there are 3 RLABL statements following an EXIT statement in the RPG program, the user's subroutine must return to the address formed by adding 3 to the contents of the first word of that subroutine.

4. Index register 3 is used by the system to pass control to subroutines. If the subroutine uses this register, it must save and restore its contents.

All subroutines to be incorporated in an RPG program should be coded on an IBM 1130 Assembler coding form. The subroutine source deck must be assembled separately using 1130 Assembler. The resulting object program deck must be stored in the System Subroutine Library.

Since the subroutines and the RPG program are to be combined, the subroutines must be relocatable and all Assembler linking conventions must be observed.

Restrictions. The following restrictions must be observed when using subroutines written in Assembler language.

1. Control cannot be transferred from one subroutine to another.

2. Subroutines should not contain input/output operations. This may interfere with the RPG routines.

3. RPG data formats are described in Types of Data Records.

Figure 99 illustrates the EXIT and RLABL operations. The RPG calculation specifications show a EXIT to the subroutine PROC which will reference indicator 01 and field AFLD. The 1130 Assembler Coding form shows one method of referring to the specified RPG indicator and field.

IBM

IBM 1130 Assembler
Coding Form

Form X26-5994
Printed in U.S.A.

Program _____ Date _____

Programmed by _____ Page No. _____ of _____

Label (21–25)	Operation (27–30)	F (32)	T (33)	Operands & Remarks (35–70)	Identification (75–80)
	ENT			PROC1	
PROC1	DC			*,* ADDR OF 1ST RLABL	
	STX		1	SAVE+1	
	LDX	I	1	PROC1 GET ADDR OF THE INDICATOR	
	.				
	MDX	L	1	PROC1,,+1 INCREASE RLABL PARM ADDR	
	.				
	LDX	I	1	PROC1 GET ADDR OF THE FIELD	
	.				
	MDX	L		PROC1,,+2 POINT TO RETURN ADDR	
SAVE	LDX		1	*,-*	
	BSC	I		PROC1 RETURN TO RPG	
	END				

IBM

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 (75–80)

Line (3–5)	Form Type (6)	Control Level (L0,L1,...LR) (7–8)	Indicators: And / Not (9–16)	Factor 1 (18–27)	Operation (28–32)	Factor 2 (33–42)	Result Field (43–48)	Field Length (49–50)	Decimal Positions (51)	Half Adjust (H) (52)	Resulting Indicators: Plus / Minus / Zero or Blank (53–55)	Resulting Indicators: Compare: High 1>2 / Low 1<2 / Equal 1=2 (56–59)	Comments (60–74)
01	C				*								
02	C		Ø1		EXIT PROC1								
03	C				RLABL		INØ1						
04	C				RLABL		AFLD						
05	C				*								
06	C												
07	C												

Figure 99. External Subroutine

INTERNAL SUBROUTINES

Subroutines may be coded in RPG. Such subroutines are a part of the RPG program which invokes them, and control does not leave the RPG object program. The RPG (internal) subroutine is coded on the Calculation Specifications form using EXSR, BEGSR, and ENDSR operations. The format of an RPG subroutine is shown in Figure 100. An RPG subroutine is delimited by the BEGSR and ENDSR operations. The BEGSR statement defines the beginning of the subroutine and the entry point into the subroutine. The name entered into Factor 1 is the name of the subroutine, and is used to invoke this subroutine.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

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 (L0-L9, LR)	Indicators: Not 9	And 10 11	Not 12	And 13 14	Not 15	16 17	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field (43-48)	Field Length (49-50)	Decimal Positions (51)	Half Adjust (H) (52)	Resulting Indicators: Plus (53)	Minus (54-55)	Zero or Blank (56-57)	Compare: High 1>2 (58)	Low 1<2 (59)	Equal 1=2 (60-61)	Comments (62-74)
01	C																					
02	C																					
03	C			Ø6	N15					EXSR	SUB1											
04	C																					
05	C																					
06	C			32						EXSR	SUB1											
07	C																					
08	C																					
09	C	L1								EXSR	SUB1											
10	C	L1																				
11	C	L1																				
12	C	SR							SUB1	BEGSR												
13	C	SR																				
14	C	SR																				
15	C	SR								GOTO	ENDSUB											
	C	SR																				
	C	SR																				
	C	SR								ENDSR												
	C																					
	C																					

Card Electro Number _____

Figure 100. RPG Internal Subroutine

The ENDSR statement defines the end of the subroutine and the point of exit from the subroutine. At the completion of the subroutine execution, program control leaves the subroutine and resumes with the calculation specification immediately following the EXSR statement that invoked the subroutine.

The ENDSR may have a name entered in Factor 1. This name may serve as the destination for a GOTO statement within the subroutine. A TAG statement is not needed when the GOTO operation is used with the ENDSR statement.

All statements of an RPG subroutine must have SR entered in columns 7-8 of the specification. These columns are otherwise used for control-level indicators. Control-level indicators may not be used to condition statements within an RPG subroutine. However, the EXSR statement that invokes the subroutine may be conditioned for total time. Thus, the entire subroutine will be executed under the conditions imposed on the EXSR statement.

An EXSR statement is used to invoke an RPG subroutine. EXSR statements may appear in detail calculations, total calculations, or within another subroutine. (However, when an EXSR is a statement in a subroutine, it may not invoke the subroutine of which it is a part.) When the EXSR statement is executed, the control of the program goes to the subroutine named in Factor 2 and that subroutine is executed. When the subroutine has been executed, control returns to the statement following the EXSR statement.

Order of Specification

RPG subroutines follow all detail and total calculations on the Calculation Specification sheet.

Order of Execution

Although the RPG subroutine specifications follow all other calculation specifications, they are executed according to the order and placement of the EXSR statements that invoke them. The EXSR may be used anywhere in the calculation specifications. However, the user should be aware of the following considerations regarding four specific positions of the EXIT operation in the operation:

1. When the EXSR operation is the first detail calculation of the program, control will be transferred upon completion of the input routine, i.e., as soon as the pertinent input data are available for processing.

2. When the EXSR operation is the last detail calculation of the program, control will be transferred to the designated subroutine immediately before heading and detail records are printed or punched.

3. When the EXSR operation is the first total calculation of the program: The exit to the subroutine will be made after the record type has been determined and the control field has been tested.

4. When the EXSR operation is the last total calculation, control will be transferred immediately before the totals are printed or punched.

5. When the EXSR operation is contained within another subroutine, the subroutine named by the EXSR statement will be invoked when the subroutine containing the EXSR statement is executed.

Since the RPG subroutine is internal to RPG, any data field may be referenced in the subroutine. Fields may be accessed and changed just as with any calculation specifications.

A GOTO operation may not be used to branch to the label of the BEGSR operation.

If a TAG operation is located within a subroutine, the corresponding GOTO must be within the same subroutine.

It is possible to have a GOTO statement within the subroutine branch to a TAG statement outside of the subroutine. However, that TAG statement cannot be in another subroutine.

Example

Figure 101 shows an updating routine using an RPG subroutine. This approach is typical of most "spread-card" type applications.

Part 1 shows how the three files involved are described on the File Description Specifications.

Part 2 shows how the two input files are described on the input specifications. The fields ITEM1, ITEM2, ITEM3 are defined for part numbers, and the fields QTY1, QTY2, QTY3 are defined for the quantity of parts for The primary file CARD. the inventory file (INVFIL) key and balance fields are also defined.

Part 3 shows the Calculation Specifications necessary to use the subroutine.

Lines 010-030 The fields ITEM and QTY are defined and the first item information moved into these fields. The subroutine UPDATE is invoked.

Lines 040-090 Information for the second and third items is used to invoke the subroutine UPDATE.

Line 110 The beginning of the subroutine UPDATE is identified by the operation BEGSR.

Line 120 The CHAIN operation is used with field ITEM and the file INVFIL to access the inventory file record for this item. (See Chaining for more information about the technique of chaining.) Indicator 20 will be turned on if the appropriate record cannot be found.

Line 130 In this example the Inventory File contains both active and inactive items. The active items are coded as a 1 in position 100 of the record. This will turn on indicator 02. If it is N02, the programmer will process it as a not found record. The SETON of indicator 20 handles this condition.

Line 140 When indicator 20 is not on (N20), the inventory file record is updated by the field QTY.

Line 150 The operation EXCPT causes any exception records defined on the Output-Format Specifications sheet to be produced at this time. (Exception records are identified by an entry of E in column 15.) In this example, the record produced is the inventory record containing the updated quantity. This record is written back into the file INVFIL, completing the update.

Line 160 The ENDSR defines the end of the subroutine and causes program control to return to the statement immediately following the EXSR statement that invoked the subroutine.

Part 4 shows the output specification coding necessary for putting out the exception records as explained previously.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	I/O/U/C 15	File Type: P/S/C/R/T 16	File Designation: E 17	End of File: A/D 18	Sequence: F/V 19	File Format 20	Block Length 21–24	Record Length 25–27	Mode of Processing: L/R 28	Length of Key Field or of Record Address Field 29–30	Record Address Type: K/I 31	File Organization: T/D/T 32	Overflow Indicator 33–34	Key Field Starting Location 35–38	Extension Code E/L 39	Device 40–46	Symbolic Device 47–52	Labels (S, N, or E) 53	Name of Label Exit 54–59	Extent Exit for DAM 60–65	File Addition: A 66	No. Tracks for Cylinder Overflows 67–68	No. of Extents 69	Tape Rewind: N/U 70	71–74
01	F	CARD	I	P			F			80								READ	42									
02	F	INVFIL	I	C			F			100	R	3	K	I			1	DISK										
03	F	PRINTER	O				F			120								PRINTER										
04	F																											
05	F																											

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	Sequence 15–16	Number (1-N) 17	Option (O) 18	Resulting Indicator 19–20	Record Identification Codes 1: Position 21–24	1: Not (N) 25	1: C/Z/D 26	1: Character 27	2: Position 28–31	2: Not (N) 32	2: C/Z/D 33	2: Character 34	3: Position 35–38	3: Not (N) 39	3: C/Z/D 40	3: Character 41	Stacker Select 42	Packed (P) 43	Field Location: From 44–47	Field Location: To 48–51	Decimal Positions 52	Field Name 53–58	Control Level (L1-L9) 59–60	Matching Fields or Chaining Fields 61–62	Field-Record Relation 63–64	Field Indicators: Plus 65–66	Minus 67–68	Zero or Blank 69–70	Sterling Sign Position 71–74
01	I	CARD	AA			01																								
02	I	* HEADING PORTION																												
03	I																			1	5		CUST							
04	I	* SPREAD PORTION																												
05	I																			6	8		ITEM1							
06	I																			9	10		QTY1							
07	I																			12	14		ITEM2							
08	I																			15	17		QTY2							
09	I																			18	20		ITEM3							
10	I																			21	23		QTY3							
11	I																													
12	I																													
13	I	INVFIL	AB			02	100		C	1																				
14	I																			1	3		KEY							
15	I																			4	7	0	BALANC							
	I		AC			03																								
	I																													
	I																													
	I																													
	I																													

Card Electro Number _____

Figure 101. RPG Subroutine (Part 1)

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And / And — Not 9 10 11 12 13 14 15 16 17	Factor 1 18–27	Operation 28–32	Factor 2 33–42	Result Field 43–48	Field Length 49–51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2 — 54 55 56 57 58 59	Comments 60–74
01Ø	C				MOVE	ITEM1	ITEM	3				
02Ø	C				MOVE	QTY1	QTY	3Ø				
03Ø	C				EXSR	UPDATE						
04Ø	C				MOVE	ITEM2	ITEM					
05Ø	C				MOVE	QTY2	QTY					
06Ø	C				EXSR	UPDATE						
07Ø	C				MOVE	ITEM3	ITEM					
08Ø	C				MOVE	QTY3	QTY					
09Ø	C				EXSR	UPDATE						
10Ø	C											
11Ø	C	SR		UPDATE	BEGSR							
12Ø	C	SR		ITEM	CHAININVFIL						2Ø2Ø	
13Ø	C	SRN02			SETON						2Ø	
14Ø	C	SRN2Ø		BALANC	SUB	QTY	BALANC					
15Ø	C	SR			EXCPT							
16Ø	C	SR			ENDSR							
	C											
	C											
	C											
	C											

Card Electro Number _____

IBM

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 3 4 5	Form Type 6	Filename 7–14	Type (H/D/T) 15	Stacker Select 16	Space: Before 17 / After 18	Skip: Before 19–20 / After 21–22	Output Indicators: And / And — Not 23 24 25 26 27 28 29 30 31	Field Name 32–37	Zero Suppress (Z) 38	Blank After (B) 39	End Position in Output Record 40–43	Packed Field (P) 44	Constant or Edit Word 45–70	Sterling Sign Position 71–74
01Ø	O	INVFIL	E				Ø1N2Ø							
02Ø	O							BALANC			8			
03Ø	O													
04Ø	O	PRINTER	E		2		Ø1							
05Ø	O							ITEM			5			
06Ø	O							QTY	1		15			
07Ø	O						2Ø						'NOT FOUND'	
08	O													
09	O													

Figure 101. RPG Subroutine (Part 2)

RPG provides a relatively easy method of documentation since it has a fixed entry format and all programs use the same generalized logic flow. To assist the programmer, the following discussion concerns ways of providing additional documentation that will assist in program writing, program testing, and program maintenance.

1. Comment cards at the beginning of a program allow the programmer to describe the functions of his program and any other important items. An example of this is found in Figure 101.1.

2. Comment cards can easily be written on any specification sheet to denote the beginning of major sections. An example of this is shown in Figure 101.2.

3. Comments can also be written in the comment section of the Extension and Calculation Specifications. See Figure 101.3 and 101.4.

4. A special RPG documentation form is available to allow the programmer to keep track of the indicators he has used and to supply a description of their purpose in his program. This is a convenient aid during the writing of a program and a good means of providing documentation. The form is designed to be keypunched and the cards are normally placed after the File Description entries. A sample is shown in Figures 101.5 and 101.6.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347-3
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 01

Program Identification: PAY050

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind N/U
01	F	*											PAY050											
02	F	*																						
03	F	* PAYROLL DEDUCTION REGISTER																						
04	F	*																						
05	F	* THIS PROGRAM TAKES THE OUTPUT OF PAY050 AND LISTS IT																						
06	F	* AS THE DEDUCTION REGISTER. GROSS PAY SHOULD BALANCE																						
07	F	* WITH PAY030. DEDUCTIONS SHOULD CROSSFOOT AND AN																						
08	F	* ERROR MESSAGE IS PRINTED IF THEY DO NOT.																						
09	F	*																						
10	F	*																						
11	F	DISKIN IP			F				100						DISK									
12	F	PRINTER O			F				120			OF			PRINTER									
13	F																							

Figure 101.1 Program Documentation

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And / And — Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank — Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C	*		DETAIL CARD								
02	C											
03	C											
04	C											
05	C											
06	C											
07	C	*		FEDERAL WITHHOLDING TAX								
08	C											
09	C											
10	C											
11	C	*		FICA CALCULATION								
12	C											
13	C											
14	C											
15	C	*		NET PAY								
	C											
	C											
	C											
	C											
	C											

Card Electro Number _____

Figure 101.2 Program Documentation

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page

Program Identification

Line	Form Type	Record Sequence of the Chaining File: Number of the Chaining Field — From Filename	To Filename	Table Name	Number of Table Entries Per Record: Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed(P)	Decimal Pos.	Sequence (A/D)	Comments
01	E	TABL1		TABMAN	1	75		3		TABRAT	4	2			MAN AND RATE
02	E	TABL2		TABCOD	1	25		2							DEPARTMENT CODES
03	E	TABL3		TABDAY	1	7		1		TABHRS	6	2			DAY AND HOURS
04	E														
05	E														

Figure 101.3 Program Documentation

IBM

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 3 4 5	Form Type 6	Control Level (LO-L9, LR) 7 8	Indicators: And 9 10 11	Indicators: And 12 13 14	Indicators: And 15 16 17	Factor 1 18–27	Operation 28–32	Factor 2 33–42	Result Field 43–48	Field Length 49–51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus 54 55	Resulting Indicators: Minus 56 57	Resulting Indicators: Zero or Blank 58 59	Compare: High 1 > 2 60 61	Compare: Low 1 < 2 62 63	Compare: Equal 1 = 2 64 65	Comments 66–74
01	C					ONHAND	SUB	QTY	TEST	60				20					TEST QTY
02	C		20				GOTO	END											NOT ENOUGH
03	C						Z-ADD	TEST	ONHAND										
04	C					MINBAL	SUB	QTY	TEST					21					MIN BAL TEST
05	C		21				EXCPT												PUNCH NOTICE
06	C																		

Figure 101.4 Program Documentation

IBM System/360 Report Program Generator Indicator Summary

Form X26-5590
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Circle Indicators Used:

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	OF
L0	L1	L2	L3	L4	L5	L6	L7	L8	L9	LR	M1	M2	M3	H1	H2	C1	C2	C3	OV

Line 3 4 5	Form Type 6 7	Indicators: Input Resulting 8 9 10	Indicators: Calc. Resulting 11 12 13	Indicators: Field 14 15 16	Indicators: Matching & Chaining 17 18 19	Indicators: Ctl. Break & Halt 20 21 22	FUNCTION OF INDICATORS 23–74
01	F *	I	C	F	M	L	FUNCTION OF INDICATORS
02	F *						
03	F *						
04	F *						
05	F *						
06	F *						
07	F *						
08	F *						
09	F *						
10	F *						
11	F *						
12	F *						
13	F *						
14	F *						
15	F *						
	F *						
	F *						
	F *						
	F *						
	F *						

Figure 101.5 Program Documentation

IBM System/360 Report Program Generator Indicator Summary

Form X26-5590
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction	Graphic								
	Punch								

Page 02

Program Identification PAY050

Circle Indicators Used:

(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	OF
L0	(L1)	(L2)	L3	L4	L5	L6	L7	L8	L9	(LR)	M1	M2	M3	(H1)	H2	C1	C2	C3	OV

Line	Form Type	Input Resulting	Calc. Resulting	Field	Matching & Chaining	Ctl. Break & Halt	
		Indicators					
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
01Ø	F *	I	C	F	M	L	FUNCTION OF INDICATORS
02Ø	F *	Ø1					MASTER RECORD
03Ø	F *	Ø2					CURRENT EARNINGS
04Ø	F *	Ø3					YTD SUMMARY
05Ø	F *	Ø4					DEDUCTIONS
06Ø	F *		11				CROSSFOOT OF MAN IS EQUAL
07Ø	F *		12				FICA CUTOFF REACHED
08Ø	F *		13				TOO MANY DEDUCTIONS
09Ø	F *					L1	MAN BREAK
10Ø	F *					L2	DEPARTMENT BREAK
11Ø	F *			H1			NO DATE CARD
12	F *						
13	F *						

Figure 101.6 Program Documentation

The RPG object program handles records that are fixed-length only (i.e., all the records of a file have the same length). One logical record is contained in one physical record.

UNBLOCKED RECORDS

All logical records of a file have the same length.

The length of a logical record is entered in Record Length on the File Description Specifications form.

BLOCKING RECORDS

RPG automatically blocks all disk files for maximum space utilization. Figure 102 shows the number of cylinders that are used to contain 1000 records of various sizes.

A disk pack consists of fixed-length sectors with a capacity of 640 characters each. A data record must be contained in one sector. However, a sector may be capable of containing more than one record. The process of grouping records on a sector is called blocking.

The blocking of disk records saves disk operations and increases the processing speed of the object program. When records are grouped in blocks of one sector, an unused area may occur at the end of the sector.

Figure 102 shows the utilization of disk space achieved with various blocking factors.

Record size in Characters	Cylinders Used Per 1,000 Records	
	Sequential	Index-Sequential: Prime Data Area
1	0.4	1.1
2	0.4	1.1
6	1.2	1.9
8	1.6	2.4
20	4.0	4.8
50	12.8	11.3
80	15.6	18.0
120	25.0	25.0
200	41.7	41.7
250	62.5	62.5
318	62.5	62.5
320	62.5	125.0
636	125.0	125.0
640	125.0	INVALID

Key Length in Characters	Index Sequential: Cylinder Index Chart
	Number of Entries On One Sector
2	45
3	35
5	24
8	16
10	13
14	10
18	8
22	6
25	6
30	5
40	3
50	3

Figure 102. Space Utilization for Various Size Records

SIGN CONTROL

1130 RPG does not change the sign of any fields defined as numeric in the following situations:

- Removed from an Input Record
- Moved by a MOVE or MOVEL operation
- Moved by a Move Zone operation (the sign of the result field will be the same as the sign of Factor 2)
- Used as a Chaining Field

The result field of any arithmetic operation (ADD, SUB, Z-ADD, Z-SUB, MULT, DIV, MVR) will always produce either the positive sign (F) or the negative sign (D).

If two numeric fields are compared and the digit portions are equal, the following will occur:

1. If the signs are identical, the compare will be equal.
2. If one sign is a D (negative) and the other is not a D (positive), the compare will be unequal.
3. If the two signs involved are not D's and are not the same (e.g., F and C) the compare will be equal.

Numeric Match Fields and Control Level Hold Areas

An F sign will be forced on all numeric fields regardless of what the sign was previously. In addition, all high order zones will be stripped.

Editing

Edit codes and edit words will treat all D signs as negative. All other signs will be considered positive.

TERMINOLOGY

The distinction between the terms file organization and file processing is very important.

File Organization is the method ofarranging 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 1130 components, carefully consider the relationship between how a file is organized and how records are retrieved from it. This is particularly important when designating data files for storage in a direct access storage device.

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. More than one method of processing the same file can be used. For example, records within a file might be processed at random during an updating run, and sequentially during a billing run.

FILE ORGANIZATION

The two types of file organization used by 1130 RPG are sequential, and indexed sequential.

Sequential File Organization

A sequentially organized file is established by arranging records into sequential order in the storage media used to contain the file. Card files are always organized in this manner. This kind of file is considered as one continuous string of records in sequence and is processed consecutively. Disk records can also be organized in this manner and processed consecutively. The size of a sequentially organized disk record can be any number from 1 to 640 characters.

It is also possible to process a sequential file randomly. This can be done by specifying the relative record number of the record in the file.

Indexed-Sequential Organization

In this type of file organization, the records are also stored sequentially in the file. But this type of file organization differs from sequential organization.

The indexed sequential type of organization uses an index table associated with a file that indicates to the program the general location of the records. Thus, the records may be retrieved from the file sequentially or in random sequence. The index table is analogous to the index card file in a library. If you know the name of a book, or the author, you can look in the index and find the location of the book in the book files.

For example, this might be a catalog number (address) of 426.25. You would then go to the book shelves, and (if it was your first time in the library) start at the first row of the book files and proceed through the rows until you have found the shelf that contains 426.25. Usually, each row contains a sign to indicate the beginning and ending numbers of all books in that particular row. Thus, as you proceed through the rows, you would compare 426.25 with the numbers posted on each row. Assume that one row was labeled 300.88-550.00. You would then search that row for the shelf that contained the book.

The RPG program uses an index table in much the same way to locate records organized in an indexed sequential file.

The size of a record in an indexed sequential file can be any number from 1 to 636 characters. Each record in the file has a field called a key. This key is the field which the file is organized by. In the library example, each book has a catalog number which is its key. The size of a key in an indexed sequential file can be any number from 1 to 50 characters.

