IBM System/3 RPG II Reference Manual

Program Numbers:

5702-RG1 (Model 10)
5704-RG1 (Model 15)
5704-RG2 (Model 15)
5705-RG1 (Model 12)

SC21-7504-6
File No. S3-28

Program Product

Seventh Edition (September 1976)

This is a major revision of, and obsoletes, SC21-7504-5 and Technical Newsletter SN21-5338. Changes to text and illustrations are indicated by a vertical line to the left of the change. New or extensively revised illustrations are denoted by a ● at the left of the figure caption.

This edition applies to the System/3 program versions listed below and to all subsequent versions and modification levels until otherwise indicated in new editions or technical newsletters.

System/3	Version	Modification	Program Number
Model 10	13	00	5702-RG1
Model 12	2	00	5705-RG1
Model 15	5	00	5704-RG1
Model 15	1	00	5704-RG2

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

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Publications, Department 245, Rochester, Minnesota 55901.

This publication is intended as a reference manual. It is organized so that all the information necessary to write RPG II programs for the IBM System/3 Model 10 Disk System, IBM System/3 Model 12, and IBM System/3 Model 15 can be easily found.

System/3 Model 8

The System/3 Model 8 is supported by System/3 Model 10 Disk System Control Programming and Program Products. The facilities described in this publication for the Model 10 are also applicable to the Model 8, although the Model 8 is not referenced. It should be noted that not all devices and features which are available on the Model 10 are available on the Model 8. Therefore, Model 8 users should be familiar with the contents of *IBM System/3 Model 8 Introduction,* GC21-5114.

System/3 Models 12 and 15 Only

The IBM 3340 Direct Access Storage Facility attaches to System/3 Model 12 and to System/3 Models 15B, 15C, and 15D. Also, the IBM 3344 Direct Access Storage attaches to System/3 Model 15D. Certain areas on the 3340 and 3344 disks are treated as 5444 disks. These areas, known as 5444 simulation areas (or simulation areas), are used for the program libraries and can also be used for data files. The remainder of the disk space, known as main data area, can be used only for data files.

References in this manual to 5444, 5445, and 3340 are to be interpreted according to the disk storage device(s) attached to the system. The following table should be used to determine the meaning of the reference:

	Model 15A	Models 12, 15B, 15C	Model 15D
Reference	Meaning	Meaning	Meaning
5444	5444 Disk Storage Drive	5444 simulation area on 3340	Simulation area on 3340 or 3344
5445	5445 Disk Storage	Main data area on 3340	Main data area on 3340 or 3344
3340	Not applicable	Main data area on 3340	Main data area on 3340 or 3344

For further information, see the appropriate publications listed under *Related Publications.*

All references to the System/3 Model 15 in this publication apply to System/3 Models 15A, 15B, 15C, and 15D unless otherwise specified.

IBM System/3 5448 Disk Storage Drive

The IBM System/3 5448 Disk Storage Drive on System/3 Models 8 and 10 uses the same program product support as the IBM 5445 Disk Storage. However, a separate system control program feature is required for the 5448. In general, references to 5445 in this manual also apply to 5448. For specific information about 5448 operating characteristics and programming support, see *IBM System/3 5448 Disk Storage Drive Program Reference Manual,* GC21-5168.

Related Publications

<table>
<tr><th rowspan="2">Publication</th><th colspan="5">Order Number of Publication for</th></tr>
<tr><th>Model 8</th><th>Model 10</th><th>Model 12</th><th>Model 15 (5704-RG1)</th><th>Model 15 (5704-RG2)</th></tr>
<tr><td>System/3 Introduction</td><td>GC21-5114</td><td>GC21-7510</td><td>GC21-5116</td><td colspan="2">GC21-5094</td></tr>
<tr><td>Introduction to RPG II</td><td colspan="5">GC21-7514</td></tr>
<tr><td>RPG II Disk File Processing</td><td colspan="5">GC21-7566</td></tr>
<tr><td>RPG II Additional Topics</td><td colspan="5">GC21-7567</td></tr>
<tr><td>RPG II Auto Report</td><td colspan="5">SC21-5057</td></tr>
<tr><td>RPG II Telecommunications</td><td colspan="5">SC21-7507</td></tr>
<tr><td>SCP Reference Manual</td><td colspan="2">GC21-7512</td><td>GC21-5130</td><td>GC21-5077</td><td>GC21-5162</td></tr>
<tr><td>System/3 3340 Reference</td><td colspan="2"></td><td colspan="2">GC21-5111</td><td></td></tr>
<tr><td>System/3 3741 Reference</td><td colspan="5">GC21-5113</td></tr>
<tr><td>Disk Concepts and Facilities</td><td colspan="5">GC21-7571</td></tr>
<tr><td>System Generation</td><td colspan="2">GC21-5126</td><td></td><td colspan="2">GC21-7616</td></tr>
<tr><td>Operator's Guide</td><td>GC21-7634</td><td>GC21-7508</td><td>GC21-5144</td><td colspan="2">GC21-5075</td></tr>
<tr><td>User's Guide</td><td colspan="2"></td><td>GC21-5142</td><td colspan="2"></td></tr>
<tr><td>Halt Guide</td><td colspan="2">GC21-7540</td><td>GC21-5145</td><td colspan="2">GC21-5076</td></tr>
<tr><td>3340 Disk Storage Reference</td><td></td><td></td><td colspan="2">GA26-1619</td><td></td></tr>
<tr><td>Overlay Linkage Editor</td><td colspan="5">GC21-7561</td></tr>
</table>

Contents

HOW TO USE THIS MANUAL

Chapters

This publication has nine chapters and several appendixes. Chapter 1 is an introduction. Chapter 2 describes RPG II coding entries common to all specification types. Chapters 3-9 describe the seven types of RPG II specifications in the order they are read by the RPG II compiler. The appendixes contain additional information useful in RPG II programming, including convenient reference tables and performance improvement tips.

Column Descriptions

Specifications are described column-by-column as a programmer would write them. The following information is included for each column description:

1. List of possible entries.

2. General discussion of use of column and considerations for all possible entries.

3. Specific discussion of each entry.

4. Charts and examples.

Special Topics

Some RPG II features require multiple, interrelated specifications or are especially important and merit expanded discussion. Examples are multifile processing, tables and arrays, and operation codes. These features are discussed near the specifications which are key to their use.