FILE PROCESSING

There are four methods of file processing:

1. Sequential processing of sequentially organized files.
2. Random processing of sequentially organized files.
3. Sequential processing of indexed sequentially organized files.

4. Random processing of files organized indexed-sequentially.

Sequential Processing (Sequential Files)

In sequential processing of a sequentially organized file, every record in the 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 cannot be processed until after the 13th card has been processed.

Random Processing (Sequential Files)

To find a random record in a file organized sequentially a record number must be supplied. The user can then make direct access to the record. Index tables are not required. This kind of processing is called random processing by a relative record number. The record identification indicates the location of the record on the file. For example, the 12th, 83rd, 212th, etc. record on the file. The program makes no comparison of key (control field) data with this type of processing.

This type of processing can also be compared to directing someone to a house location. "The Martin family lives on Harrison Street" (a file name) "and their house is the 32nd house from the beginning of the street." (The 32nd is the record number.)

Sequential Processing (Indexed Sequential Files)

Sequential processing of an indexed-sequentially organized file is similar to sequential processing of a sequentially organized file (i. e., an entire file is processed). In the example of the library, the first visit represents a sequential processing on an indexed-sequentially organized file.

An additional method of sequential processing is to process only a segment of a file. For example, a payrcll 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. Therefcre, on each week's processing, only segments of the payroll file are updated. The updating is accomplished by reading in a card file that specifies the limits of the file to be processed (called a Record Address or RA file). One such card record might indicate that the records for department's 26-41 are to be updated, another fcr department's 76-80, etc.

Random Processing (Indexed Sequential Files)

In random processing, the sequence of processing has no relationship tc the sequence in which the data is stored in the file.

To randomly find a record in an indexed sequential file, an index is first scanned to localize the area of search by determining the cylinder that contains the record. The index is a sequential list of the keys (of the data) with corresponding cylinder addresses. The entire cylinder 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 cylinder address) "and then walking down the street looking for 'Martin' on a mailbox".

CREATING AND PROCESSING SEQUENTIAL DISK FILES

A sequential disk file must be contained within one disk area. This area is defined prior to the creation of the file which occupies it. (Refer to IBM 1130 Disk Monitor System, Version 2, Programming and Operating Guide for a discussion of the DUP function used to reserve an area on disk.)

The RPG program loads the records, one after another , beginning at the start address of the file and continues loading on successive tracks (and cylinders) until the end address is reached.

If the file is defined as an update file, the records to be updated can be specified on the Output-Format Specifications form. Furthermore, only the fields of an update record that are to be changed need be defined on the output specifications. The RPG program combines these fields with the unchanged fields and restores the updated records into the same locations on disk, before retrieving a new record for processing.

To extend a sequential file or add records, the entire file must be reloaded.

If a sequential disk file is processed sequentially, it can be updated only at detail time during the processing of this record. At any other processing time, updating would be performed on another record. If a sequential disk file is processed randomly, it can be updated at either detail or total time.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
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	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
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 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	
010	F	INPUT	IP			A	F		80							DISK										
020	F	SEQUADATUS				A	F		80							DISK										
030	F	OUTPUT	O				F		80							DISK										
040	F	RANDOM	I				F		80	R						DISK										
050	F	RANDOMUPU					F		80	R						DISK										
060	F																									
070	F																									

Figure 103. File Definitions

A sequential disk file may be processed randomly by using relative record numbers but must be created through the use of a sequential output file.

Figure 103 shows the coding on the File Description Specifications form required to process records of sequential input, update and output disk files.

Each file is defined to have records 80 characters in length. Since each sector can contain 640 characters each file will have 8 records on each sector defined for that file. Line 010 shows an input file as the primary file. Line 020 shows a file being processed as a sequential file and updating is allowed. Line 030 shows a file being created. Line 040 shows a sequential file being processed randomly and being retrieved by a relative record number. Line 050 shows a file being processed randomly and updated. Retrieval is by means of a relative record number.

INDEXED SEQUENTIAL ACCESS METHOD (ISAM)

An indexed sequential file is alwayssequenced according to a key field. It must be in ascending sequence.

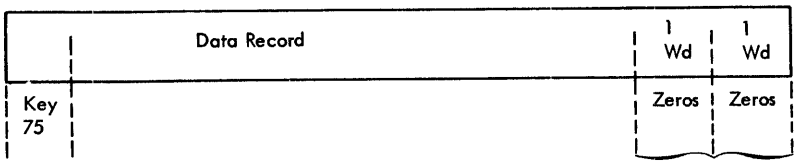
Every indexed sequential file contains an index table associated with the file which has one key for every cylinder of the file and the address of the cylinder. This index acts as a form of brackets so that any key may be isolated to begin on a particular cylinder. Thus, the records may be retrieved from the file sequentially or in random sequence

Key

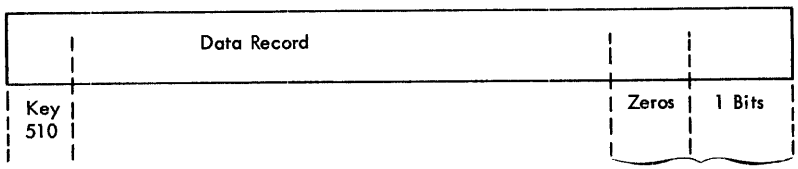
The key may be a part number, or an employee serial number, or any identifying information that is contained in any record of the file. All keys of the file must have the same length and must begin in position 1 of the first part of a record. The maximum size key is 50 alphameric characters. The records are stored in a key sequence and must have been sorted by a key. An end of file record (EOF) is a record that has a key with the highest possible value (:.e., all bits are 1). Erroneous results will be achieved if a data record with this key field is part of the file.

Organization of Records on Disk

When the file is loaded onto disk, the ISAM load routine organizes the file in a way that allows direct access to any record. A reference can be to records at random throughout the logical file, or to a series of records in the file in their presorted sequence. The records are loaded by the ISAM load routine one after another, into a specified area of a volume (the prime data area of the volume). A two word control field is suffixed to each record prior to writing the record in the prime data area. Data records cannot exceed 636 characters (reference Figure 104 for format of data record).



Example of a data record in the prime data area.



Example of the last data record on a prime data cylinder that has overflow records.

Figure 104. Data Format

Master Index

When the ISAM load routine loads a file of records an index for the file is created. This index is utilized for the random or sequential reference to records as follows:

1. Key from the last record on each prime data cylinder.
2. Sector address from the beginning of a cylinder.
3. Key - the same as item 1 except when overflow records exist for that cylinder, in which case it will be the key that was originally the key of the last record on that cylinder.
4. Sector address - same as item 2 except when overflow records exist, in which case it will be the overflow sector address of the last record forced off the cylinder.
5. Record number - zeros except when overflow records exist, in which case it will contain the record location on the overflow sector of the last record forced off the cylinder.

The entries are blocked if possible, (i.e., one disk sector contains more than one entry). The exact number of entries per sector and the number of entries required for the index depend on the users key length; the sector addresses and the record number require three words (Figure 105).

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3347
Printed in U. S. A.

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	I/O/U/C 15	File Type: P/S/C/R/T 16	File Designation: E 17	End of File: A/D 18	Sequence: F/V 19	File Format	Block Length 20–23	Record Length 24–27	L/R 28	Mode of Processing	Length of Key Field or of Record Address Field 29–30	Record Address Type: K/I 31	Type of File Organization: I/D/T 32	Overflow Indicator 33–34	Key Field Starting Location 35–38	Extension Code E/L 39	Device 40–46	Symbolic Device 47–52	Labels (S, N, or E) 53	Name of Label Exit 54–59	Extent Exit for DAM 60–65	File Addition: A 66	No. Tracks for Cylinder Overflows 67–68	No. of Extents 69	Tape Rewind: N/U 70	71–74
01Ø	F	SEQ	I	P		A	F			6Ø		5	K	I				L	DISK									
02Ø	F	SEQLIMIT	I	S		A	F			75	L	6	K	I				L	DISK									
03Ø	F	RAF	I	R			F			8Ø		6							EREAD42									
04Ø	F	SEQUPDAT	U	S			F			1ØØ		15	K	I				L	DISK									
05Ø	F	RNDM	I	C			F			8ØR		4	K	I				L	DISK									
06Ø	F	RNDMUP	U	C			F			4ØR		3	K	I				L	DISK									
07Ø	F	LOAD	O				F			5Ø		5	K	I				L	DISK	3ØØØ								
08Ø	F	ADDITNS	O				F			5Ø		5	K	I				L	DISK					A				
09	F																											
10	F																											

Figure 104.1 Typical ISAM definitions

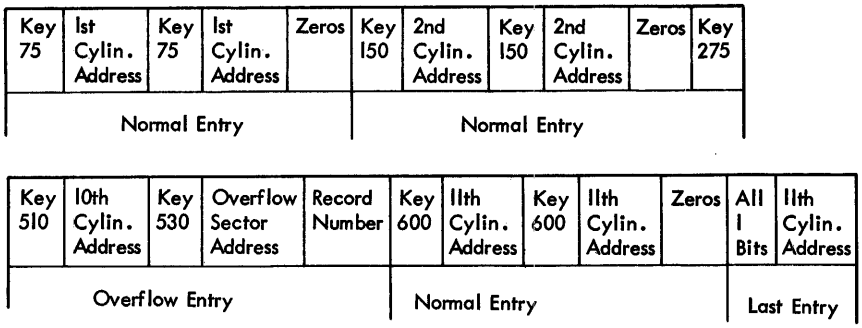


Figure 105. ISAM Master Index Format

Overflow Area and Addition of Records

After a file has been organized on disk, it may subsequently become necessary to add records to the file. The records to be added may contain:

1. Keys that are above the highest key presently in the file.
2. Keys that are lower than the lowest key presently in the file.
3. Keys that fall between keys that are already in the file.

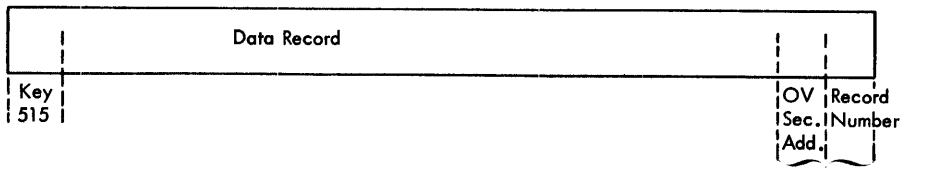
When new records are to be added, an overflow area is required. The ISAM uses the overflow area to permit the insertion of records without necessitating a complete reorganization of the established file.

The random and the sequential retrieval of records is maintained by inserting references to the overflow records in the master index and by using a linking technique in the overflow records. For chaining, the sequence-link control field is suffixed to all records. This sequence-link control field is a two word area and contains the sector address and the record number. Thus, a chain of sequential records can be followed in a search for a particular record. The sequence-link control field of zeros in the chain indicates the end of the chain (see Figure 106 for a schematic example of the overflow area). To add a record, the ISAM ADD routine searches the master index to determine on which cylinder the record is to be placed. When the proper cylinder is determined, one of the following routines is performed.

1. If the record is to be placed ahead of the records presently on the cylinder or between two of the records, the ISAM ADD routine adds the record by inserting it in the proper sequence and shifting each succeeding record one record location higher on the cylinder, until the end record is forced off the cylinder. The end record is transferred to the overflow area and the master index is updated or the sequence-link control field of the overflowed records are updated to chain the overflow records in sequence.

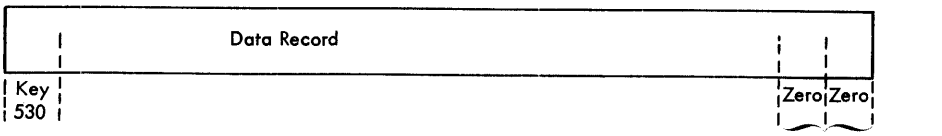
2. If the record is to be placed between the last record presently on the cylinder and the last record originally on the cylinder, the ISAM ADD routine writes the records in the overflow area following the last record previously written. The routine searches through the chain of records associated with the corresponding cylinder for this record. Then the sequence link control fields of the new record, and the record preceding it by a sequential key, are adjusted to point to the proper records.

3. If the new record has a key that is higher than the users highest key in the file, the record is inserted before the end of file record. If the last prime data sector is not full, the record is placed in the prime data area, otherwise it is added the same as in routine 1 or



Sequence-link Control Field

Data record in the overflow area with the control field pointing to next sequential record.



Sequence-link Control Field

Data record in the overflow area with the control field zeros which means it is the last record in the chain.

Figure 106. Overflow Area

2 above (the dummy end of file record key of all 1 bits is the highest key possible). The overflow area is automatically allocated immediately following the prime data area after the initial load of the file if space is available within the limits of the file area (ses Figures 107 and 108 for a schematic example of the cylinder).

Note 1: The above description of adding is logically correct, but the technique is to work backwards so that the record to be added is written last. This avoids the loss of records due to hardware malfunction.

Note 2: 1130 RPG does not permit the extents of an organized file to be changed. Therefore, if the available overflow areas are full and records are still to be added, the file must be completely reorganized.

Estimating File Size Capacity

As a technique of good systems design, it is desirable to quickly estimate possible file sizes for the three methods of file organization. Since 1130 RPG automatically blocks records and the block size cannot exceed 640 (636 for ISAM), there are certain optimal record sizes.

For example, an ISAM file of 3,000 125 character records will require approximately 140 cylinders. If the record size can be reduced to 120 characters, the file size is reduced to approximately 85 cylinders. Thus with a 4% reduction in record size a 40% reduction in file space is achieved. This will also significantly improve the retrieval speed of the file. Record sizes can often be reduced with system design considerations or with packing of numeric fields.

If on the other hand, the record size cannot be reduced to 120 characters, the user should consider expanding the record size to 200 characters since the same amount of file space will be required. This may allow additional information to be stored on the record which may reduce the total number of operations in the system design or improve the capability of the system.

Because of the potential savings, the system designer should be familiar with Figure 102. This chart shows the number of cylinders required per 1000 data records.

The user first calculates his record size in characters. Then looks down the record size column for the number which is equal or the next higher value. By looking

across, the number of cylinders required for 1000 records for either Sequential or ISAM can be determined. A simple multiplication will then produce the estimated file capacity required.

In our previous example, a 125 character record would be the same as a 200 character record. (It is larger than 120). Looking across under the ISAM column, 41.7 cylinders are required. Since this is for 1000 records 41.7 x 3 = 125.1 or 126 cylinders are required for the prime data area.

An indexed sequential file requires a cylinder index. This chart is also shown in Figure 102. Look down the key length column to the number equal to or larger than the key field of the file. (Key fields cannot be packed). Then look across to determine the number of cylinder entries that can be made on one sector.

If a key field length of 7 is required, 16 cylinder entries can be held per sector. Since the prime data area will require 126 cylinders, then eight (125 divided by 16) sectors will be required for the cylinder index. Since there are eight sectors in a cylinder, one cylinder is required for the cylinder index.

Most ISAM files require an extra area for overflow records. This will vary per application, but a typical figure may be 10%. If the prime data area requires 125 cylinders, the overflow area will be 13 cylinders.

The total file is then contained in:

```
126 cylinders for prime data
 13 cylinders for overflow area
  1 cylinder for index
---
140 Cylinders
```

The 1130 System requires that a file space be reserved for the file. This number must be expressed in terms of sectors required. Therefore a reserved area of 1120 sectors (140x8) is required for this file.

Sectors Required for an Indexed Sequential File

RPG will automatically calculate the number of sectors required for the file for each program that will load an ISAM file. This feature will assist in determining the file capacity required.

The user must specify the total number of records he wishes to have in an ISAM file and enter this on the File Description Specifications. This number should provide for a typical overflow area.

Examples. User specifies 2000 records to be loaded (100 characters long, key field=6 characters). The RPG compiler will show that 408 sectors (51 cylinders) are required for the disk area. (The user could have calculated this using Figure 102.) The user must reserve a number of sectors on the disk in a DUP run for this file. It is not mandatory to reserve the exact amount specified by RPG. The following depicts some examples and what factors should be considered.

1. If 408 sectors are specified, the user can initially load 1500 records and use the remaining space for overflow records. This will waste a small amount of disk space that has been reserved for the index to the records that were not loaded. (In most examples this is negligible. Here it amounts to one sector.) If the user initially loads 2000 records, no records can be added to the file. This file would have to be reorganized specifying a larger area. This will require another DUP run and reloading of the file.

2. If 500 sectors are specified, the user could load 2000 records and use the remaining area for additions. It would not be possible to initially load 2200 records since RPG has already set the maximum index size at 2000 records. (The load program will halt when too many records are loaded.) The RPG load program would have to be recompiled with a larger maximum file size before reloading.

3. If 350 sectors are specified the user can obviously not load 2000 records. (The actual number would be 1730 records as determined by using Figure 102.) However, 1500 records could be loaded and the remaining area could be used for additions. (As in example 1, a small amount of area would be lost due to the requirements for a larger index).

Indexed Sequential Processing

Indexed Sequential Access Method (ISAM) permits the user to process disk records in either random or sequential order, using certain control (key) information.

Sequential Processing: The entire file is processed sequentially by record key in ascending sequence. Sequential processing can be performed on an entire file, starting with the first record or upon limits of the file.

Processing Limits of an Indexed Sequential Organization

If the file is organized alphabetically from A to Z the programmer may wish tc process certain limits of the file such as Through L. It is possible tc process (in the same program) several limits of the file (e.g., A-C, G-K, Q-W). It is also possable to overlap multiple limits (A-L, B-D) or process in a varied sequence (S-W, J-L, C-E, N-P). The limits to be processed are supplied to the program in the form of Record Address file (RA file). The actual beginning and ending limits used do not have to be valid keys within the file. An RA file is used to indicate what limits of a file with an indexed sequential crganization are to be processed.

Creating an RA File for Limits

1. Only two record address entries specifying the lower and upper limit of one area can be in each record of the RA file.

2. The record address entry must begin in position 1 on the record. The first entry indicates the low limit of the file to be processed. The second entry indicates the upper limit of the file. The program will process from the lower limit through the upper limit.

3. The second entry of the record must begin in the position immediately following the first entry. No blank spaces are allowed.

Random Processing

The file can be processed randomly by one of three methods:

1. CHAIN operation on the calculation specifications. (Preferred methcd.)

2. Chaining by using C1-C3. (Alternate method.)

3. Chaining using an RA file. (This method is seldom used.)

For random processing the user supplies the control (key) information be wants and the ISAM random routine retrieves the specified record.

File Descriptien Specification Examples

Figure 108.1 shows examples of typical entries for an ISAM file. All of the entries (except line 030 which is an RA file) have several columns of identical coding.

Column 19 must contain an F for fixed-length records. Column 31 must contain a K for retrieval by key field. Column 32 must contain an I to denote an indexed sequential file. Column 40-46 must contain the name DISK as the device name.

Columns 29-30 must contain the length of the key field. This will vary depending upon file requirements. Columns 24-27 must contain the record length of the file. This will also vary depending upon requirements. Line 010 shows an input file (I in column 15) which is acting as the primary file (P in column 16) in a matching records job. The A in column 18 denotes that the match fields will be in ascending sequence. Line 020 shows a secondary file being processed within limits (L in column 28). The file which is supplying the limits is shown as an RA file in line 030. The R in column 16 denotes an RA file and the E in column 39 denotes that the file will be further described on the Extension Specifications. Line 040 shows a file being updated and processed as a sequential file. Line 050 shows a file being prccessed randomly (R in column 28) as an input file. Line 060 shows a file being processed randomly and updated. Line 070 shows a file being loaded. The O in column 15 is for output. The entry 3000 in columns 47-52 indicates that a maximum of 3000 records will be loaded into this file. Line 080 shows the same file and records being added to it. It is still on output file and the A in column 66 denotes additions.

File Information

The first sector of the file will contain the necessary file information to control the file.

Index in Core

In order to decrease the amount of seek and read/write time involved in ISAM processing, all of the ISAM routines will hold one sector of indexes in core for each file. Because of this technique it may be much faster to sort input data befor doing a random update.

Separation of Function

All files that are to be regarded as ISAM files must have been created by the ISAM load routine. Furthermore, in any one program only one ISAM function can be performed within one RPG program. Similary, additions to an ISAM file cannot be part of the same program that retrieves that file. (Retrieval and updating is only one ISAM function and can be performed in one program.) It is possible to load one ISAM file while a different function is being performed on another ISAM file.

Summary of Indexed Sequential Organization

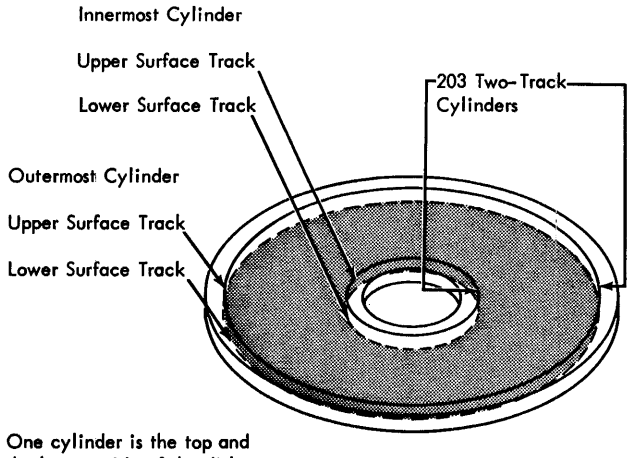
1. An ISAM file must be loaded in the ascending sequence of the key field.

2. Once loaded, records can be added to the file, but the size of the prime data area is fixed by the number of records actually loaded. (Rather than the number specified).

3. If a large number of records are added, the throughput performance in retrieving will be degraded due to the number of searches in the overflow area.

4. If the user attempts to add more records than he has alloted sectors for, there is no facility for expanding the number of sectors available and continuing to add records. The file must be copied sequentially and then reloaded into a larger area. Once this is accomplished, new records may be added.

5. If a record should be deleted, it is up to the user's program to accomplish this through record codes. When the file is reorganized, the user can drop the deletions by means of a record code. Deletions remain on the disk until the file is reorganized.



One cylinder is the top and the bottom side of the disk at one point (two tracks per cylinder).

NOTE: The thickness of the disk has been greatly exaggerated in order to show the relative positions of the upper and lower surface tracks.

Figure 107. 1130 Disk Cartridge

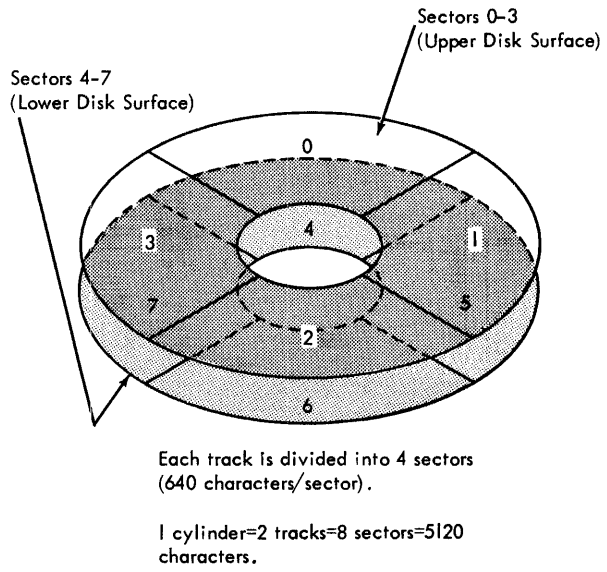


Figure 108. 1130 Disk Storage Sector Number

6. An ISAM file can only be retrieved randomly by specifying the appropriate key as a chaining field.

7. Only one of the following functions can be used per ISAM file per program: LOAD, ADD, retrieve sequentially (with or without updating), retrieve randomly (with or without updating), or retrieve sequentially within limits (with or without updating).

8. The last sector used from the cylinder index remains in core for any ISAM function. This can have a dramatic effect on system performance, especially when dealing with sorted transactions and random updating.

9. Prior to loading the file, an area on the disk containing the number of sectors for the file and overflow area must be reserved by a DUP run. This area should be given the same filename as that used for the file in the File Description Specifications.

When more than one input file is used for an RPG program, several methods of processing these files are possible. The processing method selected is determined by the nature of the job to be performed, and by the organization of the input files. Selecting the file organization is a part of problem definition, and should be selected with care, as the processing methods available to programs using the file are influenced by the choice.

Once the file organization of all input files has been determined, the processing method can be selected. Figure 109 summarizes the possible choices.

File Organization	Possible Mode of Processing
All Card Files (Sequential)	Sequentially
Sequential Disk File	1. Sequentially (Entire File Only) 2. Randomly (Retrieval by Relative Record Number)
Indexed-Sequential Disk Only	1. Sequentially a. Entire File b. Limits of the File (Using an RA File to prescribe the limits) 2. Randomly (Using the chaining technique chaining field) a. Chain Operation on the Calculation Specifications b. Chaining by the alternate method of C1-C3 c. Using an RA File to supply the chaining fields.

Figure 109. Comparison File Organization and Mode of Processing

Sequential Organization: All card files are sequentially organized. Disk files may also be sequentially organized. Such files require that every record be processed in sequence. When more than one sequentially

organized file is used for input, there are two techniques of processing available:

1. The records of the input files may be related in some way, so that processing alternates between the files. The records of the primary and secondary files are "matched" by special fields in these records. This type of processing is called Matching Records, and is discussed below.

2. The records of the input file may be processed independently. All records from the primary file will be processed first, and then each entire secondary file will be processed in the order in which it was specified on the File Description sheet.

Indexed Sequential Organization: Only disk files may be organized as indexedsequential. While such files are arranged sequentially, each record contains a key which is indexed. This allows either sequential or random processing of the file, depending on the needs of the job:

1. The file may be sequentially processed, either the entire file, or between some limits (see RA Files). Matching Record techniques may be used.

2. The file may be processed randomly, by accessing records through their keys. These keys are specified by another file through a process call Chaining (see Chaining).

3. A sequential disk file can be processed randomly by use of the CHAIN operation on the calculation specifications and using a relative record number within the file. The user must know that he wishes to process a certain record (e.g., 31st record).

Combinations: The above organizations and processing methods might be combined in several ways. For example, an application might require three input files, two sequentially organized and one indexedsequentially organized. The two sequentially organized files might be related with matching records, and the indexedsequentially organized file related to either or both of the sequentiall files through chaining.

PRIMARY AND SECONDARY FILES

When sequentially processing multiple input files, one file must be designated the primary file. There must be one and only one primary file. It is specified by entering P in column 16 of the File Description specification for that file. Other input files are all designated secondary files, and are specified by entering S in column 16 of their File Description specifications.

The primary file is generally the controlling file in a program. In the case of processing by matching records, the primary file is processed first when records match. In the case of chaining, the primary file is the chaining file, and hence controls or determines what data is extracted from the chained file for processing.

MATCHING RECORDS

The technique of controlling processing by matching records is used only when processing two or more files sequentially. Examples of matching records:

- A card file and a sequentially-organized disk file (being processed sequentially).
- Three sequentially-organized disk files.
- A card file and an indexed sequentially organized disk file being processed sequentially.

RPG processes only one record at a time. Matching record techniques determine which file supplies the next record to process.

Match fields: Matching fields are specified by entering one of the codes M1 through M9 in columns 61-62 of the Input Specification for the field. (See Matching Fields or Chaining Fields under Input Specifications Form.)

Figure 110 illustrates the specification of matching fields. In this example, there are two input files, MASTER containing certain rate information, and DETLABOR, containing detail records. MASTER is designated as the primary file. Field DEPT from file MASTER is to be compared with field DETDEP from file DETLABOR (M1), and field DIVSON from file MASTER is to be compared with field DETDIV from file DETLABOR (M2). Whenever these fields match, the MR indicator will be turned on.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 03

Program Identification: REPORT

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	I	MASTER	AA			16	80		Z	K																					
020	I																				1	18		DESC							
030	I																				19	23	3	RATE							
040	I																				24	27		DEPT	L2	M1					
050	I																				28	31		DIVSON	L3	M2					
060	I	DETLABOR	BB			14	35		D	5	80	N	Z	K																	
070	I																				1	4		DETDEP	L2	M1					
080	I																				5	8		DETDIV	L3	M2					
090	I																				9	14		EMPNO							
100	I																				15	28		NAME							
110	I																				46	501		FLDA							
120	I																				56	600		FLDB							
130	I																				66	700		FLDC							

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 04

Program Identification: REPORT

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And	Indicators: Not	Indicators: And	Indicators: Not	Indicators: And	Indicators: Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1>2	Compare: Low 1<2	Compare: Equal 1=2	Comments
010	C			MR	14				FLDA	MULT	RATE	AMT		6	2	H						
020	C			NMR	16					SETOF									16			

Figure 110. Example of Specifying Matching Fields

Matching fields determine the manner in which records are matched. Matching records determines which input file supplies the next record for processing. The program begins by reading one record from each input file. The match fields are compared, and one record is selected for processing. The program then selects succeeding records by:

1. Reading the next record from the file which the record just processed came from.
2. Comparing the new record with the record previously read but not selected for processing. One is selected for processing, based on the comparison.

Selection of the next record to process follows these rules:

1. The record with the lowest value in its match fields is processed first (assuming ascending order. If descending order is used, the highest record is selected).

2. If the primary record matches one or more secondary records, the primary record is processed before the secondary records which match it.

3. If two or more secondary records match, they are processed in the order they are specified on the File Description Specifications.

4. Records which have no match fields are processed before records which have match fields.

5. If two or more records are without match fields, they are processed primary file first, followed by secondary fields in the order specified on the File Description Specifications.

Note: If the primary file has an E entered in column 17 of the File Description form (End-of-File) and the secondary files do not, any matched secondaries will be processed before the LR indicator is turned on.

Figures 111 and 112 illustrate how records are selected for processing. Three files are shown in this example, and Figure 111 shows the files as they appear and the sequence in which they are processed. Figure 112 shows the selection process step-by-step for the first ten records.

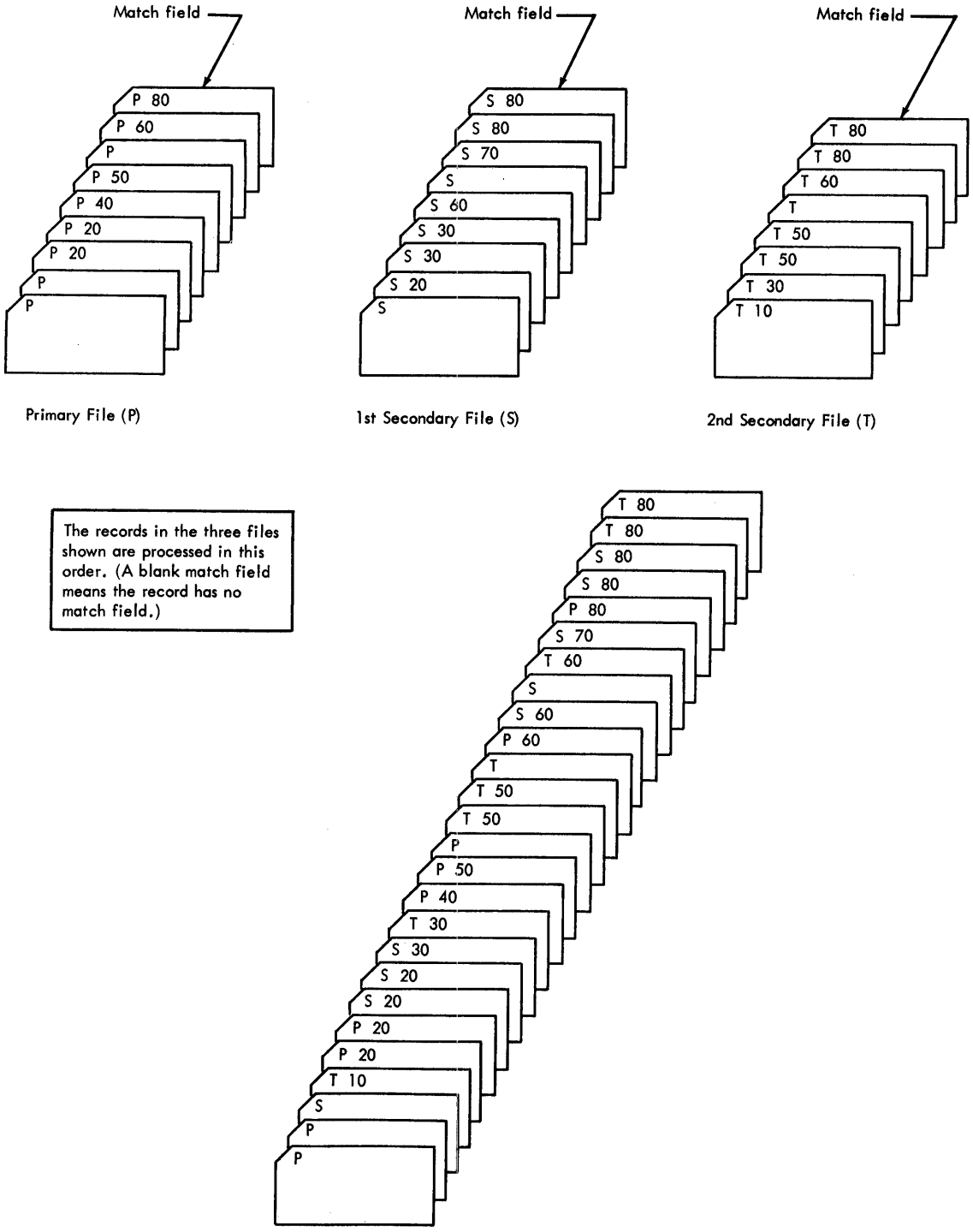


Figure 111. Record Selection (Three Files)

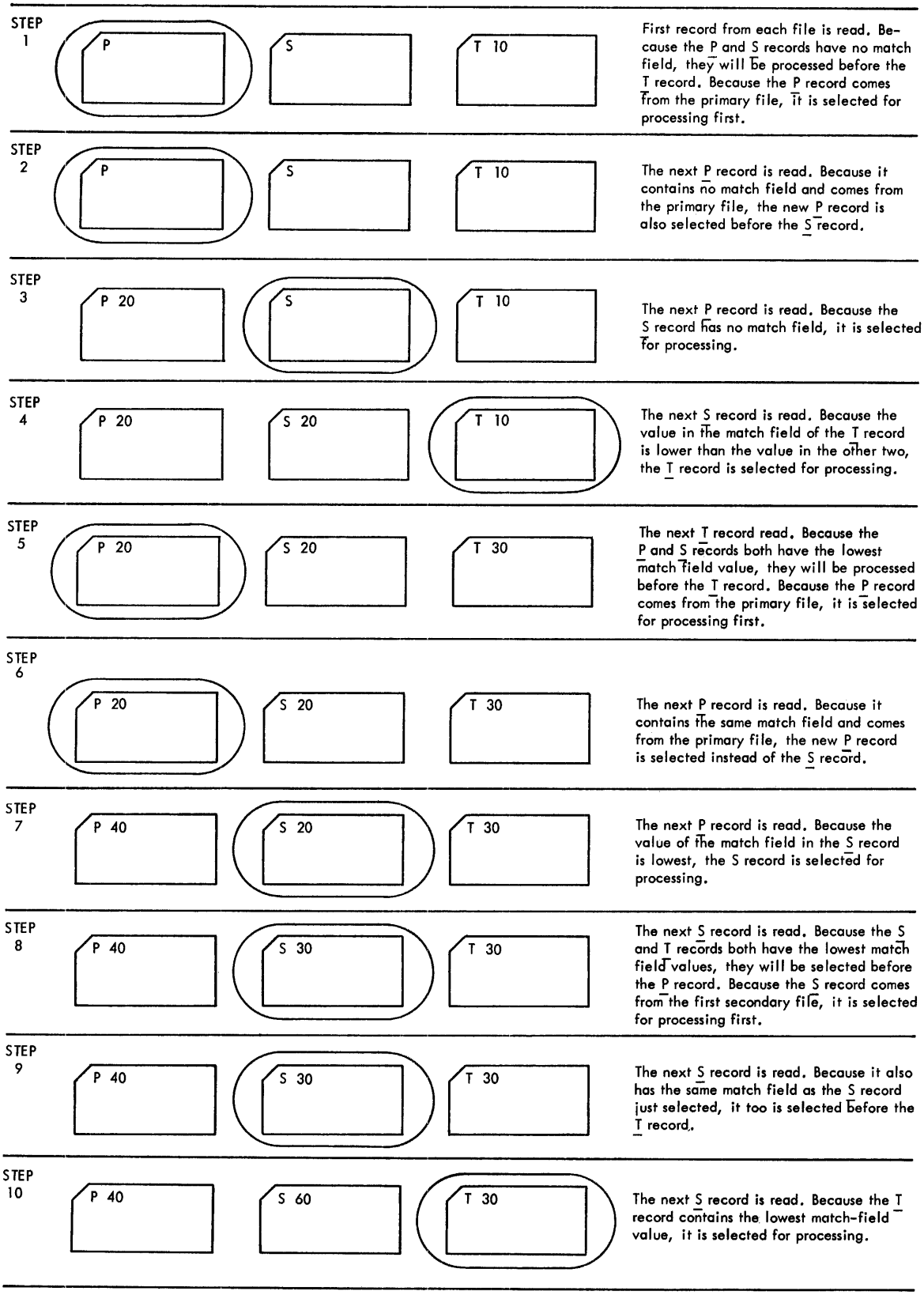


Figure 112. Record Selection (Three Files)

Matching Record Rules:

1. All files using matching fields must use the same matching fields. That is, if one file used matching fields M1 and M2, then M1 and M2 must be defined in all files using matching fields, and only M1 and M2.

2. All matching fields of the same level must be defined to be of the same length and type. That is, M1 of the primary file must have the same length and type as M1 of all secondary files, etc.

3. The order of processing when not under matching record control is primary file first, followed by the secondary files in the order they are specified on the File Description Specifications form.

4. If a sequentially-processed input file is defined without matching fields, the entire file will be processed in order of priority as described in point 3.

5. If record types without matching fields defined are interspersed in files containing records with matching fields, the records without matching fields are processed as they are encountered, and according to the priority described in point 3.

6. The matching record indicator (MR) is set ON only when the record from the primary file matches one or more of the records from the secondary files. It is not set on when secondary file records match without matching the primary file record.

7. The matching record indicator is turned off after the last record of a matched group has been processed, including total calculations and output for that group. This matched group includes all primary and secondary field records which have identical match-field contents.

Matching Record Indicator (MR)

The matching field entries of M1-M9 have an associated internal indicator MR (Matching Record). This indicator, which is similar to a resulting indicator, is used to condition functions specified on the Calculation and Output-Format Specifications forms.

The MR indicator is turned on when the matching field of a record from a secondary file matches the matching field of a record from the primary file. The comparison is made on all fields (M1-M9) at the same time. If the M1 fields match, but the M2 fields do not, the MR indicator will be turned off. The MR indicator is turned on before the first record of the matched primary file is processed. The MR indicator remains on during the processing of all following primary and secondary file records that contain the same matching field.

The MR indicator is turned off when all total calculations and output are completed for the last record of the matching group.

Figure 113 illustrates when the MR indicator is turned on and off. This example shows a simple block diagram of the object program logic to illustrate logic flow. Two input files are used, and the record selected for processing in each program cycle is shown. The status of indicators is shown by vertical bars along the right side of each step of the figure. A solid bar is used to denote the indicator ON, and a dotted line indicates OFF. Three indicators, one for each file, are used to show which file is being processed.

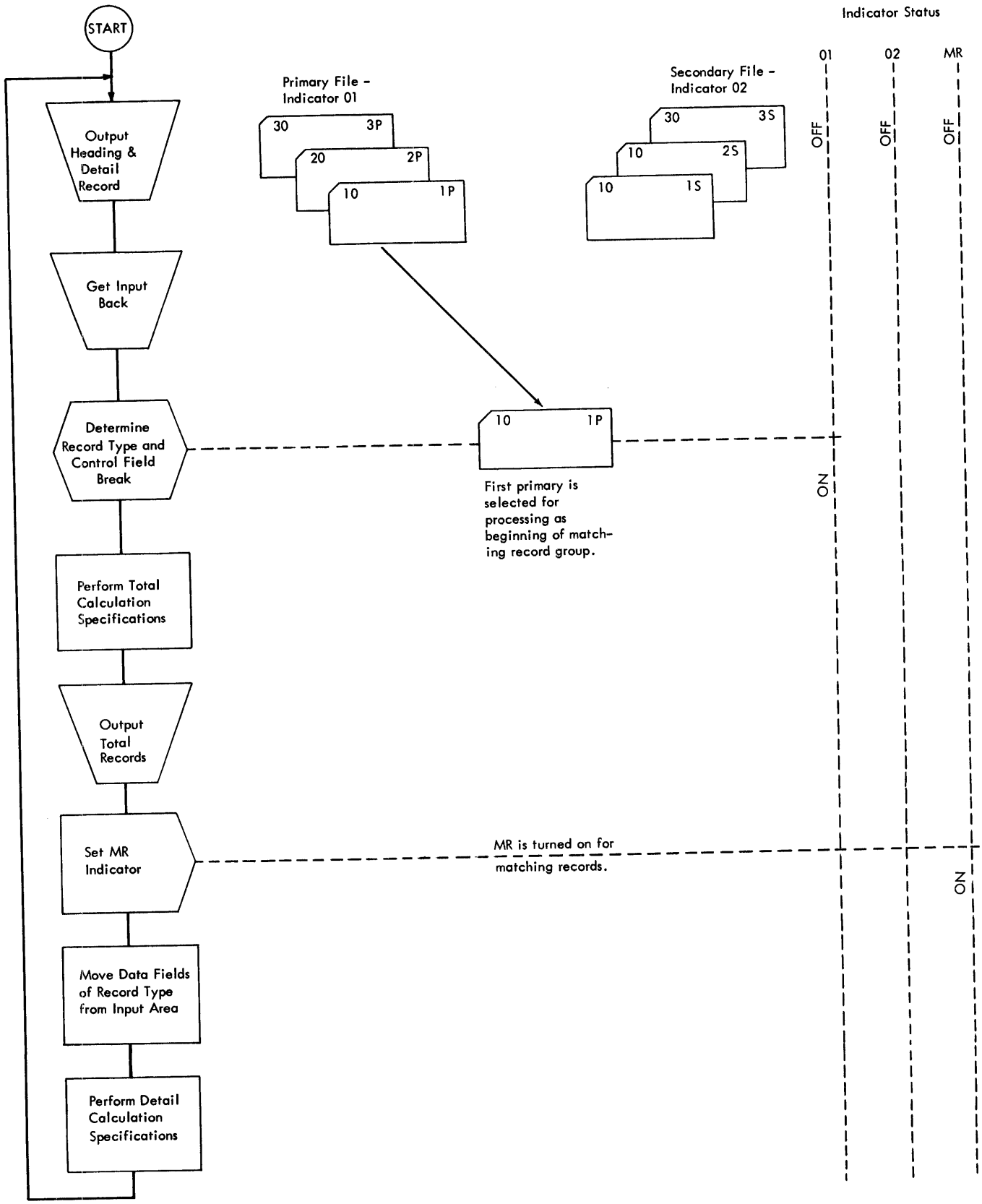


Figure 113. MR Indicator Setting (Part 1)

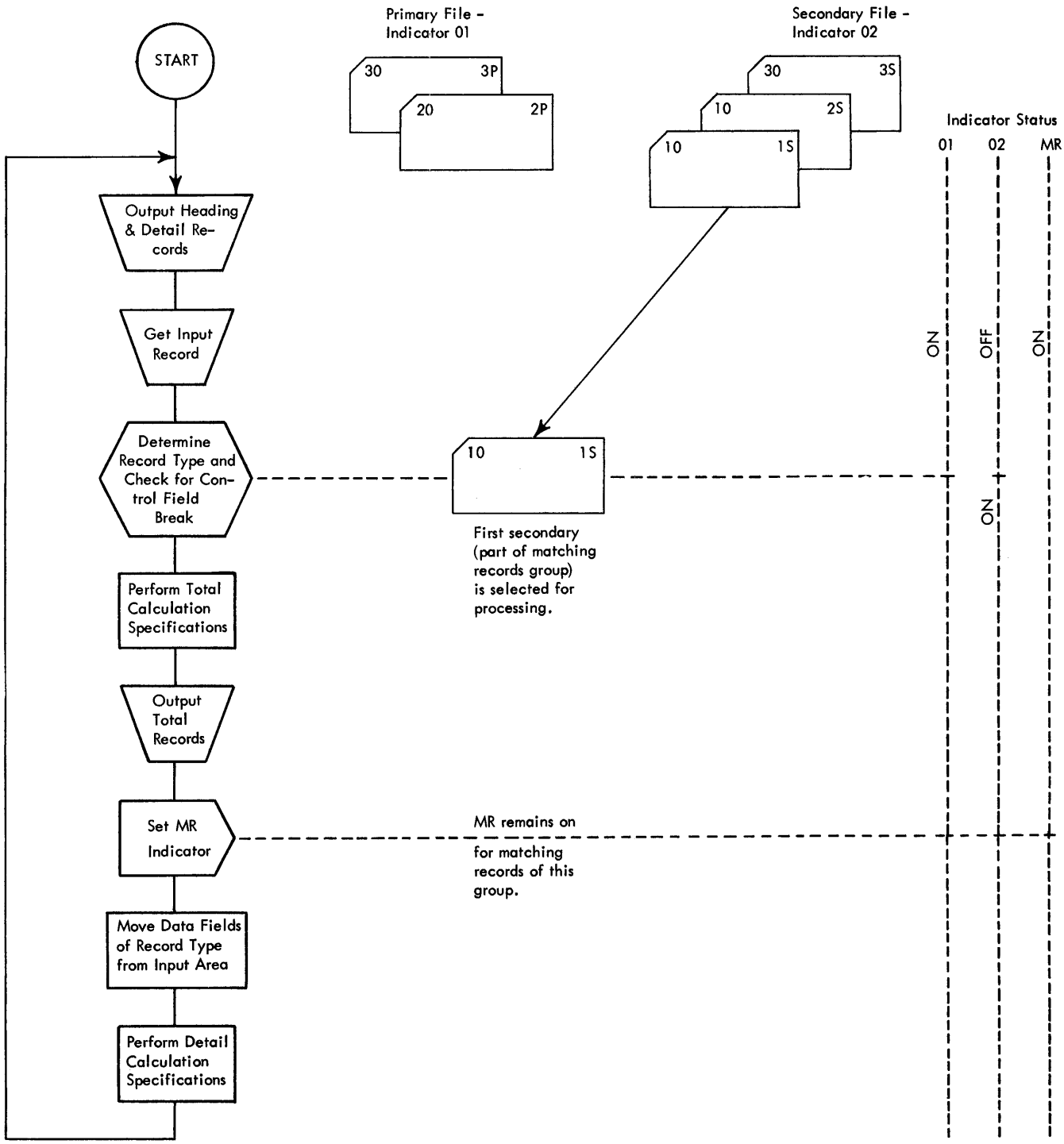


Figure 113. MR Indicator Setting (Part 2)