FUNCTION OF RPG II

RPG II consists of a symbolic programming language and a compiler program. The RPG II symbolic language is a highly flexible, problem solving language. It allows programming solutions to a wide variety of data processing problems. The compiler program translates the symbolic language program (source program) into a machine language program (object program). The object program is used by System/3 to process information according to the programmer's specifications.

Basically, the program undergoes two processes:

1. Compilation. The source program is translated into an object program.

2. Execution. The object program is used to process data.

During compilation, the program specifications you wrote 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.

During execution, the machine language instructions are combined with the input data files and both are processed through the system to do the job.

USING RPG II

Doing a job using RPG II 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 analyzes the job 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.

2. After the programmer has analyzed the requirements of the job, he provides the RPG II program with information about these requirements.
   a. He furnishes special information about his program and describes his system by making entries on the sheet containing Control Card specifications.
   b. He describes all files used by the object program (input files, output files, table files, etc.) by making entries on the File Description Specifications Sheet.
   c. If the programmer uses record address files, tables, or arrays in his object program, he furnishes information about them through entries on the Extension Specifications Sheet.
   d. He provides certain information about the format of printed reports on the Line Counter Specifications Sheet.
   e. He describes his input files by making entries on the Input Specifications Sheet.
   f. He states what processing is to be done (add, subtract, multiply, divide, etc.) by means of entries on a Calculation Specifications Sheet.
   g. He defines the layout of the desired report (print positions, carriage control, etc.) by making entries on the Output-Format Specifications Sheet.

3. After the specifications have been written on the appropriate forms, the data on the forms is recorded in punched cards, entered into the system through the keyboard or the 3741.

4. These specifications (called the source program) are preceded by the RPG II control card. The source program and the control card are processed by the RPG II compiler under control of the *Disk System*. At the end of this processing run (referred to as the compilation run), the object program is stored in an object library, punched in cards, or written on a diskette. This program contains all the machine instructions required to perform the desired job.

5. When the object program is to be executed, it is read into main storage from cards, disk, or diskette.

6. The input files are read by the system under control of the object program. This is known as the object run.

DEFINITIONS OF TERMS

EBCDIC (Extended Binary-Code-Decimal Interchange Code) Notation: The 256-character machine code used in the IBM System/3 Disk System. See Appendix D for a table of hexadecimal equivalents of the EBCDIC characters.

Alphabetic Characters: The 26 alphabetic EBCDIC characters and the three EBCDIC characters #, $, and @.

Numeric Characters: The EBCDIC characters 0-9.

Special Characters: The 217 EBCDIC characters not defined as alphabetic or numeric.

Alphameric Characters: Any of the 256 EBCDIC characters.

Alphameric Fields: All fields for which a decimal-positions specification has not been made in the appropriate column of the specifications forms. Alphameric fields can contain alphabetic, numeric, or special characters.

Numeric Fields: All fields having a decimal-positions specification in the appropriate columns of the specifications forms.

Valid RPG II Names: The following rules apply to names used in RPG II programs:

- RPG II filenames can be from 1-8 characters long; RPG II field names can be from 1-6 characters long.
- The first character of either a filename or a field name must be alphabetic (see preceding definition of alphabetic characters). The remaining characters can be any combination of alphabetic and numeric characters (special characters are not allowed).
- Blanks cannot appear between characters in the name.

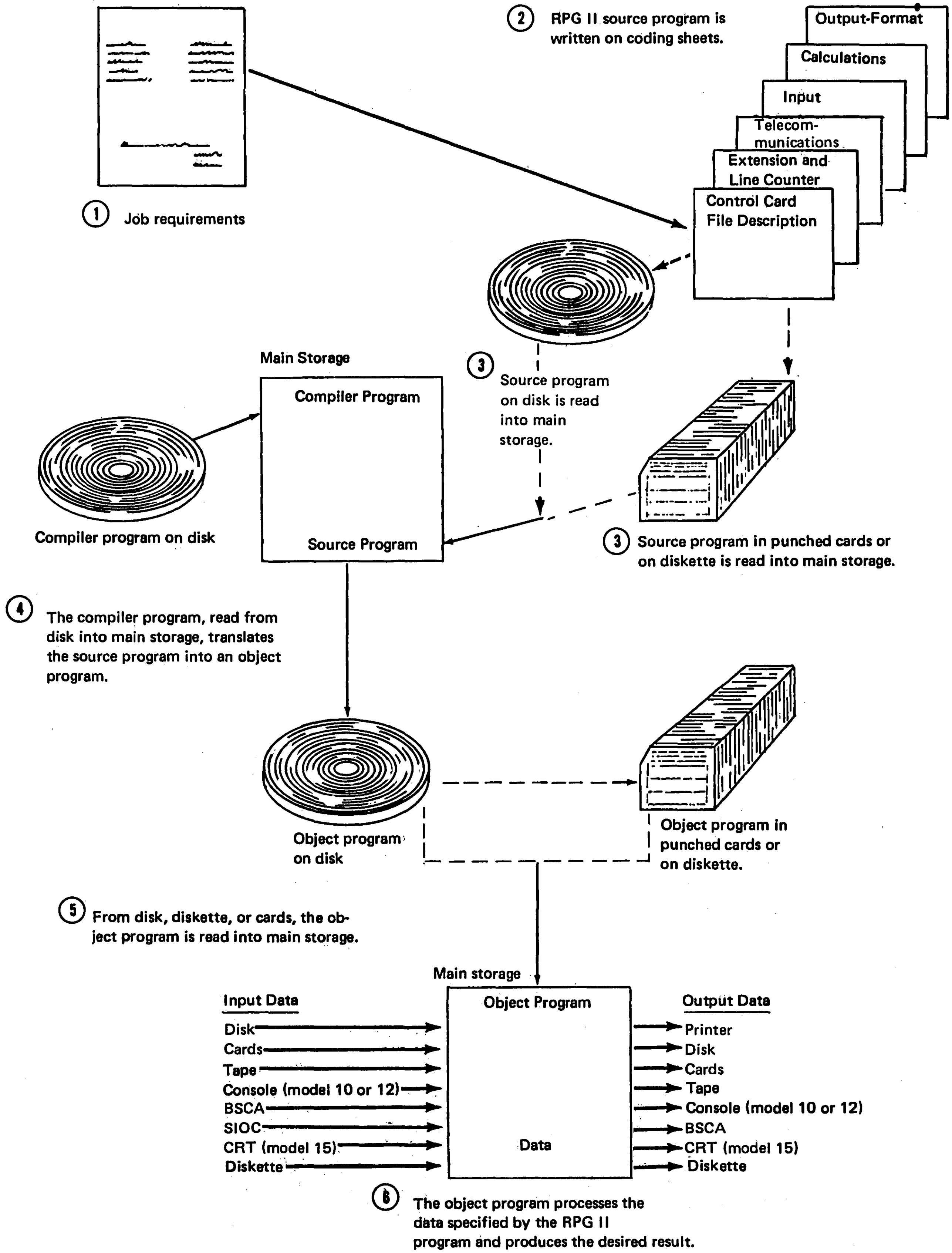


Figure 1. Performing a Job Using RPG II

GENERAL RPG II OBJECT PROGRAM LOGIC

Every object program generated by the RPG II Compiler uses the same general program logic (Figure 2). The term program logic refers to all the RPG II functions performed for each data record read.

Knowledge of RPG II logic is helpful when writing RPG II programs. For relatively simple jobs involving a single input file, an understanding of the general logic presented here is sufficient. Complex jobs require a more thorough understanding of the logic. *Appendix C: Detailed RPG II Object Program Logic* contains a detailed flowchart and explanation of the program logic.

Every program cycle involves three basic logic steps:

1. Reading information (input).
2. Performing calculations (processing).
3. Recording results (output).

Within a program cycle, these basic logic steps can be divided into numerous substeps in which the input determines when calculation and output operations occur. According to RPG II program logic, calculation and output operations (including exception output) are performed at two different times in a cycle: total time and detail time.

Total Operations

Total calculations are specified by placing an *L* indicator in columns 7-8 of the Calculation Sheet. Total output operations are specified by placing a *T* in column 15 of the Output Sheet. The appropriate control level indicator should be entered in columns 23-31 of the Output Sheet to distinguish between output operations performed for different control levels.

Total calculation and total output operations are normally performed on data accumulated for a group of related records which form a control group. Such operations are normally done only after a control break has occurred. A control break occurs when the control field of the record just read is different from the control field of the previous record. Whenever a record is read, a check is made to determine if information in a control field (when one has been specified) is different from the control field information on the previous record.

A change in the control field information indicates that all records from a particular control group have been read and a new group is starting. When all records from a group have been read (shown by control level indicators being turned on), calculation and output operations are done using information accumulated from all records in that group. Information on the record that started the new control group is not used in these total operations; only information from records in the previous control group is used.

Detail Operations

Those calculations not conditioned by *L* indicators in columns 7-8 are detail calculations. Detail output operations are specified by placing an *H* or *D* in column 15 of the Output Sheet. Detail calculation and detail output operations are normally performed for individual data records. These operations are done for each record, provided all conditioning indicators are satisfied. When any one of the following conditions are met, detail time calculation and output operations are done:

1. All total calculation and total output operations have been completed.
2. No total operations are to be done (the information in the control field has not changed).

Total operations are performed before detail operations. This prevents data from the first record in a new control group from being accumulated in the totals for the previous group. Total operations are performed only on data accumulated from previous records. Detail operations on the record that caused the control break are done after total operations are finished.

General Program Cycle

Figure 2 shows specific steps in the general flow of RPG II program logic. A program cycle begins with step 1 and continues through step 11, then begins again. Steps 7 and 8 are known as total time; steps 11 and 1 are known as detail time.

The first and last program cycles of a job are somewhat different from the normal cycle. Before the first record is read, lines conditioned by the 1P indicator are written. Any heading or detail lines having no conditioning or having all negative conditioning indicators are also written at this time. In addition, total operations are bypassed for the first record even though a control break may occur.

When the last record to be processed is read, the last record (LR) indicator turns on. This automatically causes all control level indicators to turn on also. Total operations are performed and the job ends; only steps 3-8 of the program cycle are done.

1. Before the first record is read, the program writes all heading or detail records (those having an *H* or *D* in column 15 of the Output Sheet). This is done only if all conditioning indicators are satisfied.

2. All record identifying indicators are turned off.

3. A record is read and identified by the object program. The appropriate record identifying indicator is turned on.

4. The record just read is examined to determine whether or not a control break has occurred. A control break occurs when the control field of the record just read is different from the control field of the previous record.

5. If a control break occurs, the proper control level indicators turn on except L0 which is always on.

6. A check is made to determine if this is the first cycle. If it is, total calculations and total output (steps 7 and 8) are bypassed.

7. Total calculation operations (those conditioned by control level indicators in columns 7-8 of the Calculation Sheet) are performed if the control level condition is satisfied.

8. Total records (those having a T in column 15 of the Output-Format Specifications Sheet) are written or punched out according to output specifications.

9. If matching fields have been specified, these fields are checked for a matching condition. The matching record (MR) indicator is set accordingly.

10. Data from the record read at the beginning of the cycle (step 3) is now made available for use in detail calculation and output operations.

11. All detail calculation operations (those not conditioned by level indicators in columns 7-8 of the Calculation Sheet) are performed on the data from the record read at the beginning of the cycle. Chaining and exception output can also be performed.