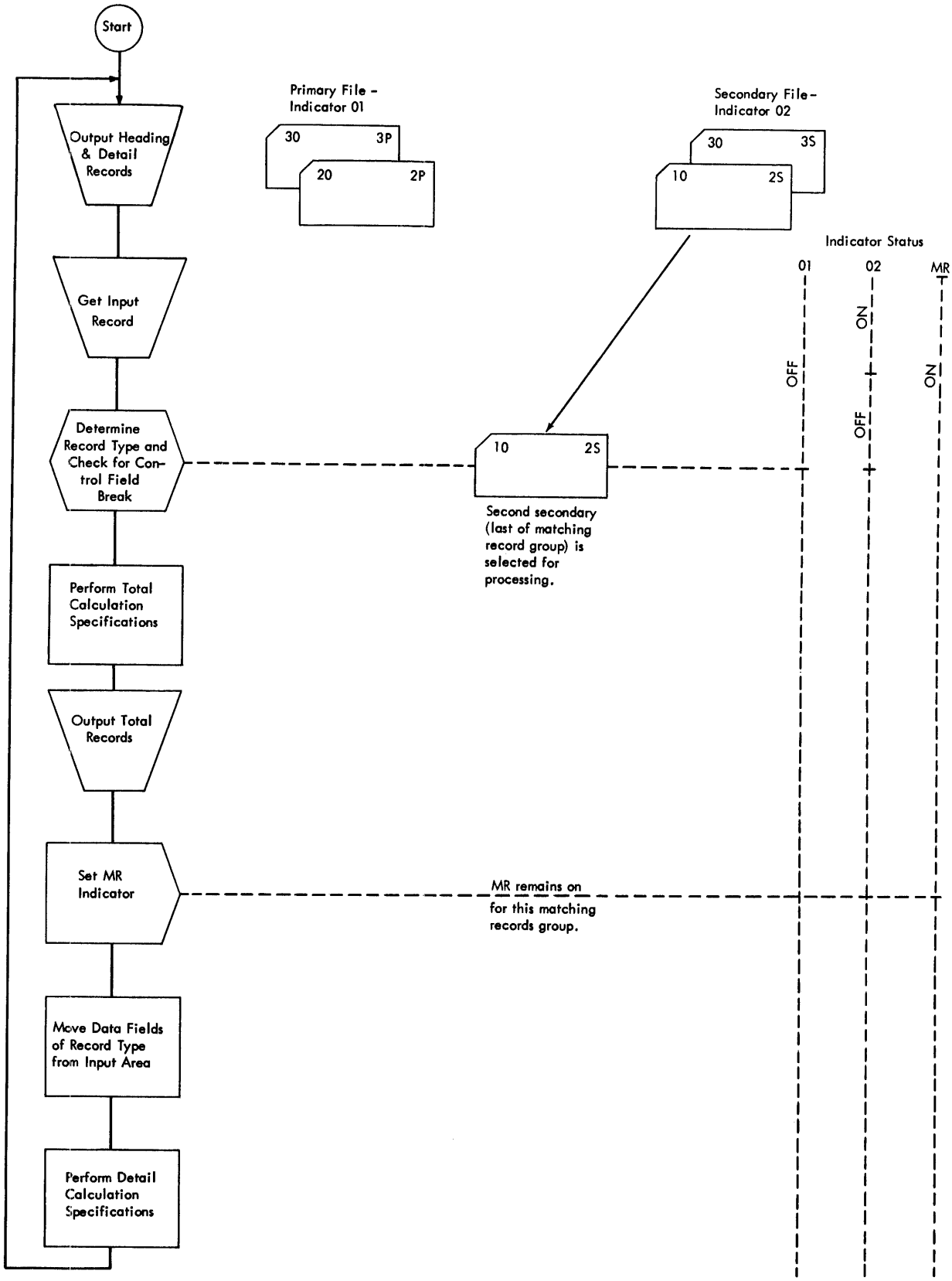


Figure 113. MR Indicator Setting (Part 3)

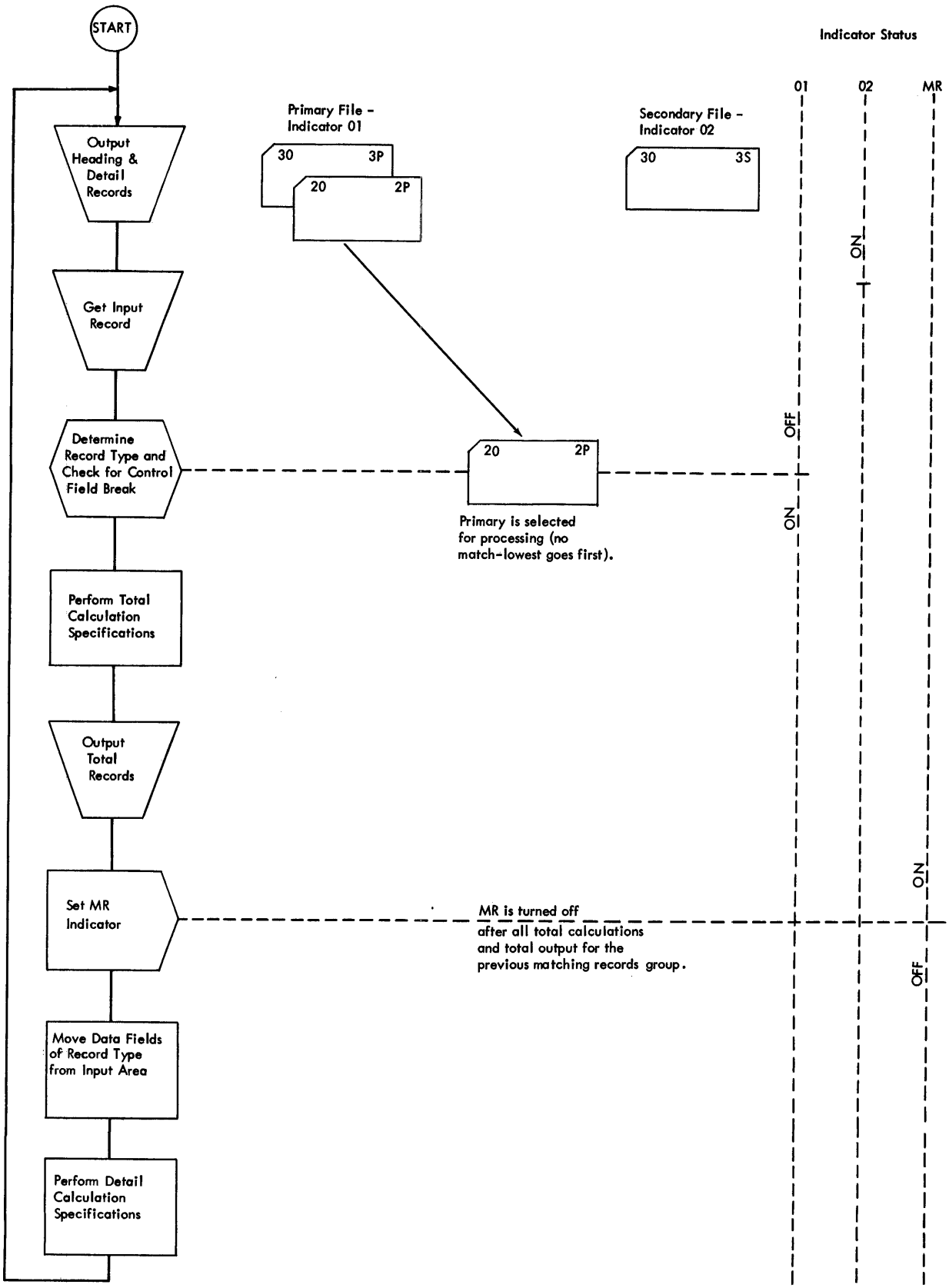


Figure 113. MR Indicator Setting (Part 4)

The MR indicator should not be turned on or off by the operations SETON and SETOF.

All record types do not require processing by the Matching record technique. When record types with nc matching fields are specified on the Input Specifications form, the matching fields specification is left blank. This indicates to the program that these record types do nct need to be checked for a matching field. They are processed immediately as they are read. The MR indicator is turned off during the processing of these record types.

Use of the MR Indicator: As specified on the calculations form in figure 110, whenever the MR indicator is on, RATE from the master file record will be multiplied by FLDA from the record in the detail labor file. If the records dc not match, indicator 16 is set off. In this case, all subsequent processing conditioned by indicator 16 is suppressed. The MR indicator could be used to condition calculation specifications to prevent calculaticns upon an unmatched detail or master record. It could also be used in the output specifications to select unmatched detail cards or unmatched primary cards.

Matching Fields and Control Fields

The matching record technique has no direct relaticnship with control level specification. These are independent functions in RPG. However, it is frequently desired to specify both match fields and control fields within the same record types.

The RPG object program uses the matching field specifications tc determine the crder of processing records when using the same field as a control field and a match field during processing of multiple input files. If it is desired to punch a total or summary card after each set of control groups has been processed (ccntrol break), it is required to specify the same control field for both the primary and secondary file.

If the same control fields are specified in both files, these files are processed in the same manner as a single file program. A control break occurs only once for each control group (either in the primary or the secondary file).

Exit to a Translate Subroutine

If the sequence of the matching fields is not EBCDIC collating sequence, 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 desired.

An entry in the RPG Control Card is all that is required to cause the RPG program to branch to the subrcutine. 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 ALTSE. The address of the hold area containing the matching fields to be translated will be contained in index register 1. This address will be the control word of the field. If index registers 2 and 3 are used in the subroutine, they must be saved and restored. The subroutine must place the translated fields back into the matching-field holding area before it returns control to RPG.

The original data fields (to be used for controlling, printing, arithmetic calculations, etc.) are not translated by the subroutine.

Examples

Figure 114 illustrates the order of processing for a primary and a secondary file.

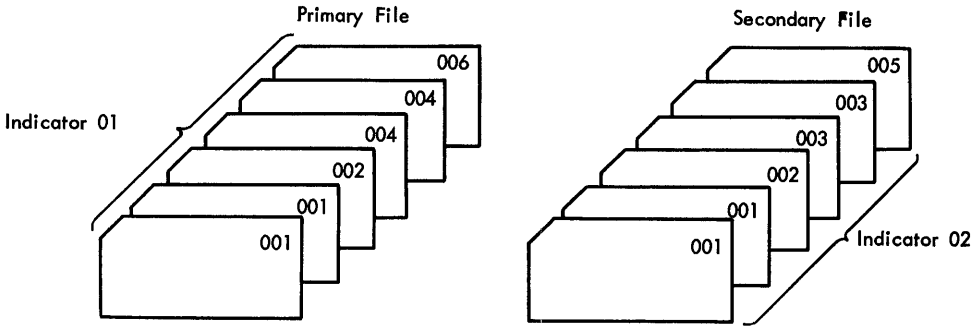


Figure 114. Order of Processing Matched Records

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 processing order for Figure 114 is as follows:

Assume that the same field is used as a matching field and a control field in both files as described in Figure 114. In this case a control break occurs before the processing of the first record and after processing the 4th, 6th, 8th, 10th, and 11th record. Assume that the same field is specified as a matching and a control field in the secondary file, but only as a matching field in the primary file. In this case a control break occurs after the detail time processing of the 2nd, 5th, 6th, and 10th records.

Using Matching Record Techniques Without Matching Fields

If in a multifile program, records that are specified without matching fields are interspersed with records that are specified with matching fields, the following occurs:

1. Any records without matching fields are processed as they are encountered. At the beginning of processing, the object program processed only initial records without matching fields specified, until the first record within each file with matching fields is encountered. The processing sequence is primary first, followed by secondary files in the order they are specified on the File Description Specifications. If there is a file with no records for which matching fields are specified, the entire file will be processed before any records of the subsequent file(s) are processed.

2. During the processing of the record for which no matching fields are specified, the MR is turned off regardless of what its previous setting was.

If only one file contains records with matching fields, this specification is used for sequence checking only. All primary and secondary files are processed in the order they are specified on the File Description Specifications form.

Note: It is invalid to have two sequentially processed files and only one which has matching fields specified. If two files have matching records specified, a third file can have no matching records.

CHAINING

Three methods of chaining exist. The CHAIN operation on the Calculation Specifications form, however, offers the most flexibility. For this reason, it will be stressed and the other methods will be discussed later.

Two or more input files are always involved in chaining. One file normally contains a field which can be used to point to the particular record in the chained file. The field which is being used to point to the chained file is called a chaining field.

In the most typical application, a field in a transaction record is used to chain to the appropriate master record. The transaction records may be a card file or any type of disk file. Typically, the transaction file is in the sequence in which the transactions have occurred and the master file is in some logical sequence (e.g., Customer Number or Item Number). Chaining allows the user to process whatever master record is affected without first sorting the transactions into the same sequence of the master file. This is known as random processing or chaining (as opposed to sequentially processing two files, described in Matching Records).

While this type of processing is the classical example of chaining, there are many variations which can be used depending upon the application and other criteria.

For those users who are not familiar with chaining, the next section discusses the most typical applicaion, that of chaining to an index sequential fileis fully understood, variations and other techniques will be observed.

The example of the library used in Disk Storage Concepts illustrates the concept of chaining. To find a particular book, the card catalog is searched (processed) for a record of the desired book. The card catalog record points (chains) to the shelf location of the book by use of a catalog number (key).

With indexedsequential organizatcn, every record of a file has a key. These keys are indexed in the file's Master Index. Thus any record can be accessed through its key. Chaining is the use of these keys to link from a chaining field to the record of another which has that key. Chaining is done either by use of the CHAIN operation in Calculation Specifications or the alternate method of chaining codes (C1-C3) on the Calculation Specifications.

Randomly Processing Multiple Input Files

To understand chaining, assume that a sequential input file, as shown in Figure 115, contains information about transactions made with several customers. The card file contains the customer's number, but does not contain his name, address, or balance. Another file called the master customer file contains this additional information about each customer. The RPG program can 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 key in each disk record contains the customer's number. The field in the transaction-file records which contains the customer's number can be used to chain the files together. The object program takes the customer's number from the transaction file and locates the record with the same key in the master customer file. The additional information, such as the customer's name, address, balance, etc., is associated with each 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 mas-

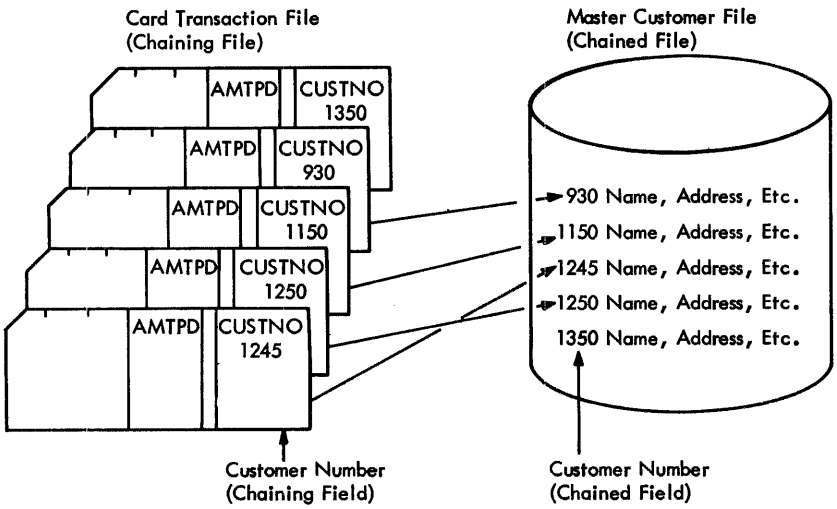


Figure 115. Example of Using Chaining to Process an Indexed Sequential File

ter customer file is the chained file because it is being linked from the transaction file.

Figure 116 illustrates the coding required for a typical chaining operation. In this case, a card file CARDIN contains transactions which are to be updated against an indexed sequential file called MASTCUST.

The File Description Specification Form shows the entries required for the file CARDIN. It is an input file and is also the primary file for this job. No entry is required to denote that a field in this will be used as a chaining field.

The MASTCUST file on line 02 describes the indexed sequential file. It is a file which is to be updated (U in column 15) and chained (C in column 16). Because it is a chained file, RPG knows that the programmer will supply a chaining field when the file is needed. If a chained file is described, the mode of processing must be random (R in column 28). The entries in columns 29-38 are required for an indexed sequential File. The length of the key field (6 in columns 29-30) may vary with each file, but the key field starting location (columns 35-38) is always 1 in 1130 RPG. The entries of K and I (columns 31-32) are mandatory to describe MASTCUST as an Indexed Sequential File. A printer is also used in this job and is discribed.

The Input Specification Form describes the transaction record in CARDIN. In this case it has only two fields. CUST will be used as the chaining field and AMTPD will be used to update the appropriate customer record.

The MASTCUST file has two types of records and is typical of Indexed Sequential files. Those records with a 1 in position seven are active records and can be updated. Records with a 2 in position seven are customer records which are no longer active. It is possible (through some error) to receive a payment for a customer who has been deleted. The programmer will normally put in some coding (as is shown in this example) to handle this condition.

The field BALANC in the MASTCUST record is the field which is to be updated.

The Calculation Specifications Form shows the CHAIN operation using the field CUST as the chaining field. MASTCUST is described as the chained file. Indicator 20 will be turned on if no record is found in MASTCUST for this number.

Line 02 showes the subtraction to get the new balance field. This is only done when indicator 02 is on which means an active record was found in MASTCUST.

The Output Format Specifications Form shows the coding necessary to update MASTCUST. The file is up dated only when a record was found and it was an active record. It is only necessary to output the fields which have actually changed. In this case only BALANC is mentioned. The other fields for this record will remain unchanged.

The printer file is used for error conditions in this job. The first error condition shown is the case of a transaction affecting a deleted customer record. This was noted by indicator 03. Lines 03-05 provide a message for this condition.

The second error condition possible is when the transaction record contains a customer number and there is no corresponding customer on the file. This error is detected by indicator 20 being set on by the CHAIN operation. An appropriate message is then printed.

The programmer does not have to program for the two error conditions. If the deleted customer record was not coded on lines 12-13 of the Input Specifications Form and a deleted record was found, the RPG program will halt with an operator message that describes an "UNDEFINED RECORD." The operator can then choose to bypass the record.

The error coding concerning a "NOT FOUND" record may also be eliminated. If a record is "NOT FOUND" and there are no indicators in columns 54-57 of the CHAIN operation, the RPG program will halt with a message to the operator describing the condition. The operator can then choose to bypass the processing of the entire transaction.

It is normally preferable for the programmer to code for possible error conditions as this minimizes operator intervention.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3347
Printed in U. S. A.

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	I/O/U/C 15	P/S/C/R/T 16	E 17	A/D 18	F/V 19	Block Length 20–23	Record Length 24–27	L/R 28	Length of Key Field or of Record Address Field 29–30	K/I 31	Record Address Type 32	Type of File Organization 33–34	Overflow Indicator 35	Key Field Starting Location 36–38	Extension Code E/L 39	Device 40–46	Symbolic Device 47–52	Labels (S, N, or E) 53	Name of Label Exit 54–59	Extent Exit for DAM 60–65	A 66	No. Tracks for Cylinder Overflows 67–68	No. of Extents 69	Tape Rewind N/U 70
01	F	CARDIN	I	P			F		80									READØL								
02	F	MASTCUSTUC					F		1ØØR		6	K	I				L	DISK								
03	F	PRINTER	O				F		120									PRINTER								
04	F																									

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3350
Printed in U.S.A.

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	Sequence 15–16	Number (1-N) 17	Option (O) 18	Resulting Indicator 19–20	Record Identification Codes 1: Position 21–24	1: Not (N) 25	1: C/Z/D 26	1: Character 27	2: Position 28–31	2: Not (N) 32	2: C/Z/D 33	2: Character 34	3: Position 35–38	3: Not (N) 39	3: C/Z/D 40	3: Character 41	Stacker Select 42	Packed (P) 43	Field Location From 44–47	Field Location To 48–51	Decimal Positions 52	Field Name 53–58	Control Level (L1-L9) 59–60	Matching Fields or Chaining Fields 61–62	Field-Record Relation 63–64	Field Indicators Plus 65–66	Field Indicators Minus 67–68	Field Indicators Zero or Blank 69–70	Sterling Sign Position 71–74
01	I	CARDIN	AA			Ø1															75	8Ø		CUST							
02	I																				11	2Ø		AMTPD							
03	I																														
04	I	MASTCUSTBB				Ø2		7	C	1																					
05	I																				1	6		CUSTNO							
06	I																				8	2Ø		NAME							
07	I																				21	3Ø	2	BALANC							
08	I																				37	5Ø		ADDR							
09	I																				51	6Ø		CITY							
10	I																				61	65		STATE							
11	I																				66	8Ø		SHIPPR							
12	I		CC			Ø3		7	C	2																					
13	I																				1	6		CUSTNO							
14	I																														

Figure 116. Example of Specifying File Chaining (Part 1)

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And / And — Not 9	10	11	12	13	14	Not 15	16	17	Factor 1 18-27	Operation 28-32	Factor 2 33-42	Result Field 43-48	Field Length 49-51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus 54 55	Minus 56 57	Zero or Blank 58 59	Compare: High 1>2 60 61	Low 1<2 62 63	Equal 1=2 64 65	Comments 66-74
0 1 Ø	C											CUST		CHAINMASTCUST								2Ø	2Ø		
0 2 Ø	C			Ø	2							BALANC		SUB AMTPD		BALANC									
0 3	C																								

IBM

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 3 4 5	Form Type 6	Filename 7-14	Type (H/D/T) 15	Stacker Select 16	Space: Before 17	After 18	Skip: Before 19 20	After 21 22	Output Indicators: And / And — Not 23	24	25	26	27	28	Not 29	30	31	Field Name 32-37	Zero Suppress (Z) 38	Blank After (B) 39	End Position in Output Record 40-43	Packed Field (P) 44	Constant or Edit Word 45-70	Sterling Sign Position 71-74
0 1 Ø	O	MASTCUSTD							Ø	1		Ø	2												
0 2	O																	BALANC			3Ø				
0 3	O	PRINTER	D			2			Ø	1		Ø	3												
0 4	O																	CUST			1Ø				
0 5	O																				3Ø		'DELETED CUSTOMER'		
0 6	O		D			2			Ø	1		2	Ø												
0 7	O																	CUST			1Ø				
0 8	O																				3Ø		'NOT FOUND'		
0 9	O																								

Figure 116. Example of Specifying File Chaining (Part 2)

Figure 117 shows an example of chaining to two files from the same transaction.

The File Description entries are very similar to Figure 116 except that two indexed sequential files are being processed randomly.

The Input Specifications show the transaction record with the two fields that will be used in chaining. The two indexed sequential files both have active and deletion records as prescribed in the previous example.