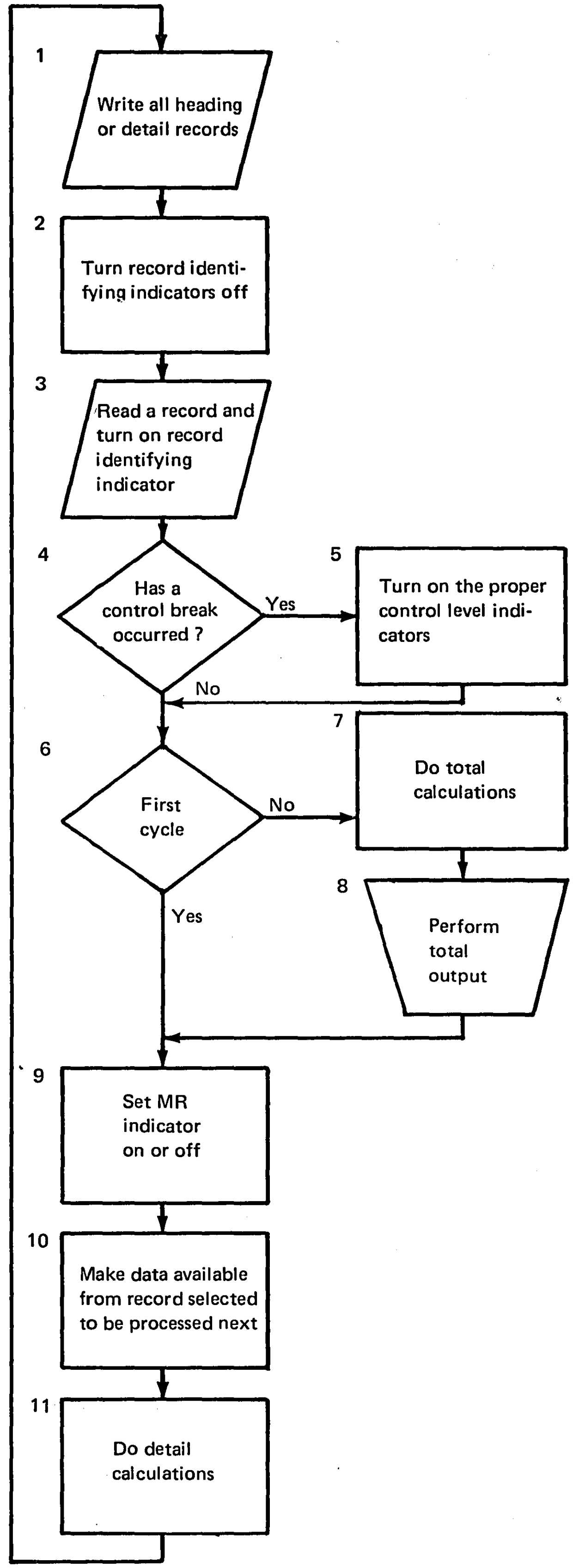


Figure 2. General Object Program Cycle

SYSTEM CONFIGURATION

For minimum system configurations and optional device support, see the appropriate system generation reference manual listed under *Related Publications* in the Preface.

RPG II SPECIFICATION SHEETS

The RPG II specification sheets are used when coding an RPG II program. The format and column headings on each of these sheets guide you in making the appropriate entries.

The sheets are designed so that one card is punched from each specification line. There are six specification sheets:

1. *Control Card and File Description.* This sheet contains two types of specifications:

   a. Control card specifications provide information to the RPG II compiler.
   b. File description specifications provide information about all files used in the program.

2. *Extension and Line Counter.* This sheet contains two types of specifications:

   a. Extension specifications provide information about tables, arrays, and record address files.
   b. Line counter specifications provide information about the number of lines to be printed on the forms that are used.

3. *Telecommunications.* This sheet is used to enter the information necessary to establish and maintain the BSC communications link. Each BSCA file defined on the File Description Sheet must have a corresponding Telecommunications Sheet entry.

4. *Input.* This sheet is used to describe the records in an input file.

5. *Calculation.* This sheet is used to describe all operations that are to be performed on the data.

6. *Output.* This sheet is used to specify the arrangement and type of data that will be written or punched on printed reports or cards, or stored on disk.

Information on specification sheets is recorded in punched cards to form a source program. The arrangement of the cards is shown in Figure 3.