The Calculation Specifications show the two chaining operations. Indicators 20 and 21 are used to denote the NOT FOUND condition that was described in the previous example.

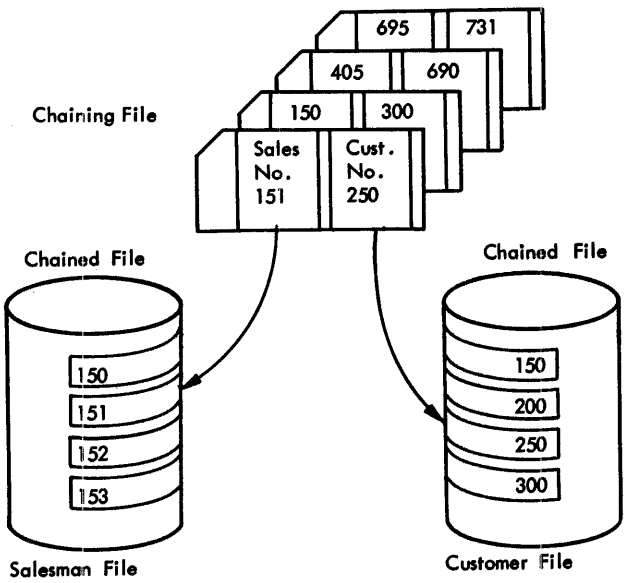


Figure 117. Chaining to Two Files (Part 1)

The Output Specifications show the valid and invalid conditions. The valid condition exists when indicators 02 and 04 are on to denote that active records have been found. Lines 01-05 print this condition.

An invalid condition is noted by the lack of either indicator 02 or 04 being on. Therefore, an OR line has been coded. A description of the error condition is printed in lines 09-10, 12-13, similar to that described in the previous example.

In Figure 118, the card file is chained to the customer file. Within each customer record is a field which contains an account number. The account number is in turn used to chain to the account file.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date _____

Program _____

Programmer _____

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

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	Mode of Processing	Length of Key Field or of Record Address Field	K/1	Record Address Type	I/D/T	Type of File Organization	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
01	F	CARD	I	P			F		80											READ42										
02	F	SLSMAN	I	C			F		60	R		3	K	I				1		DISK										
03	F	CUSTOMER	I	C			F		100	R		8	K	I				1		DISK										
04	F	PRINTER	O				F		120											PRINTER										
05	F																													

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

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

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	CARD	AA			01																									
02	I																				15	17		SALESM							
03	I																				31	38		CUST							
04	I	SLSMAN	BB			02		4	C	1																					
05	I		OR			03		4	C	2																					
06	I																				5	20		SLSNAM							
07	I	CUSTOMER	CC			04		9	C	1																					
08	I		OR			05		9	C	2																					
09	I																				21	40		CUSNAM							
10	I																														

Figure 117. Chaining to Two Files (Part 2)

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And / And — Not 9	Not 10	Not 11	Not 12	Not 13	Not 14	Not 15	Not 16	17	Factor 1 18-27	Operation 28-32	Factor 2 33-42	Result Field 43-48	Field Length 49-51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators: Plus 54-55	Minus 56-57	Zero or Blank 58-59	Compare: High 1>2 54-55	Low 1<2 56-57	Equal 1=2 58-59	Comments 60-74
0 1	Ø	C										SALESM	CHAIN	SLSMAN									20	20	
0 2	Ø	C										CUST	CHAIN	CUSTOMER									21	21	
0 3		C																							

IBM

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 3 4 5	Form Type 6	Filename 7-14	Type (H/D/T) 15	Stacker Select 16	Space: Before 17	Space: After 18	Skip: Before 19-20	Skip: After 21-22	Output Indicators: And / And — Not 23	Not 24	Not 25	Not 26	Not 27	Not 28	Not 29	Not 30	Not 31	Field Name 32-37	Zero Suppress (Z) 38	Blank After (B) 39	End Position in Output Record 40-43	Packed Field (P) 44	Constant or Edit Word 45-70	Sterling Sign Position 71-74
0 1	Ø	O PRINTER	D		2					Ø	1	Ø	2	Ø	4									
0 2	Ø	O																SALESM			3			
0 3	Ø	O																SLSNAM			25			
0 4	Ø	O																CUST			40			
0 5	Ø	O																CUSNAM			70			
0 6	Ø	O	D		2					Ø	1	N	Ø	2										
0 7	Ø	O	OR							Ø	1	N	Ø	4										
0 8	Ø	O																SALESM			3			
0 9	Ø	O													Ø	3					25		'DELETION'	
1 0	Ø	O													2	Ø					25		'NOT FOUND'	
1 1	Ø	O																CUST			40			
1 2	Ø	O													Ø	5					70		'DELETION'	
1 3	Ø	O													2	1					70		'NOT FOUND'	
1 4	Ø	O																			90		'***'	
1 5		O																						

Figure 117. Chaining to Two Files (Part 3)

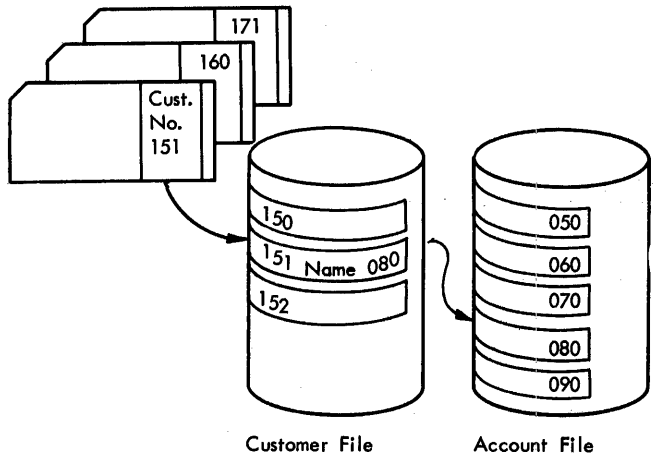


Figure 118. Using a Chained File as a Chaining File

Chaining to a Sequential Disk File

Chaining can also be used to select a record from a sequential disk file. It differs from retrieving from an indexed sequential file in that a relative record number must be used instead of a key field. A relative record number means that the user must know which record he wants to process by describing it in terms of its position within the file. A request for number 151 will produce the 151st record in the file.

This technique has advantages, but is generally limited in what applications can be conveniently processed in this manner. The following describes some of the normal uses for this approach:

1. The actual code number (e.g., customer number) is used as the relative record number. (This depends greatly on the coding structure used.)

2. A mathematical formula is used to convert the code number to a relative record number. (This method requires some randomizing technique and the user must be able to account for two or more code numbers that would randomize to the same number.)

3. A table lookup approach may be used to locate the assigned relative record number. This method may require a large table in core.

Figure 118.1 shows the coding required for converting an item number to a relative record number by means of a randomizing technique.

The File Description Specifications describe the card file CARDIN which contains the item number. The MASTITEM file is a sequentially organized file (blank in column 32) and a chained file (C in column 16) to be processed randomly (R in column 28). A printer file is also described.

The Input Specifications describe the field ITEM within the CARDIN file. The MASTITEM file contains an item number description and the relative record number (NEXT) of the next record in a series of records whose item numbers randomize to the same number. (These are normally called synonyms). The field NEXT is being tested for a zero condition by indicator 22.

The Calculation Specifications show the randomizing formula and the chaining technique. The item number is squared (line 01) and the middle four digits of the result are extracted. This is accomplished by a MOVE and MOVEL operation (lines 02-03). The result (SAVE4) is then tested to see if it exceeds 5000 and if so, 5000 is subtracted. These steps (lines 04-05) limit the size of the file to 5000 records.

Lines 07-10 show the chaining technique. The field SAVE4 is chained to MASTITEM in line 07. Line 08 turns on indicator 32 if the item from the card file is equal to the item in the disk file. If they are not equal, a synonym probably exists. Line 09 moves the next relative record number to be used to SAVE4 if indicators 32 and 22 are off. This denotes that there is a synonym to be tried. Line 10 branches to LOOP to repeat the chaining operation if required.

Either the proper record is found (indicator 32 is on) or an invalid item number exists.

The Output Specifications show the valid condition in lines 01-03. The invalid condition is handled in lines 04-06.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
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	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
01	F	CARDIN	I	P			F		80							READ42									
02	F	MASTITEMIC					F		150	R						DISK									
03	F	PRINTER O					F		120							PRINTER									
04	F																								
05	F																								

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
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-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	CARDIN	AA			01																									
02	I																				1	60	ITEM								
03	I	MASTITEMBB				02																									
04	I																				1	60	ITEMB								
05	I																				7	20	DESCRP								
06	I																				101	1040	NEXT						22		
07	I																														
08	I																														

Figure 118.1 Randomizing Method of Obtaining a Relative Record Number.

IBM

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 3 4 5	Form Type 6	Control Level (L0,L9, LR) 7 8	Indicators: And / And — Not 9	Not 10	11	Not 12	13	Not 14	15	16	17	Factor 1 18–27	Operation 28–32	Factor 2 33–42	Result Field 43–48	Field Length 49 50	Decimal Positions 51	Half Adjust (H) 52	Resulting Indicators: Plus 53	Minus 54	Zero or Blank 55	Compare: High 1>2 56	Low 1<2 57	Equal 1=2 58 59	Comments 60–74
01	C											ITEMA	MULT	ITEMA	SAVE12	12	Ø								
02	C												MOVE	SAVE12	SAVE8		8	Ø							
03	C												MOVEL	SAVE8	SAVE4		4	Ø							
04	C											SAVE4	COMP	5ØØØ								31			
05	C		31									SAVE4	SUB	5ØØØ	SAVE4										
06	C											LOOP	TAG												
07	C											SAVE4	CHAIN	MASTITEM											
08	C											ITEMA	COMP	ITEMB										32	
09	C		N32	N22									MOVE	NEXT	SAVE4										
10	C		N32	N22									GOTO	LOOP											
11	C																								
12	C																								

IBM

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 Identificotion 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	Type (H/D/T) 15	Stacker Select 16	Space: Before 17	Space: After 18	Skip: Before 19–21	Skip: After 22	Output Indicators: And / And — Not 23	Not 24	25	Not 26	27	Not 28	29	30	31	Field Name 32–37	Zero Suppress (Z) 38	Blank After (B) 39	End Position in Output Record 40–43	Packed Field (P) 44	Constant or Edit Word 45–70	Sterling Sign Position 71–74
01	O	PRINTER	D		2				Ø1			32												
02	O																	ITEMA			1Ø			
03	O																	DESCRP			3Ø			
04	O		D		2				Ø1			32												
05	O																	ITEMA			1Ø			
06	O																				5Ø		'NOT FOUND'	
07	O																							
08	O																							

Figure 118.1 Randomizing Method

Use of CHAIN Operation Code

The chaining field may be specified by use of the CHAIN operation in Calculation Specifications. This enables the chaining function to be:

1. Performed with a field whose value is calculated or read through an input file.
2. Performed at either detail or total time in the program cycle.
3. Conditioned on the results of another calculation or test.
4. Performed more than once on one chaining field.
5. Performed more than once on one or more chained files.
6. Up to nine chained files can be in one program.

Refer to CHAIN under Calculation Specification Form for further information.

Application Uses of the CHAIN Operation

The application uses of the CHAIN operation are varied. It will allow processing of basic jobs where a master file is retrieved randomly or complex jobs where the same file is accessed more than once for the same transaction.

The following list attempts to describe some of the functions which may be performed.

1. Simple chaining to randomly processed transactions (See Figure 116).
2. Chaining to two different files from the same transaction (See Figure 117).
3. Chaining to one indexed sequential file and then using a field in that file to chain to a second indexed sequential file (See Figure 118).
4. Chaining to the same file more than once from the same input record .
5. Handling of cards which contain multiple transactions commonly called spread cards.

6. Handling inventory problems which involve substituting items when the original requested item is out of stock.

7. Handling Bill of Material explcsions where one input record causes several chaining requests.

8. Accessing Bill of Material items and printing a list of them from random input.

Use of C1-C3 Codes

This is an alternate method of chaining to the CHAIN operatien on the Calculation Specifications. It can only be used on an indexed sequential file.

Input fields may be defined as chaining fields by entering C1 through C3 on the Input Specifications line (columns 61-62) for that field. Refer to Chaining Fields (Disk Only) under Input Specifications Form.

The C1-C3 codes differ from the CHAIN operation in that C1-C3 applies cnly to input fields, and cause chaining to occur only before detail time in the program cycle.

Note: There is no specific relatienship between levels C1 to C3, other than specifying three possibilities for chaining fields and the order in which chaining will occur.

The same input field may chain, with C1-C3 codes, to more than one file if it is defined more than once en the Input Specification form - with a different chaining code every time. There is nc restriction to the number of times that a chaining field defined with the CHAIN operation may be used.

Two chaining files may chain to the same file, using C1-C3, if the same chaining code is specified for each file on the Input Specifications fcrm.

Figure 118.2 is an example of chaining using C1-C3. The File Description entry for the chaining file CARDIN has an E in the Extension code (column 39) denoting that the file will be further described on the Extension Specifications. The other File Description entries are similar to those of previous examples of chaining to an indexed sequential file.

The Extension Specifications have an entry of AA in Record Sequence of the chaining file (columns 7-8). This corresponds with columns 15-16 of the chaining file CARDIN on the input specifications. An entry of C1 is shown in the Number of the chaining field (columns 9-10). This corresponds with the entry in columns 61-62 of the Input Specifications for the chaining field. CARDIN is written in From Filename (columns 11-18) to denote the name of the chaining file. MASICUST is written in To Filename (columns 19-26) to denote the name of the chained file.

The Input Specifications show the chaining field CUST in the file CARDIN. The C1 in Chaining Fields (columns 61-62) denotes CUST as the chaining field. MASTCUST has both active and inactive records. The active records have a 1 in position 7 and the deletions have a 2 in position 7.

No Calculation Specifications are required for this job.

The Output Specifications show the valid conditions in lines 01-03 and the invalid conditions in lines 04-07. The valid condition is denoted by indicator 02 being on. The invalid condition has indicator 02 off. If indicator 03 is on, a deleted master record was found.

Using C1-C3, there is no possibility for the programmer to prevent a system halt in the event that a no record found condition exists. The system will come to a halt and the operator will be given the choice of bypassing the input record or ending the job.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
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	File Designation	End of File	Sequence	File Format	Block Length	Record Length	L/R	Mode of Processing	Length of Key Field or of Record Address Field	Record Address Type	Type of File Organization	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind
			I/O/U/C	P/S/C/R/T	E	A/D	F/V				K/I	I/D/T											A			N/U
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
01	F	CARDIN	I	P			F		80								E	READ42								
02	F	MASTCUSTIC					F		100	R	6	KI					L	DISK								
03	F	PRINTER	O				F		120									PRINTER								
04	F																									
05	F																									

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

Form X24-3348
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	Number of the Chaining Field	From Filename	To Filename	Table Name	Number of Table Entries Per Record	Number of Table Entries Per Table	Length of Table Entry	Packed (P)	Decimal Pos.	Sequence (A/D)	Table Name (Alternating Table)	Length of Table Entry	Packed(P)	Decimal Pos.	Sequence (A/D)	Comments
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
01	E	AA	C1	CARDIN	MASTCUST													
02	E																	
03	E																	

Figure 118.2 Chaining Using C1-C3

Chaining with an RA File

This is a seldom used function and is supported as a modification of processing between limits. Only an indexed sequential file can be processed in this manner. The coding is similar to processing within limits except that the indexed sequential file is processed randomly (R in column 28). The record address fields may be single or multiple entries per record. If there are multiple entries, they must be adjacent. The next record is read from an RA file when a blank is found. The RA file may be on cards or disk.

Split Chaining Fields

C1-C3 Codes: Several fields that are not in adjcining positicns 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 fcrm, and the fields are then used as one chaining field. The first field defined on the input fcrm is placed in the high-order position, and the last field is placed in the low-crder position. The fields are placed in the same sequence as they are defined on the Input Specifications form.

CHAIN: Several fields can be collected into one chaining field through the use of move operations, and the result field used with the CHAIN operation. Thus, the order in which the several fields are collected is completely under the control of the programmer.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
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-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	CARDIN	AA			01																									
02	I																														
03	I	MASTCUSTBB				02			7	C1											21	26		CUST		C1					
04	I	OR				03			7	C2																					
05	I																				31	40		NAME							
06	I																														
07	I																														

IBM

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	Type (H/D/T)	Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: Not	And: 24	And: Not 25	And: 26	And: Not 27	And: 28	And: Not 29	And: 30	Not 31	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	PRINTER	D		2					01		02												
02	O																CUST			10			
03	O																NAME			40			
04	O		D		2					01		03												
05	O																CUST			10			
06	O																						
07	O																			50		'DELETION'	
08	O																						

Figure 118.2. Chaining Using C1-C3

An example of this is shown in Figure 118.3. In this case the chaining field is made up of two fields from the same card, but they are in different locations. The Input Specifications show their locations. The Calculation Specifications show how they are placed together to form one field.

The field SAVE will be the chaining field. The field HIGH is moved to the high order (leftmost) positions of the field by a MOVEL operation. The field LOW is moved to the low order (rightmost) positions of the field by a MOVE operation. Then the CHAIN operation is used to retrieve the appropriate record.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Filename 7–14	Sequence 15 16	Number (1-N) 17	Option (O) 18	Resulting Indicator 19 20	Record Identification Codes 1: Position 21–24	1: Not (N) 25	1: C/Z/D 26	1: Character 27	2: Position 28–31	2: Not (N) 32	2: C/Z/D 33	2: Character 34	3: Position 35–38	3: Not (N) 39	3: C/Z/D 40	3: Character 41	Stacker Select 42	Packed (P) 43	Field Location: From 44–47	Field Location: To 48–51	Decimal Positions 52	Field Name 53–58	Control Level (L1-L9) 59 60	Matching Fields or Chaining Fields 61 62	Field-Record Relation 63 64	Field Indicators: Plus 65 66	Field Indicators: Minus 67 68	Field Indicators: Zero or Blank 69 70	Sterling Sign Position 71–74
01	I	CARD	AA			01																									
02	I																				1	2		HIGH							
03	I																				21	25		LOW							
04	I																														

IBM

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 3 4 5	Form Type 6	Control Level (L0-L9, LR) 7 8	Indicators: And Not 9–17	Factor 1 18–27	Operation 28–32	Factor 2 33–42	Result Field 43–48	Field Length 49–50	Decimal Positions 51	Half Adjust (H) 52	Resulting Indicators: Plus 53	Resulting Indicators: Minus 54	Resulting Indicators: Zero or Blank 55	Compare: High 1>2 56	Compare: Low 1<2 57	Compare: Equal 1=2 58	Comments 59–74
01	C				MOVELHIGH		SAVE	7									
02	C				MOVE LOW		SAVE										
03	C			SAVE	CHAINCUSTOMER									2020			
04	C																

Figure 118.3 Collected Chaining Field.

Processing Between Limits

Only an indexed sequentially organized file can be processed between limits. This is accomplished by using an RA file to specify the limits. The RA file is never defined on the Input Specifications.

The file being limited is processed sequentially. It can be used in a single input program or with multiple inputs and matching records.

The limits provided do not have to be actual key numbers to be found on the indexed sequential file. They can be any numbers and can be lower or higher than the lowest or highest key fields on the file.

It is possible to process limits which overlap as long as the matching records are not used.

Example: In this example a disk file contains customer records. The file is sequenced on customer number (5 digits). The first digit of customer number denotes the type of account (e.g., All customer numbers between 10000 and 19999 are a type 1 account).

A report is desired which prints customer types 1,2,4,5,6 and 9. This will be accomplished by describing a record address file of the appropriate limits. This is shown in Figure 118.4. The beginning and ending customer number of each record must be specified in columns 1-5 and 6-10 respectively.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction	Graphic							
	Punch							

Page [ ][ ]

Program Identification [ ][ ][ ][ ][ ][ ][ ][ ]

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	Length of Key Field or of Record Address Field	K/1	I/D/T	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
01	F	RAFLIMITIR					F		80		5					E	READ42									
02	F	MASTCUSTIP					F		100	L		KI			1		DISK									
03	F	PRINTER	O						120				OF				PRINTER									
04	F																									
05	F																									

IBM

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 [ ][ ]

Program Identification [ ][ ][ ][ ][ ][ ][ ][ ]

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And	Not	And	Not	And	Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Compare: High 1 > 2	Low 1 < 2	Equal 1 = 2	Plus	Minus	Zero or Blank	Comments
01	C			01					AMT	ADD	TOTAL	TOTAL	9	2								
02	C																					
03	C																					

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction	Graphic							
	Punch							

Page [ ][ ]

Program Identification [ ][ ][ ][ ][ ][ ][ ][ ]

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes: 1 Position	1 Not (N)	1 C/Z/D	1 Character	2 Position	2 Not (N)	2 C/Z/D	2 Character	3 Position	3 Not (N)	3 C/Z/D	3 Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Minus	Zero or Blank	Sterling Sign Position
01	I	MASTCUSTAA				01	6	C	1																						
02	I	OR				02	6	C	2																						
03	I																				1	5	CUST								
04	I																				7	26	NAME								
05	I																				71	72	AMT								
06	I																														

Figure 118.4 Processing Between Limits

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE EXTENSION SPECIFICATIONS

IBM System/360

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

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic | Punch

Page 1 2

Program Identification 75 76 77 78 79 80

Line 3 4 5	Form Type 6	Record Sequence of the Chaining File: Number of the Chaining Field 7 8 9 10	From Filename 11 12 13 14 15 16 17 18	To Filename 19 20 21 22 23 24 25 26	Table Name 27 28 29 30 31 32	Number of Table Entries Per Record: Number of Table Entries Per Table 33 34 35	Length of Table Entry 36 37 38 39 40 41 42	Packed (P) 43	Decimal Pos. 44	Sequence (A/D) 45	Table Name (Alternating Table) 46 47 48 49 50 51	Length of Table Entry 52 53 54	Packed(P) 55	Decimal Pos. 56	Sequence (A/D) 57	Comments 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74
0 1	E		RAFLIMITMASTCUST													
0 2	E															
0 3	E															

IBM

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 3 4 5	Form Type 6	Filename 7 8 9 10 11 12 13 14	Type (H/D/T) 15	Stacker Select 16	Space: Before 17	Space: After 18	Skip: Before 19 20	Skip: After 21 22	Output Indicators: And Not 23	And 24 25	And Not 26	And 27 28	And Not 29 30	And 31	Field Name 32 33 34 35 36 37	Zero Suppress (Z) 38	Blank After (B) 39	End Position in Output Record 40 41 42 43	Packed Field (P) 44	Constant or Edit Word 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	Sterling Sign Position 71 72 73 74
0 1	O	PRINTER	H			3Ø	1			1P											
0 2	O			OR						OF											
0 3	O																	2Ø		'CUSTOMER REPORT'	
0 4	O																	45		'TYPES 1-2-4-5-6-9'	
0 5	O		D		2					Ø1											
0 6	O														CUST			5			
0 7	O														NAME			3Ø			
0 8	O														AMT		B	45			
0 9	O		T		2					LR											
1 0	O														TOTAL	B		45			
1 1	O																				

Figure 118.4 Processing Between Limits

The RA file is described on the File Description Specifications with a 5 in columns 29-30. This denotes the length of the Record Address Field and it must be the same length as the key field of the file it is limiting. The E in column 39 describes the need of the Extension Specifications.