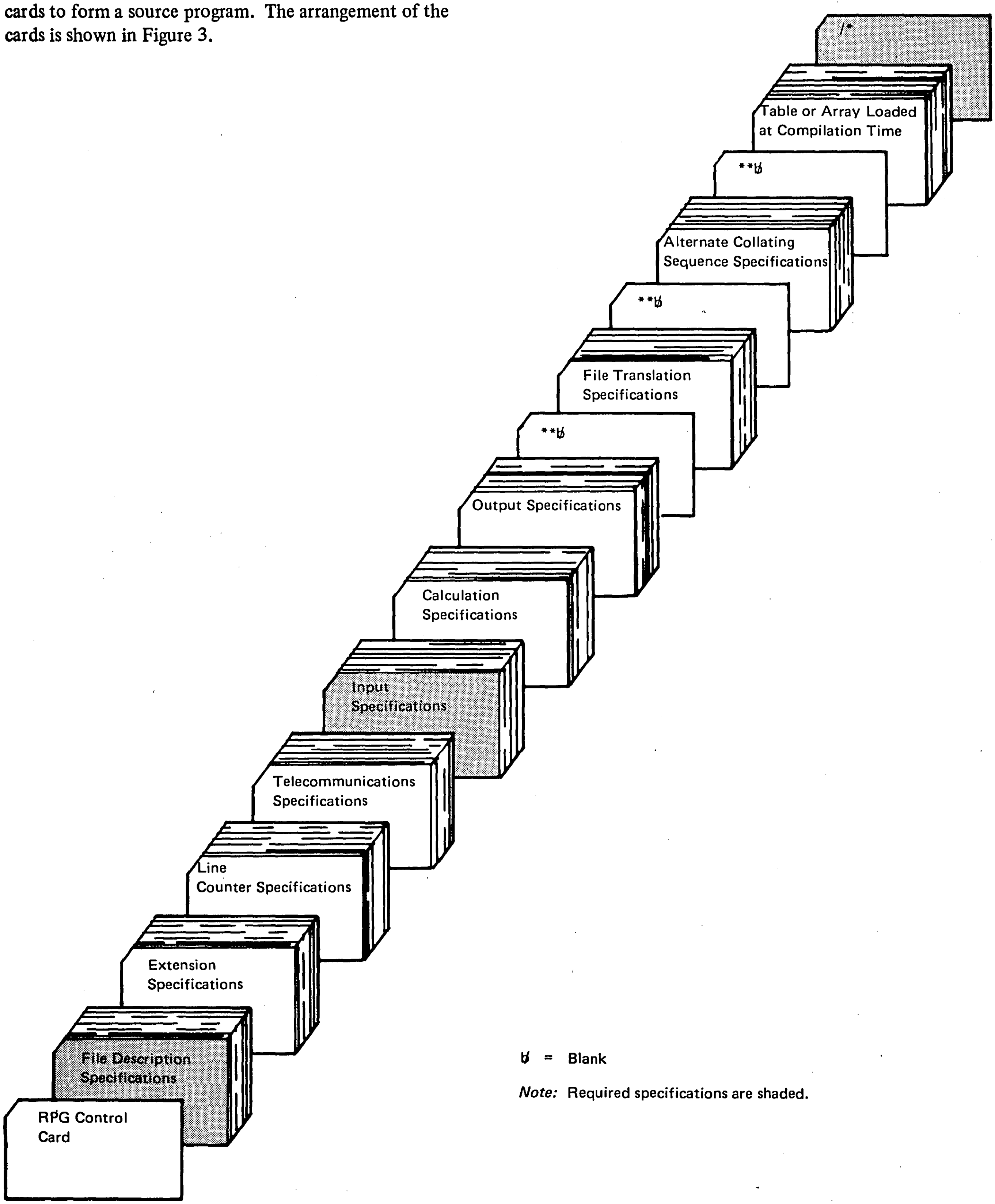


Figure 3. Record Arrangement in the RPG II Source Deck

Chapter 2. Common Entries

This chapter defines entries common to all RPG II coding sheets. Each coding sheet contains the following entries:

1. Columns 1-2 (page).
2. Columns 3-5 (line).
3. Column 6 (form type).
4. Column 7 (comments).
5. Columns 75-80 (program identification).

COLUMNS 1-2 (Page)

Entry	*Explanation*
01-99	Page number
/*, /&, or /.	Indicates end of source specifications data. /. is used on Model 15 only.
**	Followed by a blank in column 3 is a delimiter for table or array, data, alternate collating sequence specifications, and file translation specifications.

Columns 1-2 in the upper right corner of each sheet are for numbering the specification sheets used in a job. You can use more than one of each sheet, but all sheets of the same type should be kept together. When all the specifications sheets are filled out, arrange them in the following order and number them in ascending sequence:

1. Control Card and File Description
2. Extension and Line Counter
3. Telecommunications
4. Input
5. Calculation
6. Output

COLUMNS 3-5 (Line)

Entry	*Explanation*
Any numbers	Line numbers

Columns 3-5 are used to number the lines on each sheet. Columns 3 and 4 contain preprinted line numbers so, in most cases, line numbering is already done for you. For instance, the Control Card and File Description Sheet contains line numbers for lines 01-10. The unnumbered lines below the preprinted numbers can be used for additional lines or to insert a line between two other completed lines (see *Example*).

The control card specification line is always line 01. Any other lines on the specification sheets can be skipped. The line numbers you use need not be consecutive, but should be in ascending order.

Example

Figure 4 shows the insertion of a line between two completed lines. To show that a line belongs between line 02 and line 03, a *5* is placed in column 5 (any number 1-9 can be used). Line 025 should be inserted between 02 and 03. All lines inserted between existing lines should be written after the last line with a printed line number.

Note: After the source cards have been punched, cards from insert lines must be placed in proper sequence.

COLUMN 6 (FORM TYPE)

Entry	*Explanation*
H	Header card (Control Card Specification Sheet).
F	File Description Specifications Sheet.
E	Extension Specifications Sheet.
L	Line Counter Specifications Sheet.
T	Telecommunications Specifications Sheet.
I	Input Specifications Sheet.
C	Calculation Specifications Sheet.
O	Output Specifications Sheet.

Column 6 contains a preprinted letter on all sheets. The letter identifies the type of specifications for each line.

COLUMN 7 (COMMENTS)

Entry	*Explanation*
*	Comment line

You may wânt to write comments to help you understand or remember what is being done in a certain section of coding. RPG II allows an entire line to be used for these comments. The comment line is identified by placing an asterisk in column 7. Any characters in the character set may be used in a comment line. A card is punched from this line and the comments appear in the source program listing.

Comments are *not* instructions to the RPG II program. They serve only as a means of documenting the program. A comment line cannot be written in the control card specifications line.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

Entry	*Explanation*
Valid RPG II name	Program identification (the first character must be alphabetic but cannot be #, $, or @; the remaining characters must be alphameric with no imbedded blanks. Special characters cannot be used.)
Blank	RPGOBJ is assumed.

Control Cards

Columns 75-80 (at the top of the Control Card Sheet) are used to name your object program. This name is used in a directory that contains the location of your program on disk. The compiler places the first four characters (columns 75-78) into positions 89-92 (or columns 73-76 if the object program is put on 80-column cards) of each record in your object program. Columns 75-80 of the control card must contain an entry when an object program is permanently cataloged on the object library (a *C* in column 10 of the control card). If columns 75-80 are left blank, the compiler assumes the entry is RPGOBJ. The compiler (the overlay linkage editor on Model 15) uses columns 93-96 of each object program record for consecutive numbering of the records. The name should be unique.

Note: DIR, ALL, and SYSTEM are reserved names and must not be used as the name of an object program.

All Other Source Cards

Columns 75-80 on all source program cards, except the control card, may contain any characters. These columns may use the program name in the control card, or the column may contain any other characters to identify a certain portion of the program. These entries are ignored by the compiler, but will appear in the source program listing.