The L in column 28 of the disk file denotes that this file will be processed by limits.

The Extension Specifications describe the fact that the RA file is being used to limit the indexed sequential file (only an indexed sequential file can be processed by limits).

The Input Specificaticns describe the fields needed in the Customer Record. The record identifying indicator 02 is used to avoid deleted records in the customer file that should not be processed.

The Calculation and Output Specifications describe the needed results.

SUMMARY OF MULTIPLE FILE PROCESSING

The following methods are available to RPG for processing one file against ancther:

1. Sequentially, using the matching record technique.
2. Sequentially between limits, using an RA file.
3. Randomly, using the chaining technique.

Figure 119 shows these processing possibilities.

MODE OF PROCESSING	TYPE OF FILE ORGANIZATION		
	CARD FILES (SEQUENTIAL)	DISK FILES (SEQUENTIAL)	DISK FILES (INDEXED SEQUENTIAL)
Sequential - As a single file or multiple files using matching records			
a. Entire File	YES	YES	YES
b. Limits of the File as Described in an RA File	NO	NO	YES
Random or Chaining			
a. CHAIN	NO	YES - Using a relative record number	YES - Using a key field
b. C1-C3	NO	NO	YES - Using a key field
c. RA File	NO	YES - Using a relative record number	YES - Using a key field

Figure 119. Processing Multiple Input Files

A sequential organization (Figure 119) is processed sequentially and controls the processing of records in:

1. Another sequential organization. Both files are processed sequentially, using matching records to govern processing.

2. 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.

An indexed sequential organization (Figure 119) may be processed sequentially or randomly, and controls processing records in:

1. A sequential organization. The records in both files will be processed sequentially, using matching records to control processing.

2. 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 will be used to control processing.

SAMPLE PROGRAM ONE (SALES COMMISSION CALCULATION)

In this sample program, a commission report is to be prepared using invoice summary cards (Figure 120). The cards are in sequence by salesman number. The invoice summary cards are coded with a 5 in column 1; other cards that are maintained as part of the invoice file that are not to be processed are coded as having no 5 in column 1. The commission amount is calculated using the invoice amount:

10% commission for invoice amounts up to $10,000.

12% commission for invoice amounts above $10,000.

The invoice summary card for each salesman is calculated and detail printed. A total commission amount for each salesman and a final commission total amount are accumulated.

The specifications necessary to produce this report are shown in Figure 121.

File Description Specifications Form

Only two specification lines are necessary for this application. The first specification line describes the invoice summary cards that will be read into the system from an IBM 1442 Card Read-Punch. The second line describes the output file PRINT. The size of the print line is 120 characters (columns 25-27).

Input Specifications Form

The input specifications identify and describe the invoice summary card and those cards in the invoice file that are not to be processed (line 6 of the form). Resulting indicator 12 identifies the cards not used by the program. These cards are selected into stacker number 2.

Calculation Specifications Form

The first specification line causes the invoice amount to be compared with factor $10,000. If the invoice amount is greater, resulting indicator 22 is turned on.

The second specification line causes the invoice amount to be multiplied by a factor of 12 percent only if resulting indicator 22 is on. Therefore, this operation takes place only if the invoice amount is greater than $10,000.

The third specification line causes the invoice amount to be multiplied by a factor of 10 percent, only if resulting indicator 22 is off. Therefore, this operation takes place only if the invoice amount is less than or equal to $10,000. Thus, for any one invoice summary card, either specifica-

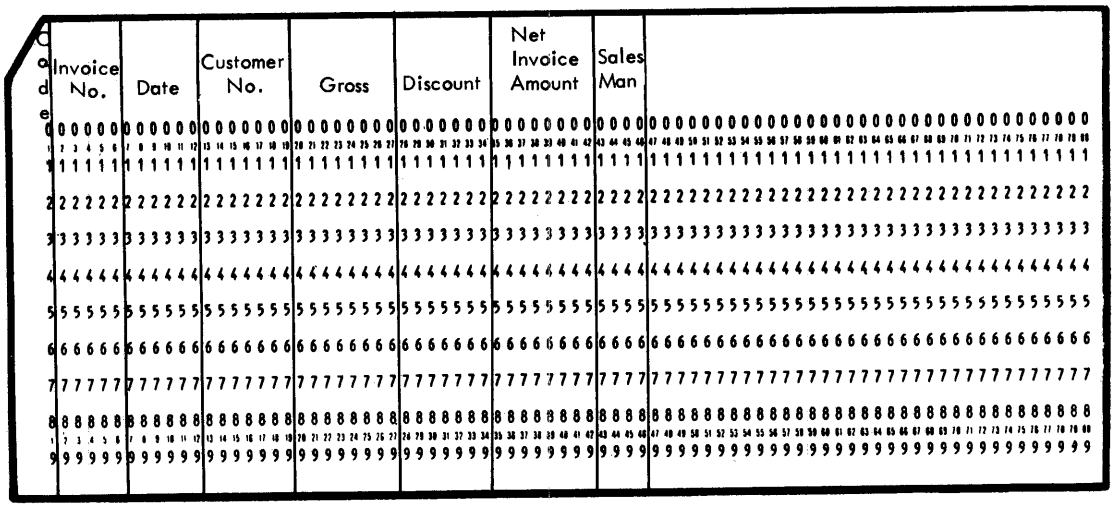


Figure 120. Input Card for Sample Sales Commission Report

tion line 2 or specification line 3 is processed.

The fourth specification line causes the calculated amount COMM from each detail card to be added to the field SUM. The field SUM is used to accumulate the commission amount for each salesman.

The fifth specification line is operated upon only when a level 1 control break occurs, i.e., when there is a change in salesman number. This specification line causes the accumulated commission amount for each group of salesmen cards (SUM) to be added to the field FINTOT. The field FINTOT is used to accumulate a final total of all salesmen's commissions.

Output Specifications Form

Specification lines 1 through 8 provide the specifications necessary for printing field headings. The heading will be printed and the form will be skipped to channel 1 each time one of the following conditions exist:

1. On control break (line 04010, columns 24-25: L1).

2. On overflow condition (line 04020, columns 24-25: OF).

The indicators OFNL1 are required to omit a double printing of the heading when conditions 1 and 2 above occur together.

Specification lines 9 through 16 cause the printing of the detail card information. Note that line 11 or 12, either but not both, is operative for any one detail card.

The remaining specification lines cause:

1. The printing of the level 1 total SUM, including the constant TOTAL (lines 17-19).

2. The printing of the final total, including the constant FINAL TOTAL (lines 1-3 lower half of Figure 121 Part 2).

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date

Program SALES COMMISSION REPORT

Programmer

Punching Instruction: Graphic / Punch

Page 01

Program Identification

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	1/D/T	Type of File Organization	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
010	P	INPUT	I	P			F		80								READ42									
020	P	PRINT	O				F		120				OF				PRINTER									

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System 360

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

Date

Program SALES COMMISSION REPORT

Programmer

Punching Instruction: Graphic / Punch

Page 02

Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (-O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	I	INPUT	AA			11	1		C	5																					
020	I																				2	6		INVOIC							
030	I																				13	190		CUST							
040	I																				35	422		INVAMT							
050	I																				43	46		SALESML	1						
060	I		BB			12	1	N	C	5									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 SALES COMMISSION REPORT

Programmer

Punching Instruction: Graphic / Punch

Page 03

Program Identification

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And Not	Indicators: And Not	Indicators: And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1 > 2	Compare: Low 1 < 2	Compare: Equal 1 = 2	Comments
010	C		11			INVAMT	COMP	10000.00								22			
020	C		22	11		INVAMT	MULT	0.12	COMM		82	H							
030	C		N22	11		INVAMT	MULT	0.10	COMM			H							
040	C		11			COMM	ADD	SUM	SUM		82								
050	C	L1				SUM	ADD	FINTOT	FINTOT		92								

Figure 121. Sample Sales Commission Specification Forms (Part 1 of 2)

IBM

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 04

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Space: Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And Not	Output Indicators: And Not	Output Indicators: And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
010	O	PRINT	H		2	01			L1									
020	O		OR						OF	NL1								
030	O														83		'COMMISSION'	
040	O														69		'PERC'	
050	O														61		'AMOUNT'	
060	O														47		'INVOICE'	
070	O														36		'CUST'	
080	O														28		'SALESMAN'	
190	O		D		2				11									
100	O											COMM	1		83			
110	O								N22						69		'10%'	
120	O								22						69		'12%'	
130	O											INVAMT	1		61			
140	O											INVOIC			46			
150	O											CUST			36			
160	O								L1			SALESM			26			
170	O		T		12				L1									
180	O														69		'TOTAL'	
190	O											SUM	1	B	83			

IBM

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 05

Program Identification

Line	Form Type	Filename	Type (H/D/T)	Space: Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And Not	Output Indicators: And Not	Output Indicators: And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
010	O		T		000	1			LR									
020	O														69		'FINAL TOTAL'	
030	O											FINTOT	1		83			
040	O																	

Figure 121 Sample Sales Commission Specification Forms (Part 2 of 2)

SALESMAN	CUST	INVOICE	AMOUNT	PERC	COMMISSION
2513	11110	12066	9,850.40	10%	985.04
	12129	13444	10,896.00	12%	1,307.52
	14893	14999	110.20	10%	10.02
				TOTAL	2,303.58
4490	15121	25930	1,250.00	10%	125.00
	78230	25220	12,359.20	12%	1,483.10
	72914	44873	690.70	10%	69.07
	49690	25118	8,255.12	10%	825.51
				TOTAL	2,502.68
				FINAL TOTAL	26,482.24

Figure 122 Sample Sales Commission Report

Figure 122 shows a sample of the finished report.

SAMPLE PROGRAM TWO (CUSTOMER TRANSACTIONS)

In this example two input files are used. The transaction file is a card file with fields as shown in Figure 124. Another input file (master customer file), which is on disk, contains information about the firm's customers (Figure 125). The flow of this sample program is shown in Figure 123.

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 126.

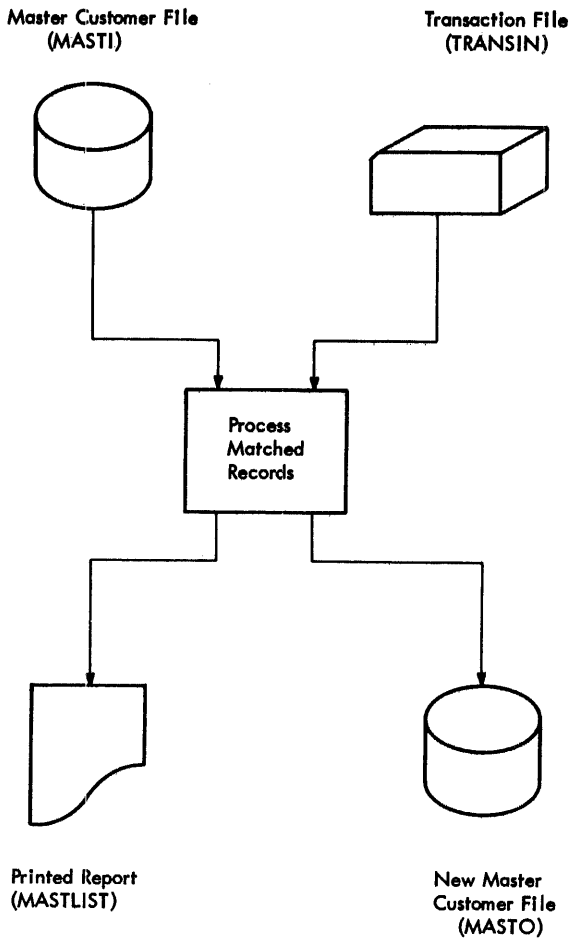


Figure 123. Flow of Sample Program Two

Field	Label	Card Columns
Code	Minus (-) Zone, or Plus (+)	1
Customer name	NAME	8-29
Invoice date	MONTH	32-33
Invoice number	INVNO	34-38
Customer number	CUSTNO	39-43
State	STATE	44-45
City	CITY	46-48
Invoice amount	INVAMT	74-80

Figure 124. Layout of Transaction File Cards

Field	Label	Location
Customer number	MASNUM	1-5
Customer name	MASNAM	6-27
Street	MASTRT	28-46
City	MASCTY	47-57
State	MASTAT	58-62
Customer balance	MASBAL	63-70
Date of last payment	PAYDAT	71-76
Date of last purchase	PAYPUR	77-82

Figure 125. Layout of Master Customer File

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347-
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 01

Program Identification: REPORT

Line	Form Type	Filename	I/O/U/C	F/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind N/U
010	F	MASTI	I	P	E	A	F	300	100							DISK									
020	F	TRANSIN	I	S		A	F	80	80							READ2									
030	F	MASTO	O				F	300	100							DISK									
040	F	MASTLIST	O				F	120	120				OF			PRINTER									

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 02

Program Identification: REPORT

Line	Form Type	Filename	Sequence	Number (1-99)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	I	MASTI	AA			01																									
020	I																				1	100		RECORD							
030	I																				1	50		MASNUML1M1							
040	I																				63	70		MASBAL							
050	I																				71	76		PAYDAT							
060	I																				77	82		PAYPUR							
070	I	TRANSIN	AA			02	1		Z	-									2												
080	I		OR			03	1		Z	&																					
090	I																				8	29		NAME							
100	I																				30	31		MONTH							
110	I																				32	33		DAY							
120	I																				34	38		INVNO							
130	I																				39	43		CUSTNO	L1	M1					
140	I																				44	45		STATE							
150	I																				46	48		CITY							
160	I																				74	80	2	INVAMT							
170	I		BB			04	1		CS																						
180	I																				2	5	0	DATE							

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 03

Program Identification: REPORT

Line	Form Type	Control Level (L0-L9, LR)	Indicators: Not	And	Not	And	Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Minus	Zero or Blank	Compare: High 1 > 2	Low 1 < 2	Equal 1 = 2	Comments
010	C			MR	02			MASBAL	ADD	INVAMT	MASBAL										
020	C			MR	02				MOVE	DATE	PAYDAT										
030	C			MR	03			MASBAL	SUB	INVAMT	MASBAL										
040	C			MR	03				MOVE	DATE	PAYPUR										
050	C																				

Figure 126. Sample Program Two Specifications (Part 1 of 2)

IBM

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 04

Program Identification: REPORT

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	MASTLIST	H		2	01			1P									
02	O	OR	R						OF									
03	O														66		'DAILY TRANSACTION REPO'	
04	O														68		'RT'	
05	O		H		1				1P									
06	O	OR	R						OF									
07	O														25		'CUSTOMER'	
08	O														80		'LOCATION INVOICE'	
09	O														109		'INVOICE DATE INVOICE'	
10	O		H		2				1P									
11	O	OR	R						OF									
12	O														42		'NUMBER CUSTOMER'	
13	O														46		'NAME'	
14	O														79		'STATE CITY NUMBER'	
15	O														108		'MO DAY AMOUNT'	

IBM

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 05

Program Identification: REPORT

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O		D		2				MR N01									
02	O											CUSTNO	Z		22			
03	O											NAME			53			
04	O											STATE	Z		59			
05	O											CITY	Z		67			
06	O											INVNO	Z		79			
07	O											MONTH	Z		89			
08	O											DAY	Z		97			
09	O											INVAMT			109			
11	O								03						14		'CREDIT'	
12	O	MAST0	D						01	NMR								
13	O											RECORD			100			
14	O		T						L1	MR								
15	O											RECORD			100			
	O											MASBAL			70			
	O											PAYDAT			76			
	O											PAYPUR			82			
	O	MASTLIST	T		1				L1	MR								
	O											MASBAL		B	120		'$, , 0 . CR **'	

Figure 126. Sample Program Two Specifications (Part 2 of 2)

File Description Specifications Form

The two input files MASTI and TRANSIN, the output file MASTO (the updated master file) and MASTLIST (the printed report), are defined on the File Description Specifications form.

MASTI is the sequential primary file. 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 in ascending order, fixed-length. Each record is 100 characters in length.

TRANSIN is designated as the seccndary file because it may nct contain transactions for all the customers on the master customer file. TRANSIN must also be in ascending order. It has fixed-length recorrds which are 80 characters long.

The output file MASTO is a disk file, sequentially organized, containing the updated master customer records. The output records are fixed in length, and are 100 characters long.

The file MASTLIST is the printed report. The file format is fixed. The record length can be up to 120. The overflow indicator OF is assigned to this file.

Input Specifications Form

The two input files MASTI and TRANSIN are defined on the Input Specifications form.

MASTI: The disk 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 referred to on the Output-Format Specifications form. MASNUM corresponds to CUSTNO in the transaction file. Whenever a new master number is read, L1 is set. The entry M1 indicates that the master number will be matched with the customer number in the transaction file.

TRANSIN: The input records may be obtained from three types of cards. Sequence AB has been assigned to two types. If card column 1 contains an 11-punch, record identifying indicator 01 is turned on. If card column 1 contains a 12-punch, indicator 03 is turned on. The cards that have an 11-punch in column 1 are selected into stacker 2 (column 42). The locaticns of the input records and their labels are defined in columns 44-58 of the form.

The field CUSTNO (customer number) has entries in columns 59-60 (Control Level) and columns 61-62 (Matching Fields) of the Input Specifications form. Whenever a new customer number is read, control level-1 (L1) is turned on. This condition is tested on the Output-Format Specifications form to govern printing of total lines and to produce the updated customer file. The entry in Matching Fields specifies that CUSTNO will be used to match another field (MASNUM) in the MASTI file.

The first card in the transaction file is a date card. It is assigned sequence BB. Whenever column 1 contains an S, indicator 04 is turned on. The date is contained in columns 2-5 of the card.

Calculations Specifications Form

Whenever the matching record indicator MR is on and indicator 02 is on, the contents of the field INVAMT are added to 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, INVAMT is subtracted from MASBAL, and the result is stored in MASBAL. The date is moved to PAYPUR.

Output-Format Specifications Form

The specifications for the printed report are listed under the name of the output file MASTLIST.

The output file MASTO is the updated disk file. The entries in Output Indicators allow for the following.

Whenever conditions 01 and NMR are satisfied (record identifying indicator 01 is on and no matching record is present), the entire record from MASTI will be written out on disk at detail time. This condition results if 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 on the updated disk file when no information is present in the transaction file.)

If, however, L1 and MR are on, indicating that there was a matching record on the transaction file and calculations were performed, the input record from MASTI is written on disk 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 updated master customer file masto , and the customer's name and address from MASTERIN are retained.

SAMPLE PROGRAM THREE

This example shows the steps required to load or reorganize an indexed sequential file. One reason for reorganization might be the fact that no space is left in the originally defined disk area to load or add any new records.

Both the old file and the new file may be located on the same disk.

File Description Specifications Form

The file OLDINV, which is to be reorganized, is defined as an input file. The organization is indexed sequential (I in column 32). Therefore, the records are arranged in ascending sequence by means of a key field (K in column 31). Begin location of this field in position 1 in each record (1 in column 38). The length of the key field is entered in columns 29-30. The file INVFIL, which is defined as an output file, is the indexed sequential file to be created. Since the data arrangement of the new file should correspond to that of the old file, all entries concerning file organization must be equal. The entry 3000 in columns 47-52 is the number of records that will be provided for in the file.

To execute a LOAD operation, from card only the first file description card has to be replaced by the specifications shown in part 2 of Figure 127.

Input Specifications Form

The records of the indexed sequential file OLDINV are to be automatically retrieved from the disk in the sequence of their key fields. However, only data from the records with character 1 in column 9 is actually transferred to the input fields. Indicator 01 is to be turned on for these records. All other record types, if any, are to be ignored. This is achieved by placing an alphabetic sequence entry in columns 15-16 and record identifying indicator 02. These record types are being considered as deleted and are dropped out of the new file.

Output-Format Specifications Form

All records for which indicator 01 is on are to be included in the new file INVFIL.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System 360

Form X24-3347
Printed in U.S.A.

Date

Program REORGANIZATION

Programmer

Punching Instruction: Graphic / Punch

Page 01

Program Identification ISL0:D

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
01	F	OLDINV	I	P			F		60		8	K	I		DISK						02			
02	F	INVFIL	O				F		60		8	K	I		DISK	3000					302			

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System 360

Form X24-3350
Printed in U.S.A.

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 02

Program Identification ISL0AD

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C Z D	1: Character	2: Position	2: Not (N)	2: C Z D	2: Character	3: Position	3: Not (N)	3: C Z D	3: Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Minus	Zero or Blank	Sterling Sign Position
01	0	OLDINV	AA			01	9		C	1																					
02	0																				1	8	0	KEY							
03	0																				9	9		IDENT							
04	0																				10	60		RECORD							
05	0		BB			02																									

IBM

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 03

Program Identification ISL0AD

Line	Form Type	Filename	Type (H D T)	Space: Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And Not	Output Indicators: And Not	Output Indicators: And Not	Field Name	Zero Suppress (Z) Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	INVFIL	D						01								
02	O											KEY		8			
03	O											IDENT		9			
04	O											RECORD		60			

Figure 127. Sample Program Three (Part 1) Reorganization

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date _____

Program LOAD

Programmer _____

Punching Instruction: Graphic / Punch

Page 01

Program Identification 1SLOAD

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	Length of Key Field or of Record Address Field	K/I	Record Address Type	I/O/T	Type of File Organization	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
01	F	OLDINV	I	P			F		60										READ2								
02	F	INVFIL	U				F		60		8	K	I					L	DISK	3000							
03	F																										

Figure 127. Sample Program Three (Part 2) Load

SAMPLE PROGRAM FOUR

The fourth sample program (Figure 128) illustrates retrieval and updating of records in an indexed-sequentially-organized file by means of the CHAIN operation.

File Description Specifications Form

A card file CARDIN is specified as a primary input file. It is read in from the 2501 Card Reader. The E in column 17 instructs the program to terminate the job when the file CARDIN is exhausted. Chaining is accomplished by the use of the CHAIN operation (see Calculations). Records are fixed length and 80 bytes long.

The chained file INVFIL is located on disk as indicated by the entries in columns 40-53. The U in column 15 describes the file as an update file (used for both input and output). The file is organized indexed-sequentially (I in column 32) and is to be processed randomly as specified by the R in column 28. The begin location of the key field (columns 35-38) is position 1 in each record, and the field is 8 bytes long as specified in columns 29-30. The records for a disk file must be of fixed length (in this case 60 bytes).

Two output files are required: CARDOUT, to punch the exception cards (located on a 1442 Card Punch); and PRINTOUT, to list the transaction report.

Input Specifications Form

The data fields of the input file CARDIN and the update file INVFIL are described on the Input Specifications Form.

Two kinds of data cards may appear in the file CARDIN. The DATE card (sequence entry DD) is identified by an asterisk (*) in column 1. Indicator 02 is turned on when this record type is read in.

Cards identified by a 1 in column 1 (sequence entry CC) supply the fields used for calculations and output with actual data. Indicator 02 is turned on when this card type is read.

The indexed sequential file INVFIL is also defined. Active records are denoted by a 1 in position 9. All other record types will turn on indicator 99. The fields from the active record that will be used are defined.