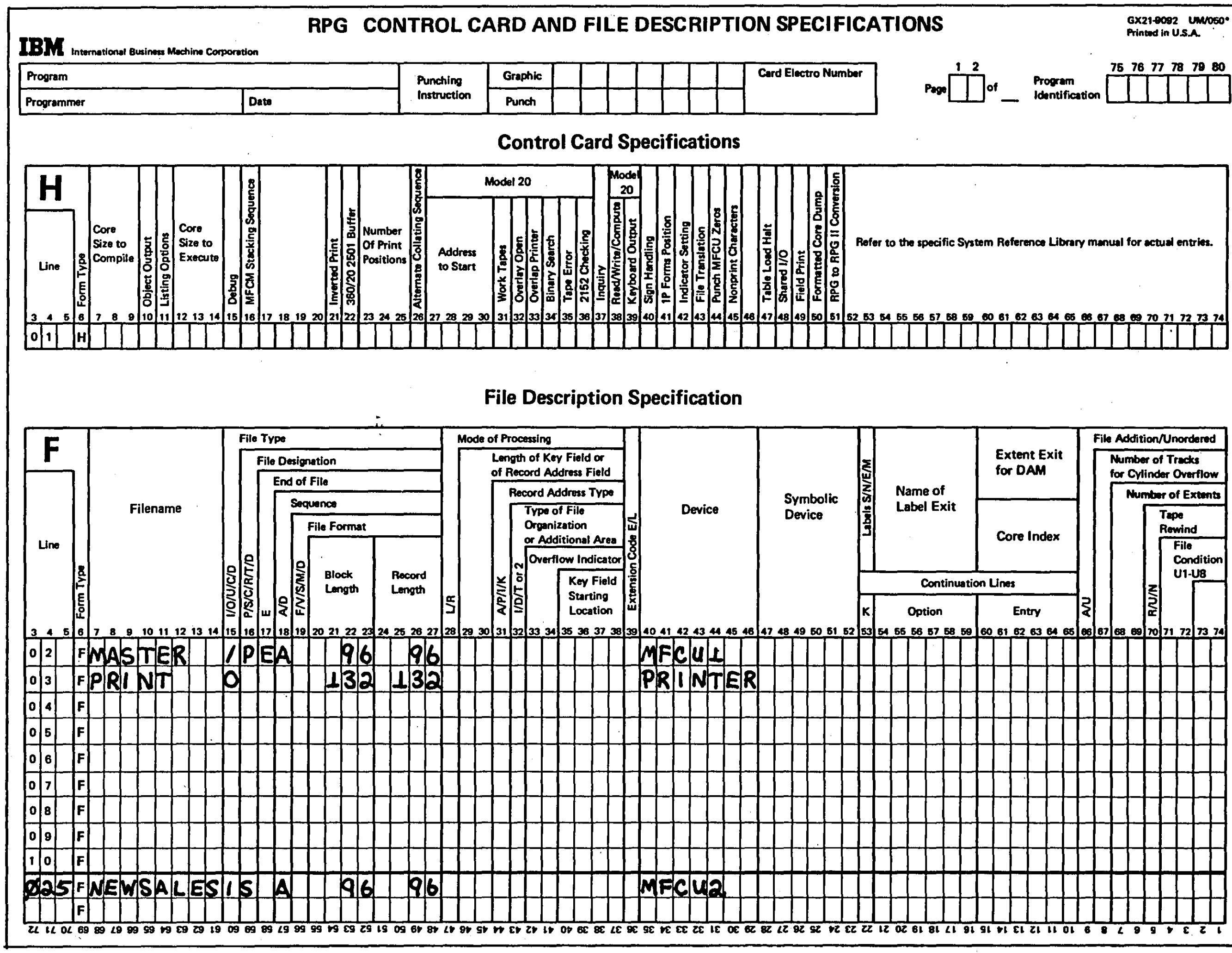


Figure 4. Insertion of Lines

Only one control card may be submitted for each program. It provides information about your program and your system to the RPG II compiler. If the control card is omitted, a blank control card is assumed. For coding the control card, one specification line is provided on the Control Card and File Description Sheet (Figure 5).

COLUMNS 1-2 (PAGE)

See Chapter 2.

COLUMNS 3-5 (LINE)

See Chapter 2.

COLUMN 6 (FORM TYPE)

An *H* must appear in column 6. A control card with an *H* punched in column 6 must be entered for every program even if all the other columns are left blank.

COLUMNS 7-9 (CORE SIZE TO COMPILE)

Columns 7-9 are not used. The program is compiled in the available main storage.

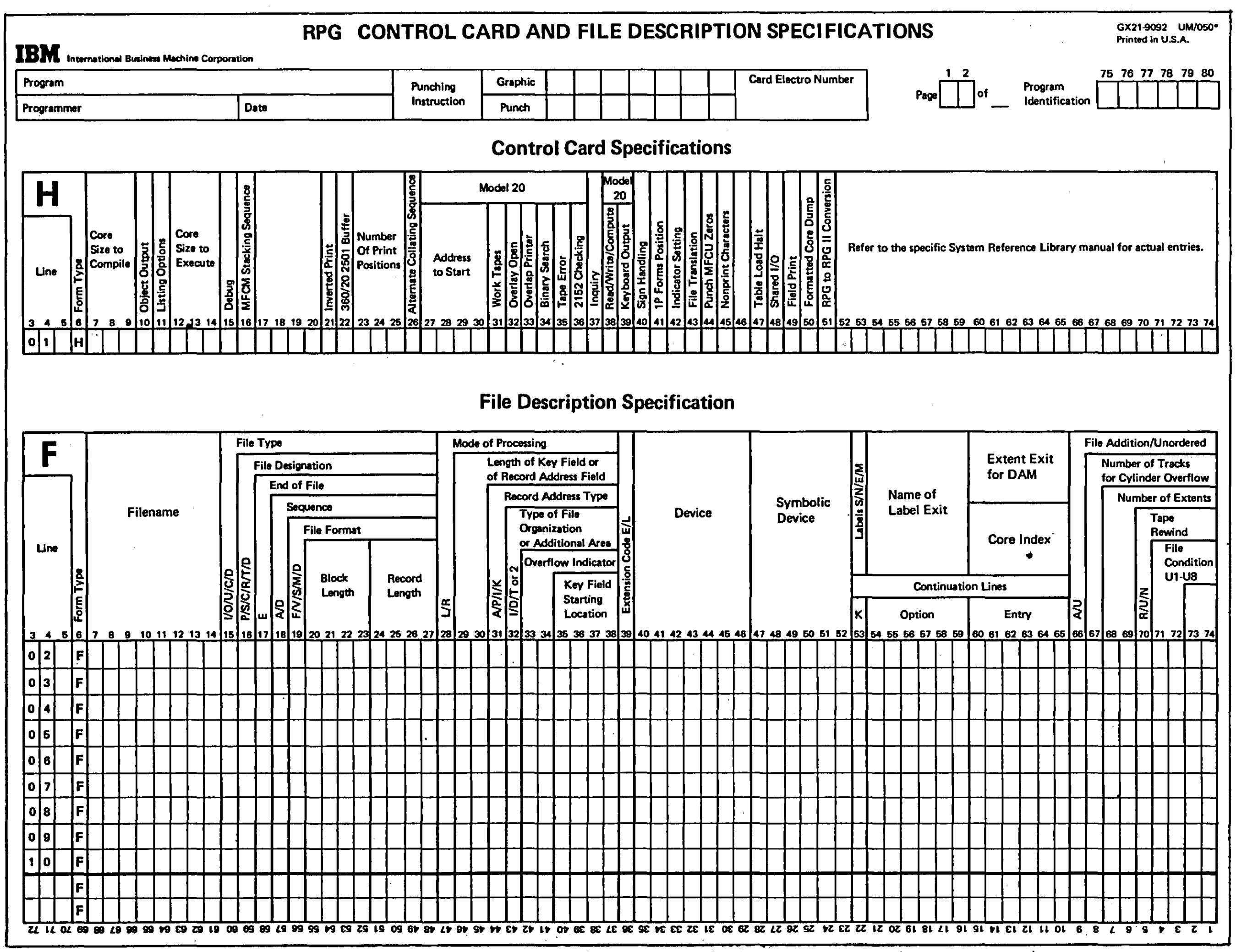


Figure 5. Control Card and File Description Sheet

COLUMN 10 (OBJECT OUTPUT)

Entry	*Explanation*
Blank	Object program is written temporarily in the object library. The system halts only when severe errors are found.
D	Object program is written temporarily in the object library. The system halts for both warning errors and severe errors. The operator can continue the job after a halt occurs for a warning error.
C	Object program is written permanently in the object library.
P	Object program is punched into cards or written on diskette. *Note:* An object program in punched cards cannot be run in level two under Dual Program Feature (Models 10 and 12 only).
R	Place non-link-edited object program in library as a permanent entry (Model 15 only).
T	Place non-link-edited object program in library as a temporary entry (Model 15 only).
B	Punch non-link-edited object program (Model 15 only).

Column 10 is used to indicate the output you want as a result of compiling the source program. The object program is written in the same object library in which the compiler resides, unless specified to another unit via OCL.

You will usually want the object program written temporarily in the object library until you have tested your program. When a program is written permanently in the object library, it deletes all programs temporarily written in the object library. (Every object program written permanently in the object library must be assigned a valid program name in columns 75-80 of the Control Card Specifications Sheet.)

A program identification (columns 75-80) is required when the object program is written permanently in the object library (*C* entry in column 10).

No object program is produced when severe (terminal) errors are present in the source statements.

If NOHALT (with severity 2 on Model 15) is specified on your OCL for this job, no halts will be issued for either warning or terminal errors. When terminal errors are found, end-of-job occurs.

On the Model 15, there are advantages when a non-link-edited object program is obtained and a link edit is performed using the Overlay Linkage Editor. See *IBM System/3 Overlay Linkage Editor Reference Manual,* GC21-7561, for a description of the Overlay Linkage Editor.

When doing a separate link edit using the Overlay Linkage Editor you can:

1. Control the start address of your RPG program.
2. Control the overlay structure of your RPG program.
3. Make changes to assembler subroutines used in your RPG program without recompiling. Another link edit must be performed to include the changed subroutine in the RPG program.

A non-link-edited program may require less disk space in the object library.

COLUMN 11 (LISTING OPTIONS)

Entry	*Explanation*
Blank	1. The object program is produced (if no severe errors are found). 2. The program listing is printed.
B	1. The object program is produced (if no severe errors are found). 2. The program listing is not printed.
P	1. The object program is produced (if no severe errors are found). 2. A partial program listing is printed, which includes the source program, information on indicator usage, and diagnostics.

Column 11 provides for listing options at the time your source program is compiled. If any severe errors are found during compilation, the system halts after completing the listing (provided a listing is to be printed).

The blank entry is the usual case, producing an object program (if no severe errors are found) and a source program listing. The RPG II listing consists of the source program listing, table array information, indicator usage information, the relative locations of fields and their attributes, unreferenced field names, diagnostics, a main storage usage map, and a statement defining the total number of library sectors required for the object program. The main storage usage map lists the identification, the start address, and the size of each uniquely identifiable segment of code in the object program, and defines the amount of main storage required for execution. The main storage map is printed only if the program is successfully compiled.

The B entry means that no program listing is printed; however, an object program is produced. This entry can be used if you want to produce an object program for which you already have a listing.

The P entry means that a partial listing is printed, which includes the source program, information concerning indicator usage, and diagnostics. This can be used if you don't need a complete listing of the program. Excluded from this printout are table/array information, field information, a main storage usage map, and disk storage information.

Note: For Models 10 and 12 only, the compiler forces logging of some error messages to the printer during compilation. The printer is allocated to the program level the compiler is executing in. Any programs executing concurrently in the other program level cannot use the printer for error logging.

COLUMNS 12-14 (CORE SIZE TO EXECUTE)

Column 12

Entry	*Explanation*
Blank,0	No additional 256-byte increments are needed.
Q	One additional 256-byte increment is needed.
H	Two additional 256-byte increments are needed (512 bytes).
T	Three additional 256-byte increments are needed (768 bytes).

Column 12 may be used on Models 10 and 12 to specify additional 256-byte increments of storage. These increments allow an extra 1/4K, 1/2K, or 3/4K of storage to be available in addition to the storage specified in columns 13-14. These additional increments are particularly useful when using the dual programming feature (Models 10 and 12).

Column 12 must be 0 or blank for Model 15.

Example

The following chart shows examples of the possible entries that can be made in columns 12-14 and the amount of storage that would be made available for that entry:

Entry	*Available Bytes*
004	4,096
Q04	4,352 (4,096 + 256)
H04	4,608 (4,096 + 512)
T04	4,864 (4,096 + 768)
005	5,120

Columns 13-14

Entry	*Explanation*
Blank	The main storage required for object program execution is the same as that used to compile the program (maximum of 48K, Model 15).
01-61 (Models 10 and 12) 02-48 (5704-RG1) 02-56 (5704-RG2)	The main storage required for program execution (if different from main storage available for object program generation).