Calculations Specifications Form

Line 010 specified that the field PART will chain to the file INVFIL, to provide the inventory file record which corresponds to the number in PART. If a record of the transaction file CARDIN has an active record in the inventory file INVFIL (i.e., key field and chaining field are of equal contents) both record identifying indicators 01 and 03 are on at the same time. Calculations are executed only if this con-

dition is satisfied. The ONHAND field from the INVFIL record is reset and added to a work area (SAVE). Data fields RECPTS and RETURN from CARDIN records are added to SAVE; ISSUES are subtracted. When each input card for an INVFIL record is processed, the resulting new ONHAND field in SAVE is compared to the minimum balance. If ONHAND equals the minimum, resulting indicator 05 is set on. If ONHAND is below the minimum, resulting indicator 04 is set on.

Output-Format Specifications Form

The PRINTOUT file's heading informatior can be printed under control of either record identifying indicator 02, which is set for the date card (the first card in the CARDIN file), or overflow (OF). The OF indicator will govern all heading printing after the first page. The entry PAGE in line 05040 causes the page number to be updated automatically for each new page.

The detail line described by entries 05140-05240 requires the presence of both the CARDIN and INVFIL records (record identifying indicators 01 and 03 on). If resulting indicator 04 is on, the words BELOW MINIMUM indicate the stock violation. If indicator 05 is on, the message EXPEDITE is added.

The output line described in PRINTOUT entries 05250-05280 is the message printed when the error condition of a CARDIN record with no corresponding INVFIL record occurs. This condition is identified by record identifying indicator 01 being off when indicator 03 is on.

The exception file CARDOUT will have a card punched for each transaction that results in a below-minimum or at-minimum stock level. These cards will contain the part number and description, vendor number, date, and the E or B code for EXPEDITE or BELOW MINIMUM. Below-minimum cards, identified by the simultaneous on settings of indicators 01, 03, and 04 will be selected to stacker 1. At-minimum cards, with indicators 01, 03, and 05 on, will be selected into stacker 2.

Lines 05370 and 05380 provide for the updating of the INVFIL records. If indicators 01 and 03 are both on, indicating that the INVFIL record has been updated by a CARDIN transaction, the new ONHAND is moved into its INVFIL location on disk from the work area SAVE.

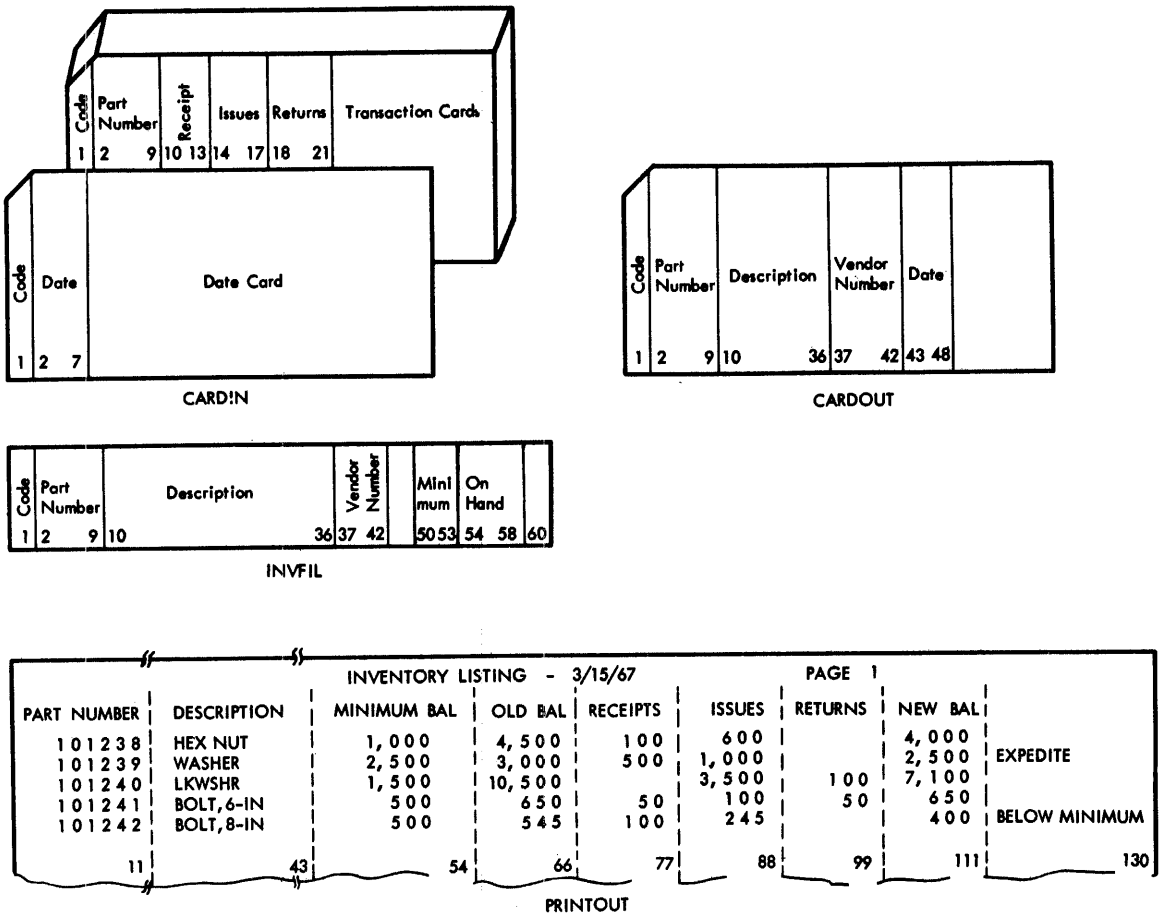


Figure 128. Input and Output Formats for Sample Program Four

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3347
Printed in U. S. A.

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System 360

Date

Program UPDATING

Programmer

Punching Instruction: Graphic / Punch

Page 01

Program Identification: ISUPDT

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Type of File Organization	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind
010	F	CARDIN	I	P	E		F		80								READ01								
020	F	INVFIL	U	C			F		60	R	08	K	I			1	DISK								
030	F	CARDOUT	O				F		80								PUNCH42								
040	F	PRINTOUT	O				F		120					OF			PRINTER								

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

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

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System 360

Date

Program

Programmer

Punching Instruction: Graphic / Punch

Page 03

Program Identification: ISUPDT

Line	Form Type	Filename	Sequence	Number (1-N)	Option (IO)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	I	CARDIN	CC				1		C	1																					
020	I																				2	9		PART							
030	I																				10	13		RECPTS							
040	I																				14	17		ISSUES							
050	I																				18	21		RETURN							
060	I		DD			02	1		C	*																					
070	I																				2	7		DATE							
080	I	INVFIL	AA			01	9		C	1																					
090	I																				10	36		DESC							
100	I																				37	42		VENDOR							
110	I																				50	53		MIN							
120	I																				54	58		ONHAND							
130	I		BB			99																									

Figure 128. Specifications for Sample Program Four

IBM

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 04

Program Identification: ISUPDT

Line	Form Type	Control Level (L0,L9, LR)	Indicators: And Not	Indicators: And Not	Indicators: And Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Resulting Indicators: Minus	Resulting Indicators: Zero or Blank	Compare: High 1>2	Compare: Low 1<2	Compare: Equal 1=2	Comments
01	C		03			PART	CHAININVFIL												
02	C		01				Z-ADD	ONHAND	SAVE		50								
03	C		01			SAVE	ADD	RECPTS	SAVE										
04	C		01			SAVE	SUB	ISSUES	SAVE										
05	C		01			SAVE	ADD	RETURN	SAVE										
06	C		01			SAVE	COMP	MIN									0405		
07	C																		
08	C																		

IBM

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 05

Program Identification: ISUPDT

Line	Form Type	Filename	Type (H,D,T)	Stacker Select	Space: Before	Space: After	Skip: Before	Skip: After	Output Indicators: And Not	Output Indicators: And Not	Output Indicators: And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
010	O	PRINTOUT	H			3			02									
020	O	OR	R			3	01		CF									
030	O											PAGE	Z		104			
040	O														99		'PAGE'	
050	O														68		'INVENTORY LISTING -	
060	O											DATE	Y		77			
070	O	H			3				02									
080	O	OR	R						OF									
090	O														11		'PART NUMBER'	
100	O														27		'DESCRIPTION'	
110	O														66		'MIN. BAL. OLD BAL'	
120	O														88		'RECEIPTS ISSUES'	
130	O														111		'RETURNS NEW BAL'	
140	O	D			2				01	03								
150	O											PART	Z		11			
160	O											DESC			43			
170	O											MIN	1		54			
180	O											ONHAND	1		66			
190	O											RECPTS	1		77			
200	O											ISSUES	1		88			
210	O											RETURN	1		99			
220	O											SAVE	1		111			
230	O								04						130		'BELOW MINIMUM'	
									05						125		'EXPEDITE'	
290	O	CARDOUT	D	1					01	03	04							
300	O	OR		2					01	03	05							
310	O											PART			9			
320	O											DESC			36			
330	O											VENDOR			42			
340	O											DATE			48			
350	O								05						1		'E'	
360	O								04						1		'B'	
370	O	INVFIL	D						01	03								
380	O											SAVE			58			
	O																	

Figure 128. Specifications for Sample Program Four

SAMPLE PROGRAM FIVE

This program (Figure 129) shows how new item records are added to an indexed sequential file. The new records are in card form. The indexed sequential routines for the 1130 do not allow retrieving a file (either randomly or sequentially) in the same program that additions are being made.

File Description Specifications Form

The additions are in the file ADDITNS. The indexed sequential file to be added to is defined as an output file. The record length is 50 characters and the key field begins in position 1 and is 8 characters long. The A in column 66 indicates that this file will be added to. A printer file is also described.

Input Specifications Form

The fields within the ADDITNS file are defined. All records must have a 4 in column 1.

Output-Format Specifications Form

The Output Specifications Form shows heading show lines and the fields of the record being added. The MASTITEM file is being added to at detail time on indicator 01. The entry ADD in columns 16-18 denotes the ADD condition. A record code of 1 is placed in position 9 of the record to denote that it is now an active record. The fields DESCRP and VENDOR are inserted into the record in a normal fashion. The fields MINBAL and BALANC are output in a packed format to conserve disk file space. Each of these fields was defined as 7 positions long and as numeric fields. Packing allows two digits (or one digit and a sign) to be placed in one position on the disk. Each field takes 7 + 1 (1 for the sign) 2 or 4 positions on the disk. (If the fields were 8 positions long, they would require 5 positions each on disk.)

Packing is an effective means of reducing disk file space required. Using Figure 102 (to determine file capacity) shows that if this file requires 5,000 records, almost 7 cylinders of disk space can be saved by packing these two fields.

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3347
Printed in U. S. A.

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page: 01

Program Identification: ADDTO

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	File Addition A	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind N/U
01	F	*																					
02	F	*								ADDTO													
03	F	*																					
04	F	* THIS PROGRAM MAKES ADDITIONS TO THE MASTER INVENTORY FILE																					
05	F	*																					
06	F	ADDITNS	IP			F			80						READ42					A			
07	F	MASTITEMO				F			50		8KI			I	DISK								
08	F	PRINTER				F			120			OF			PRINT32								
09	F																						

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

Form X24-3350
Printed in U.S.A.

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page: 02

Program Identification: ADDTO

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
01	I	ADDITNS	AA			01	1		C	4											2	9		ITEM							
02	I																				10	36		DESCRP							
03	I																				37	42		VENDOR							
04	I																				43	49	0	MINBAL							
05	I																				50	56	0	BALANC							
06	I																														
07	I																														

Figure 129. Sample Program Five (File Description and Input Specifications)

IBM

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 03

Program Identification: ADDTO

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	PRINTER	H		2		01		1P									
02	O		OR						OF									
03	O														47		'ADDITIONS TO INVENTORY'	
04	O											PAGE	Z		67			
05	O														71		'PAGE'	
06	O		H		3				1P									
07	O		OR						OF									
08	O														13		'ITEM'	
09	O														35		'DESCRIPTION	
10	O														55		'VENDOR'	
11	O														68		'MINIMUM'	
12	O														80		'BALANCE'	
13	O		D		2				01									
14	O											ITEM			15			
15	O											DESCRP			45			
	O											VENDOR			55			
	O											MINBAL1			68			
	O											BALANC1			80			
	O																	
	O																	

Card Electro Number _____

IBM

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 04

Program Identification: ADDTO

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	MASTITEMDADD							01									
02	O											ITEM			8			
03	O														9		'1'	
04	O											DESCRP			36			
05	O											VENDOR			42			
06	O											MINBAL			46P			
07	O											BALANC			50P			
08	O																	

Figure 129. Sample Program Five (Output Specifications)

SAMPLE PROGRAM SIX

This program (Figure 130) shows how a typical file maintenance program is written. The file which was added to in sample program five is now being maintained.

The changes can take the form of:

Changes to Active Records
Deletion of Active Records
Reactivation of Deleted Records

The programmer in this program should print an error message for an invalid condition such as trying to delete a record which has already been deleted.

File Description Specifications Form

The changes are in a card file called CHANGES. The indexed sequential file to be changed is called MASTITEM. It is defined as an update file (U in column 15) and a chained file (C in column 16). The file will be processed randomly (R in column 28). The other entries are the same as the previous example. A printer file is also defined.

Indicator Summary Form

The Indicator Summary form is used only for documentation purposes and describes the indicators and their use.

Input Specifications Form

The Input Specifications form shows the file CHANGES and the three types of records possible. The fields are defined for all record types even though the fields will only be present when a record is read with a 1 in position 1.

The file MASTITEM is the Inventory File to be updated. It has both active records (1 in position 9) and deleted records (2 in position 9). The fields MINBAL and BALANC are held in a packed form on disk as described in the previous example.

Calculation Specifications Form

The calculations show all input records chaining to the MASTITEM file using ITEM as the chaining filed. Indicator 20 is used for the "Not Found" condition.

If the record is to be changed, the new fields are moved to the appropriate field names by the MOVE operation. The fields are moved only in the case of Indicator 01 being on (change to active record) and Indicator 20 being off (a record was found for the chaining field).

Output-Format Specifications Form

The output specifications show two heading lines and the fields from the input card to be printed. The detail line also contains messages which show the status of the line. All of the valid conditions are printed in positions 84 to 94. The invalid conditions are printed in positions 96 to 113.

The MASTITEM file is updated for the three valid conditions. A deletion and reactivation require only a change to the record code (position 9).

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

IBM System/360

Form X24-3347
Printed in U. S. A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 01

Program Identification: MAINTN

Line	Form Type	Filename	I/O/U/C	P/S/C/R/T	E	A/D	F/V	Block Length	Record Length	L/R	K/I	I/D/T	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels (S, N, or E)	Name of Label Exit	Extent Exit for DAM	A	File Addition	No. Tracks for Cylinder Overflows	No. of Extents	Tape Rewind	N/U
01	F	*																								
02	F	*								MAINTN																
03	F	*																								
04	F	* THIS PROGRAM MAINTAINS THE MASTER INVENTORY FILE																								
05	F	*																								
06	F	CHANGES IP					F		80							READ	42									
07	F	MASTITEMUC					F		50R		BKI				1	DISK										
08	F	PRINTER O					F		120			OF				PRINT	32									

IBM

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 06

Program Identification: MAINTN

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space Before	Space After	Skip Before	Skip After	Output Indicators And Not	Output Indicators And Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
210											MINBAL	1		68			
220											BALANC	1		80			
230									01	04				90		'CHANGED'	
240									03	05				94		'REACTIVATED'	
250									02	04				90		'DELETED'	
260									01	05				113		'CHANGE TO DELETION'	
270									02	05				110		'DELETE DELETION'	
280									03	04				112		'REACTIVATE ACTIVE'	
290									20					110		'NO RECORD FOUND'	
300		MASTITEMD							01	04							
310											DESCRP			36			
320											VENDOR			42			
330											MINBAL			46	P		
340											BALANC			50	P		
350									02	04							
360														1		'2'	
370									03	05							
380														1		'1'	

Figure 130. Sample Program Six (Part 1)

IBM System/360 Report Program Generator Indicator Summary

Form X26-5590
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 02

Program Identification: MAINTN

Indicators: Input Resulting; Calc. Resulting; Field; Matching & Chaining; Ctl. Break & Halt

Circle Indicators Used:

(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	(OF)
L0	L1	L2	L3	L4	L5	L6	L7	L8	L9	LR	M1	M2	M3	H1	H2	C1	C2	C3	OV

Line	Form Type	Input Resulting	Calc. Resulting	Field	Matching & Chaining	Ctl. Break & Halt	Comments
010	F *	1	C	F	M	L	FUNCTION OF INDICATORS
020	F *	01					CHANGE TO AN ACTIVE RECORD
030	F *	02					DELETION OF AN ACTIVE RECORD
040	F *	03					REACTIVATE A DELETED RECORD
050	F *	04					ACTIVE MASTER RECORD
060	F *	05					DELETED MASTER RECORD
070	F *		20				NOT FOUND CONDITION
080	F *						

INTERNATIONAL BUSINESS MACHINES CORPORATION

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

IBM System/360

Form X24-3350
Printed in U.S.A.

Date _____

Program _____

Programmer _____

Punching Instruction: Graphic / Punch

Page 03

Program Identification: MAINTN

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Resulting Indicator	Record Identification Codes 1: Position	1: Not (N)	1: C/Z/D	1: Character	2: Position	2: Not (N)	2: C/Z/D	2: Character	3: Position	3: Not (N)	3: C/Z/D	3: Character	Stacker Select	Packed (P)	Field Location: From	Field Location: To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field-Record Relation	Field Indicators: Plus	Field Indicators: Minus	Field Indicators: Zero or Blank	Sterling Sign Position
010	I	CHANGES	AA			01	1		C	1																					
020	I		OR			02	1		C	2																					
030	I		OR			03	1		C	3																					
040	I																				1	1		CODE							
050	I																				2	9		ITEM							
060	I																				10	36		NEWDES							
070	I																				37	42		NEWVEN							
080	I																				43	490		NEWMIN							
090	I																				50	560		NEWBAL							
100	I	MASTITEMBB				04	9		C	1																					
110	I		OR			05	9		C	2																					
120	I																				10	36		DESCRP							
130	I																				37	42		VENDOR							
140	I																			P	43	460		MINBAL							
150	I																			P	47	500		BALANC							
	I																														

Figure 130. Sample Program Six (Part 2)

IBM

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 04

Program Identification: MAINTN

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And / And — Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / Minus / Zero or Blank; Compare: High 1>2 / Low 1<2 / Equal 1=2	Comments
01	C			ITEM	CHAINMASTITEM						2020	
02	C		01N20		MOVE	NEWDES	DESCRP					
03	C		01N20		MOVE	NEWVEN	VENDOR					
04	C		01N20		MOVE	NEWMIN	MINBAL					
05	C		01N20		MOVE	NEWBAL	BALANC					
06	C											

IBM

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 05

Program Identification: MAINTN

Line	Form Type	Filename	Type (H/D/T)	Stacker Select	Space: Before / After	Skip: Before / After	Output Indicators: And / And — Not	Field Name	Zero Suppress (Z)	Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
01	O	PRINTER	H		2	01	1P							
02	O		OR				OF							
03	O										47		'INVENTORY MAINTENANCE'	
04	O							PAGE	Z		67			
05	O										61		'PAGE'	
06	O		H		3		1P							
07	O		OR				OF							
08	O										4		'CODE'	
09	O										13		'ITEM	
10	O										35		'DESCRIPTION'	
11	O										55		'VENDOR'	
12	O										68		'MINIMUM'	
13	O										80		'BALANCE'	
14	O		D		2		01							
15	O		OR				02							
16	O		OR				03							
17	O							CODE			3			
18	O							ITEM			15			
19	O							DESCRP			45			
20	O							VENDOR			55			

Figure 130. Sample Program Six (Part 3 of 3)

FORMS

This summary contains a brief column-by-column description of each of the five RPG specifications forms. The purpose of the summary is to provide the user with a concise reference guide. For a complete discussion of the entries, the user should refer to the applicable section.

INFORMATION COMMON TO ALL FORMS

RPG source program cards must be in ascending numeric sequence by column 1 through 5 when they are read by RPG.

Page (Columns 1-2)

Enter the page number of the specifications forms in the following sequence:

File Description Specifications form Extension Specifications form, Input Specifications form, Calculation Specifications form, Output-Format Specifications form.

Line (Columns 3-5)

First two digits of line number are pre-printed. Use column 5 to identify additional lines to be inserted between two pre-printed lines.

Form Type (Column 6)

Contains pre-printed code F, E, I, C, or O which must be punched into all RPG specification cards.

Comments (Column 7)

Enter an asterisk in each line to be used as a comments line.

Program Identification (Columns 75-80)

Insert any characters to identify certain cards or portions of the program.

FILE DESCRIPTION SPECIFICATIONS (F)

File Name (Columns 7-14)

Enter a name for each file. Maximum length: 8 characters of which the first five must be unique. Names must be left-justified. First character must be alphabetic. Special characters or embedded blanks must not be used.

File Type (Column 15)

I = Input File
O = Output File
U = Update File (only disk files)
C = Combined File (only card files)

File Designation (Column 16)

P = Primary file
S = Secondary file
C = Chained file
T = Table file
R = RA File
Leave blank for output files.

End of File (Column 17)

An E in column 17 is required whenever processing of other input files is to be discontinued after the last card of a given input file has been read and processed.

Sequence (Column 18)

If matching fields are specified for an input, combined, or update file, enter:
A if the file is in ascending sequence
D if the file is in descending sequence

File Format (Column 19)

F = Fixed-length records

Block Length and Record Length (Columns 20-27)

Enter right-justified the length of one record in Record Length. Leave Block Length blank.

Record Sizes

Device	Minimum	Maximum
Card	1	80
1403	1	120
1132	1	120
1130 disk		
Sequential	1	640
DA	1	640
ISAM	1	636

Mode of Processing (Column 28) (Disk Only)

Enter the mode by which the file is processed:
R = random processing of a sequentially or indexed sequentially organized

file.

L = sequential processing of the indexed sequentially-organized file between limits (limits are supplied by a RA file).

blank = processing of the sequentially-organized file or sequential processing of the indexed-sequentially-organized file.

Length of Key Field or of Record Address Field (Columns 29-30)

Enter number of positions that each entry in the RA file occupies.

Enter the length of the key, if the file is an indexed-sequentially-organized file. Maximum key length is 50 characters.

Record Address Type (Column 31) (Disk Only)

K = Indexed-sequential file (I in column 32)

Type of File Organization (Column 32) (Disk Only)

blank = sequential organization

I = indexed ssequential organization
Note When an I is entered, there must be a K in column 31, a 1 in columns 35-38, and the length of the key field in columns 29-30.

2 = two I/O areas assigned to a punch or reader with no stacker select specified (any digit 1-9 will denote two I/O areas).

Overflow Indicators (Columns 33-34)

If overflow indicators are used, enter the overflow indicator associated with the file. A maximum of two overflow indicators is allowed: OF and OV. Do not specify an overflow indicator for the console printer.

Key Field Starting Location (Columns 35-38)

If this record is for an indexed-sequential file, enter a 1, right-justified. The key field must begin in position 1 of 1130 indexed sequential records.

Extension Code (Column 39)

E = The file specified on this line is a table file, a chaining file (specified by codes C1-C3) or an RA file.

Device (Columns 40-46)

Enter the device code for the input/output unit used by the file specified in columns 7-14.

Input/Output Unit	Device Code
IBM 2501 Card Reader	READ01
IBM 1442 Card Reader/Punch	READ42
IBM 1442 Card Punch	PUNCH42
IBM 1403 Printer	PRINTER
IBM 1132 Printer	PRINT32
IBM 1130 Console Keyboard	CONSOLE
IBM 2310 Disk	DISK

symbolic device (Columns 47-52)

If the file is indexed sequential and is to be loaded enter the maximum number of records in the file. (Number must be left justified with no commas.)

Columns 53-65

Leave blank.

File Addition (Column 66) (Disk Only)

Enter an A in this column if the file defined on this line is an indexed-sequentially-organized file and new records are to be added. The record to be added is described on the output specifications in columns 16 to 18. If this column contains an A, column 15 must contain an O (output). It is not possible to add and retrieve from the same file in the same program. Column 28 must be blank.

Columns 67-74

Leave blank.

EXTENSION SPECIFICATIONS (E)

Columns 7-8

If the file defined on this line is a chaining file, enter the sequence of that file from columns 15-16 of the Input Specifications sheet. Leave blank if the file described is a record address file.

Number of the Chaining Field (Columns 9-10) (Disk only)

If the file is an RA file or a table file, leave these columns blank; otherwise, enter the number of the chaining field that has also been entered in Chaining Fields (Cols. 61-62) of the Input Specifications form.

From Filename (11-18)

Enter left-justified the name of the table input file, chaining file, or RA file defined on the File Description Specifications form.

To Filename (19-26)

If the From file is a chaining file (using the alternate chaining method), enter the

name of the chained file. If the From file is an RA file, enter the name of the file that contains the data records to be processed. If the From file is a table file, enter the name of the output file to be used after the table has been updated. Leave blank if the table is not to be put out. Note: columns 7-18 are used only for the alternate chaining method.

Table Name (Columns 27-32)

Enter name of table as follows:

- For alternating input format, enter name of table to which the first entry in the input record belongs.
- For non-alternating input format, enter name of single table.

The name must be left-justified and consist of 4, 5, or 6 characters: TAB and one to three alphabetic or numeric characters.

Number of Table Entries per Record (Columns 33-35)

Enter right-justified the maximum number of table entries contained in each input record.

Number of Table Entries per Table (Columns 36-39) - First Table

Enter right-justified the number of table entries. Leading zeros may be omitted.

Length of Table Entry (Columns 40-42) - First Table

Enter right-justified the length of each table entry. Maximum lengths are:

- For alphameric entries, 248 characters
- For numeric entries, 14 numeric digits

For a packed field enter the number of digits of the field.

Packed (Column 43) - First Table

P = Table entries are in packed-decimal format
blank = Table entries are in unpacked-decimal format.

Decimal Position (Column 44) - First Table

Digits 0 through 9 = Number of decimal positions in numeric table entries (cannot exceed number specified in columns 40-42.
Blank = Alphameric table entries.

Sequence (Column 45) - First Table

A = Table entries in ascending sequence
D = Table entries in descending. sequence.
blank = Table entries not in sequence.

Table Name (Columns 46-51) - Second Table

For alternating input formats only. Enter name of table to which the second entry in each input record belongs. Name must consist of 4, 5, or 6 characters: TAB and one to three alphabetic or numeric characters.

Length of Table Entry (Columns 52-54) - Second Table

For alternating input formats only. Enter right-justified the length of each table entry. 248 columns maximum for alphameric, 14 columns maximum for numeric table entries.

For a packed field enter the number of digits of the field.

Packed (Column 55) - Second Table

P = Table entries are in packed-decimal format
blank = Table entries are in unpacked-decimal format.

Decimal Positions (Column 56) - Second Table

Digits 0 through 9 = Number of decimal positions in numeric table (cannot exceed number specified in columns 40-42).
Blank = Alphameric table entries.

Sequence (Column 57) - Second Table

A = Table entries in ascending sequence.
D = Table entries in descending sequence.
blank = Table entries not in sequence.

Comments (Columns 58-74)

Enter any desired comments. An asterisk in column 7 must not be entered in this type of comment if in a line containing specifications.

INPUT SPECIFICATIONS (I)

File Name (Columns 7-14)

Enter a name for each file. (One entry per file). Maximum length: 8 characters. Name must be left-justified. First character must be alphabetic. Special characters or embedded blanks must not be

used. The first five charcters must be unique.

AND/OR-Relationship (Columns 14-16)

AND = The AND-relationship of record identification codes in the preceding line is to be continued.
OR = The entries in record identification codes of this line are to be in an OR-relationship tc the entries of the preceding line.

Sequence (Columns 15-16)

Enter a number, beginning with 01 for each file and continuing in consecutive sequence to 99, to specify sequence-checking of record types. Enter leading zeros. Enter any two alphabetic characters to indicate that sequence checking is not required. Lines with alphabetic entries must precede lines with numeric entries. Any numeric entry in Sequence requires an entry in column 17.

Number (Column 17) (Used Only with Numeric Entry in Columns 15-16)

1 = Only one record of a specific record type may be present before reading another record type.
N = One or more records of a specific record type may be present before reading another record type.

Option (Column 18) (Used Only with Numeric Entry in Coluns 15-16)

O = This record type may not be present.
blank = Record must be present.

Record Identifying Indicator (Columns 19-20)

Two digits 01-99 = Indicator for the input record type defined in columns 21-41.

Record Identification Codes (Columns 21-41)

This field is divided into three identical subfields:

Columns 21-27
Columns 28-34
Columns 35-41.

An AND-relationship exists between these three fields.

Position (Columns 21-24, 28-31, 35-38)

Enter the number of the input record position containing the identifying code. Must be right-justified.

Not (Columns 25, 32, 39)

N = Code described must not be contained in the record position.

C/Z/D (Columns 26, 32, 40)

D = Digit portion of the specified position is to be checked.
Z = Zone portion of the specified position is to be checked.
C = Entire character of the specified position is to be checked.

Character (Column 27, 34, 41)

If C/Z/D contains C or D, enter any one of the 256 EDCDIC characters. If C/Z/D contains Z, enter
12-zone - check for & or any of the 16 characters that have the same zone as an A
11-zone - check for - or any of the 16 characters that have the same zone as J
no zone - absence of 11 or 12 zone-- check for blank or any one of the 16 characters that have the same zone as 1
all others-check for any one of the 16 characters that has the same zone.

Stacker Select (Column 42)

Number of stacker to which input cards are to be selected. Leave blank for normal stacker (of multi-stacker devices) or for single-stacker devices.

Note: Stacker-select entries for combined-file cards that are punched and/or printed must not be made on the Input Specifications form. Stacker select will be ignored if two I/O areas (2 in column 32 of the File Description Specification) are specified.

Packed (Column 43)

P = input data in packed-decimal format.
blank = input data in unpacked format.

Field Location (Columns 44-51)

Location of fields in input record.

FROM (COLUMNS 44-47). Left-most location of the field specified in Field Name.

TO (COLUMNS 48-51). Right-most location of the field defined in Field Name. Number must be right-justified. Leading zeros may be omitted.

Decimal Positions (Columns 52)

For numeric fields only. Enter digits 0 through 9 to indicate number of decimal positions in input field (cannot exceed the field length).

Field Name (Columns 53-58)

Name of each field defined in Field Location. Maximum length is 6 characters. First character must be alphabetic. Special characters or embedded blanks may not be used. Field Name must be left-justified.

Control Level (Columns 59-60)

Any one of the control-level indicators L1 (lowest) through L9 (highest) to identify control fields.

Matching Fields or Chaining Fields (Columns 61-62)

M1 to M9 = record/matching for up to nine input fields or sequence checking for the fields of a single input file.
C1 to C3= alternate method of chaining is specified; codes identify the chaining field. If the operation is used, no entry is necessary in this field.

Field-Record Relation (Columns 63-64)

Enter any indicator specified to provide field-record relation for identical fields contained in different locations (OR-relationship). Normally, these are the record identifying indicators specified in columns 19-20 of the Input Specifications form. Only record identifying indicators may be used when the field is a control field or a match field.

Field Indicators (Columns 65-70)

If a field is alphameric, only zero or blank specifications may be used. Enter any one of the indicators 01-99, H1-H9, as required. If the field specified in columns 46-58 contains any positive value, except +0, the indicator in Plus (columns 65-66) is turned on. If the field contains any negative value except -0, the indicator in Minus (columns 67-68) is turned on. If the field contains only zero or blank, the indicator in Zero or Blank (columns 69-70) is turned on. It is also turned on if the field is numeric and contains only +0 or -0. These indicators are off at the beginning of the program.

Sterling Sign Position (Columns 71-74) Used only for programs processing sterling currency amounts.

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 S in column 74.

CALCULATION SPECIFICATIONS (C)

Control Level (Columns 7-8)

Leave blank to denote operation is to ccurr during detail time. Enter any one of the control-level indicators L0 through L9, or LR (last record) to specify that the calculation contained in this line is to be performed at total time.

Enter SR if statement is part of a RPG subroutine.

Indicators (Columns 9-17)

Enter one to three indicators right-justified. Any of the indicators may be used.

Columns 9, 12, and 15 may contain blank or N only. If there is more than one indicator in a line, RPG assumes an AND-relationship between the individual indicators.

Factor 1 (Columns 18-27)

Enter a field name, a label, or a literal.

Field Name -- Left-justified, maximum 6 characters. First character must be alphabetic. Special characters and embedded blanks may not be used. Must be defined on the input form or as a result field on the calculation form.
Literal -- Numeric: left-justified, maximum length 10 characters. One decimal point and/or one sign (plus or minus) may be used. Alphameric: left-justified, must be enclosed in apostrophes ('), maximum length 8 characters. Any one of the 256 EBCDIC characters may be used.

Label

Enter a label name for use in a TAG or ENDSR statement. First character must be alphabetic. Special characters and blanks may not be used.

Factor 1 is only used with the operations ADD, SUB, MULT, DIV, COMP, LOKUP, TAG, CHAIN, BEGSR, and ENDSR.

Operation (columns 28-32)

Operation code left-justified.

Factor 2 (Columns 33-42)

Field name, label, or literal to be used in the specified operation. (See Factor 1 for definition of field name, label, and literal.)

Factor 2 is not used with the operations MVR, TESTZ, RLABL, TAG, SETOF, SETON, BEGSR, ENDSR, and EXCPT.

Result Field (Columns 43-48)

Result field name. First character must be alphabetic. Special characters or embedded blanks may not be used. Field name must be left-justified. It is not used with the operations COMP, GOTO, EXIT, TAG, SETOF, SETON, CHAIN, BEGSR, ENDSR, EXSR, and EXCPT.

Field Length (Columns 49-51)

Number of storage positions required for result field. Maximum length of numeric fields 14 digits; maximum length of alphameric fields 256 characters. Must be right-justified. May be left blank if already defined in another line of the calculation form or in the input form.

Decimal Position (Column 52)

Decimal positions of the result field (0 through 9). Must be blank if the result field is alphameric. The number of decimal positions cannot exceed the field length.

Half Adjust (Column 53)

H = Half-adjustment of result field.
blank = No half-adjustment of result field

Resulting Indicators (Columns 54-59)

Any indicator 01-99, H1-H9, L1 through L9, LR, OF or OV may be used.

Arithmetic Operations: Enter up to three indicators to be turned on whenever the result is positive (indicator in columns 54-55), or negative (indicator in columns 56-57), or zero (indicator in columns 58-59).

Compare Operations: Enter up to three indicators to be turned on whenever Factor 1 > Factor 2 (indicator in columns 54-55), or Factor 1 < Factor 2 (indicator in columns 56-57), or Factor 1 = Factor 2 (indicator in columns 58-59).

LOKUP Operation: Enter one or two indicators in High, or Low, or Equal, or High and Equal, or Low and Equal.

An indicator in the high column will cause a search for the first table argument (Factor 2) higher than the search argument (Factor 1). An indicator in the low column works in the opposite manner. If there is an indicator in the high or low columns, the table name in factor 2 must be specified as ascending or descending on the extension specifications.

TESTZ Operation: Enter up to three indicators to be turned on whenever a 12-punch (indicator in columns 54-55), or an 11-punch (indicator in columns 56-57), or any other zone or no zone (indicator in columns 58-59) is detected in the field specified in Result Field in this line.

SETOF, SETON Operations: Enter up to three indicators to be turned on (SETON) or off (SETOF).

CHAIN An optional entry of an indicator (the same indicator) in columns 54-57 to be turned on in the case of a not found record.

Comments (Columns 60-74)

Enter any desired comments.

OUTPUT-FORMAT SPECIFICATIONS (C)

File Name (Columns 7-14)

Enter name for each input file. Maximum length 8 characters. First character must be alphabetic. Special characters or embedded blanks may not be used. Name must be left-justified. If several lines of the same file are specified in sequence, the name may be entered in the first line only. The first five characters must be unique.

AND/OR-Relationship (Columns 14-16)

AND = Records in an AND-relationship.
OR = Records in an OR-relationship. (columns 14-15).

Type (H/D/T/E) (Column 15)

H = Heading line.
D = Detail line.
T = Total line.
E = Exception line.

Adding Records to an Indexed-Sequentially-Organized File (columns 16-18)

Enter ADD in these columns if output records are to be added to an IS-organized file. The file description for this file must have an A in column 66.

For card and/or printer files, use these columns as described below.

Stacker Select (Column 16)

Number of stacker to which cards are to be selected. Leave blank for normal stacker (multi-stacker device) or for single-stacker device. Entry is allowed only if the file is described as combined or output in the File Description Specifications.

Space (Columns 17-18)

Enter at least one entry in columns 17-22 if line is to be printed.

BEFORE (COLUMN 17). Enter 0, 1, 2, or 3 to specify line spacing before printing.

AFTER (COLUMN 18). Enter 0, 1, 2, or 3 to specify line spacing after printing.

A entry of 0 for Space Before or Space After for use with the console printer is not permitted.

Skip

At least one entry required in columns 17-22 if line is to be printed.

BEFORE (COLUMNS 19-20). Enter any number from 01 through 12 to specify skipping before printing. Leave blank for no skip before printing.

A skip may not be specified for the console printer, or for channel 7, 8, 10, and 11 with 1132 printer.

AFTER (COLUMNS 21-22). Enter any number from 01 through 12 to specify skipping after printing. Leave blank for no skip after printing.

Output Indicators (Columns 23-31)

Enter right-justified up to three indicators to identify files or describe fields. Columns 23, 26, 29 may contain blank or the letter N (not) only.

Field Name (Columns 32-37)

Enter any name defined in either input or calculation form. Leave blank for constants specified in columns 45-70. Use field name PAGE to cause automatic page numbering on normal printer. Use any field name beginning with PAGE (e.g. PAGE1) for additional page counters.

Edit Codes (Column 38)

The entry in this column controls editing of numeric fields.

1 Print with commas, print zero balance, suppress sign.

2 Print with commas, suppress zero balance and suppress sign.

3 Print without commas, print zero balance, suppress sign.

4 Print without commas, suppress zero balance and suppress sign.

A Print with commas, print zero balance, print sign as CR.

B Print with commas, suppress zero balance, print sign as CR.

C Print without commas, print zero balance, print sign as CR.

D Print without commas, suppress zero balance, print sign as CR.

J Print with commas, print zero balance, print sign as -.

K Print with commas, suppress zero balance, print sign as -.

L Print without commas, print zero balance, print sign as -.

M Print without commas, suppress zero balance, print sign as -.

X Remove positive zone from units position of numeric field. No zero suppression. (This code is accepted by 1130 RPG, but no function is performed since a plus field will print or punch with only a digit in the units position.

Y Edit date field:
3 digit - (n)n/n
4 digit - (n)n/nn
5 digit - (n)n/nn/n
6 digit - (n)n/nn/nn
The first n will be zero suppressed.

Z High order zero suppression and remove sign..

An edit word may be used with a numeric field by leaving this column blank.

Note: If inverted print is specified on the RPG control card (I in column 21), the use of commas and periods will be inverted in all editing with edit codes. Also, the Y code will use period (.) instead of slash (/).

Blank After (Column 39)

B = Reset alphameric output fields to blanks or numeric output fields to zeros.

Reset occurs after execution of the specified output operation.

End Position in Output Record (Column 40-43)

Enter number of output-record position to contain rightmost character of output field.

Packed (Column 44)

P = Output data in packed decimal format (disk only).
blank = Output data in unpacked decimal format.

Constant or Edit Word (Column 45-70)

Constant: Enter any desired constant enclosed in apostrophes. May consist of any of the 256 EBCDIC characters. Maximum length 24 characters.

Apostrophe required within constants must be represented as two consecutive apostrophes.

Edit Word: Enter any edit word to specify editing with respect to punctuation, printing of $ sign status, zero suppression, etc. Must be enclosed in apostrophes (for numeric fields only.)

With edit codes 1-4,A-D, and J-M two entries are possible:

'*' in columns 45-47 to denote check protecting asterisks.

'$' in columns 45-47 to denote a floating dollar sign.

Sterling Sign Position (Columns 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 S in column 74.

The deck of input cards to the Report Program Generator must be preceded by an RPG control card. This control card must be prepared by the programmer or operator according to the functions required of the RPG compiler.

The format of the RPG control card is:

Cols.	Contents	Meaning
1-5	XXXXX	Page and Line Numbers.
6	H	Identifies the card as an RPG control card.
7-10	blank	Not used.
11	B, D, blank	Type of run: D - produce a listing but no compilation. B - bypass listing. blank - compilation with listing.
12-16	blank	Not used.
17	1	The shillings portion of a sterling-currency input field is in the IBM format.
	2	The shillings portion of a sterling-currency input field is in the BSI format.
	blank	The input does not contain sterling-currency fields.
18	1	The pence portion of a sterling-currency input field is in the IBM format.
	2	The pence portion of a sterling-currency input field is in the BSI format.
	blank	The input does not contain sterling-currency fields.
19	0	The shillings portion of a sterling-currency output field is in the printer format.
	1	The shillings portion of a sterling-currency output field is in the IBM format.
	2	The shillings portion of a sterling-currency output field is in the BSI format.
	blank	The output does not contain sterling-currency fields.
20	0	The pence portion of a sterling-currency output field is in the printer format.
	1	The pence portion of a sterling-currency output field is in the IBM format.
	2	The pence portion of a sterling-currency output field is in the BSI format.
	blank	The output does not contain sterling-currency fields.
21	I	The inverted-print option is chosen (i.e., commas are used instead of decimal points. In numeric literals and with edit codes 1-4, A-D, J-M periods are used instead of a slash when using the Y edit code.
	blank	The inverted-print feature is not to be used.
22-25	blank	Not used.

26	A or blank	Alternate collating sequence: Enter an A if an external subroutine is used to translate the sequence of a matching field to the EBCDIC collating sequence. This external subroutine must have the name ALTSE. If an external translating subroutine is not used, leave this column blank.
27-74	7	Not used.
75-80	XXXXXX	These columns can contain the name (six characters) of the user's program. All six characters appear in the listing included in the program

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 extension, and calculationforms 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 eight 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 positions 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 positions 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 Specification 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. It is necessary to add two additional positions for printed output when input is in the BSI format and one extra print position for input in the IBM format.

Note 2: The pounds field consists of at least one and no more than 8 positions. Zero suppression on the pounds field may be obtained by placing Z in column 38 of the specification sheet.

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 and 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.
5. The high order shilling and pence positions will be zero suppressed when a sterling field is edited.

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: 1 for IBM code or 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 or 2.

Lengths of Pence-Format Fields

If a pence-format result field is to be reconverted into a sterling output field, the highest amount it is permitted to contain is 23,999,999,999.999. This converts to a field containing eight pound positions which is the maximum allowed.

Note: If the two high order digits of a pence field are greater than 23, a high order digit will be lost when the field is converted to pound-shillings-pence format for output.

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 the 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 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.

Column 19 (Sterling Shilling 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.

Indicator	Where Located	Where Used	Turned On	Turned Off	Notes
Field Indicators 01-99 Zero and Blank Plus Minus	Input form	Indicator (calc.), Output Indicators	by Blank or Zero in specified field, by Plus in spec. field by Minus in spec. field	before this field status is to be tested the next time	Note 1
H1 through H9	Input form Calculation form	Indicator (calc.), Output	whenever the specified field status or record identification condition is satisfied	internal, at the end of the detail cycle (see Sign. of Program Logic)	Note 1
IR	Internal	Control Level (calc.), Output Indicators	after processing the last record of the last file (see column 17 of File Descr.)	at the beginning of processing	Note 1 Note 2
L0 (Level Zero)	Internal	Control Level (calc.) Output Indicators	at the end of every processing cycle	is never turned off by RPG	
Control Level Indicators L1 through L9	Input form Columns 59 60	Control Level (calc.), Indicators (calc.), Output Indicators	when the value in a control field changes. All indicators of the lower levels are also turned on	at end of following detail cycle	Note 1
MR (Matching	Internal	Indicators (calc.), Output Indicators	if the matching-field contents of the record of a secondary file match the matching-field contents of a record in the primary file	when all total calculations and output are completed for the last record of the matching group.	
OF/OV Overflow	Internal	Indicators (calc.), Output Indicators	(see Significance of Program Logic)	after the following heading and detail lines are completed	Note 3

Indicator	Where Located	Where Used	Turned On	Turned Off	Notes
Record Identifying Indicator 01-99	Input form Columns 19-20	Indicators (calc.), Output Indicators Field-Record Relation	when specified record has been read and before total calculations are executed	before the next record is read during the next processing cycle	Note 1
Resulting Indicators 01-99 Plus Minus Zero Compare operation High Low Equal Lokup operation High Low Equal	Calculation form	Indicators (calc.), Cutput Indicators	by a positive balance in field, by a negative balance in field, by Zero balance in field if Factor1>Factor2 if Factor<Factor2 if Factor=Factor2 if table > Factor1 if table < Factor1 if table=Factor1	the next time a calculation is performed for which the program specifies the indicator as a resulting indicator and the specified condition is not satisfied	Note 1
1P (First Page)	Internal	Output Indicators	at beginning of processing before any input records are read	before the first detail card is read	Note 4

Note 1. Turning indicators on or off can also be accomplished by using SETON and SETOF operation codes.
Note 2. All control level indicators (L1-9) are also turned on when LR is turned on.
Note 3. The OF indicator remains on during the following detail calculations and output cycles.
Note 4. This indicator is used to condition printing of the first page of the report.

READER'S COMMENT FORM

IBM 1130 Planning
For RPG

Form C21-5002-0

- 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?	☐	☐
• Did you find the material:		
Easy to read and understand?	☐	☐
Organized for convenient use?	☐	☐
Complete?	☐	☐
Well illustrated?	☐	☐
Written for your technical level?	☐	☐

- What is your occupation? _____
- How do you use this publication?

As an introduction to the subject?	☐	As an instructor in a class?	☐
For advanced knowledge of the subject?	☐	As a student in a class?	☐
For information about operating procedures?	☐	As a reference manual?	☐

Other _____

- Please give specific page and line references with your comments when appropriate.

COMMENTS:

IBM 1130 Printed in U.S.A. C21-5002-0

This SRL manual is part of a library that serves as a reference source for systems analysts, programmers and operators of IBM systems. Your answers to the questions on the back of this form, together with your comments, will help us produce better publications for your use. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material. All comments and suggestions become the property of IBM.

Please note: Requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or to the IBM sales office serving your locality.

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

IBM 1130 Printed in U.S.A. C21-5002-0

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]