Use columns 13-14 to specify some multiple of 1K bytes (Models 10 and 12) or 2K bytes (Model 15) of storage (K = 1024).

Columns 13-14 define the main storage required for program execution (not including main storage requirements for external buffers, 5704-RG2). The entry must end in column 14.

For Models 10 and 12 RPG II, the size of the object program will be less than, or equal to, the value specified in these columns. For both Model 10 and Model 15 RPG II, if these columns are blank, the size of the object program will be less than, or equal to, the storage size available for compilation. If the total program cannot be contained in the amount of main storage specified, RPG II automatically tries to create overlays.

This entry can differ from the main storage available for object program generation because: (1) your program can be executed on a system other than the one that compiled it, or (2) you might be using the Dual Program Feature on the Models 10 and 12 (see *IBM System/3 RPG II Additional Topics Programmer's Guide,* GC21-7567).

If the system used for program execution is different from that used for compilation, subtract the amount of main storage occupied by the supervisor from the total main storage of the system used for execution (Models 10 and 12 only).

If you are using the Dual Program Feature on the Models 10 and 12, subtract the amount of main storage allocated to the second object program and the supervisor from the total main storage of the system used for program execution. On Model 15, main storage for object code will be allocated in 2K increments. The compiler will diagnose any entry that is not a 2K multiple and round up to the next 2K.

Whether or not an entry is made in these columns, the supervisor size must be considered. Remember, for Models 10 and 12, that the DPF supervisor is larger than the dedicated supervisor. In all cases, even if no entry is made in these columns, the maximum core available to load the programs is the total main storage of the system less the size of the supervisor.

Model 15 programs normally start at address X'4000'. Programs larger than 48K (Program Number 5704-RG2) start at a lower address. See the appropriate SCP reference manual (5704-SC2) listed under *Related Publications* in the Preface for details. Data management and other library routines can be added to the generated object code on Model 15, thus allowing the total program to exceed 64K before overlays are created.

If at any time during compilation the total program size (Models 10 and 12) or generated object code (Model 15) last address exceeds X'FFFF' (65,535 in decimal), the compilation ceases. A terminal halt occurs before an attempt is made by the compiler to generate overlays.

When external buffers are specified (Program Number 5704-RG2), the total storage required for execution is the size specified in columns 13-14, *plus* the size of external buffers rounded to a 2K multiple. (The overlay linkage editor shows the calculated partition size for execution of the program.)

COLUMN 15 (DEBUG)

Entry	*Explanation*
Blank	DEBUG operation is not performed.
1	DEBUG operation is performed.

In order to perform a DEBUG operation:

1. A *1* must appear in column 15 when the source program is compiled.

2. The DEBUG operation code must appear in calculation specifications.

See *Operation Codes, DEBUG Operation* in Chapter 8 for more information.

COLUMNS 16-20

Columns 16-20 are not used.

COLUMN 21 (INVERTED PRINT)

Entry	*Explanation*
Blank	Domestic format.
I	World Trade format.
J	World Trade format (leading zero remains for zero balances).
D	United Kingdom format.

Use column 21 to describe the format and punctuation used for numeric literals in the calculations specifications, the order of the system date (referenced by UDATE) field and edit codes used on output.

Note: The input for UDATE must be in the format expected as output. For example, if D (United Kingdom format) is specified in column 21, the input format must be DD/MM/YY.

Figure 6 shows inverted print specifications and resulting formats.

COLUMNS 22-25

Columns 22-25 are not used.

COLUMN 26 (ALTERNATE COLLATING SEQUENCE)

Entry	*Explanation*
Blank	Normal collating sequence is used.
S	Alternate collating sequence is used.

Use column 26 only to alter the normal collating sequence for a job. Additional specifications are required, as described in the following discussion.

Inverted Print Option	Numeric Literal using Period/Comma as a Decimal Point	Edit Codes using a Period/Comma as a Decimal Point	Zero Suppress to the Left/Right of the Decimal Point	UDATE Appears as a Slash/Period
Blank	4123.57	3,210.89	.50	MM/DD/YY
D	4123.57	3,210.89	.50	DD/MM/YY
I	4123,57	3.210,89	,50	DD.MM.YY
J	4123,57	3.210,89	0,50	DD.MM.YY

Figure 6. Inverted Print Specifications

Collating Sequence

Every alphabetic, numeric, or special character holds a special position in relation to all other characters (see Figure 7 and *Appendix D,* Table D-5). This order is known as the collating sequence. System/3 uses a collating sequence based on the way characters are represented in the machine (see *Character Structure* under *Columns 21-41* in Chapter 4).

You can change this collating sequence if you wish. If you want characters to appear in a sequence other than the one used by System/3, or if you want two or more characters to have the same position in the sequence (this means they are considered equal), you must describe an alternate collating sequence.

Note: An alternate collating sequence applies to matching fields, sequence checking, and alphameric compare operations (COMP). It has no effect on control levels, numeric compares, look up, or sequence checking of tables or arrays.

Defining an Alternate Collating Sequence

To define an alternate collating sequence you must enter an *S* in column 26 of the Control Card Specifications Sheet.

A table also must be entered which lists the changes you wish to make in the normal collating sequence. This is a special table requiring no File Description or Extension Specifications Sheet. The following entries are needed for each table record entered:

Positions 1-6: Enter ALTSEQ to indicate that you are altering the normal sequence.

Positions 7-8: Leave these positions blank.

Positions 9-10: Enter the hexadecimal number of the character whose normal collating sequence is being changed. Table D-5 in *Appendix D* and Figure 7 list characters and their hexadecimal equivalents.

Positions 11-12: Enter the hexadecimal number of the character that is replaced by the character being changed.

Positions 13-16, 17-20, 21-24, etc.: These positions are used in the same way as positions 9-12. There may be as many position entries as the record can contain. Additional records may be used with the above format. The first blank position terminates the record. ** or /* ends the table.

The alternate sequence table must be preceded by a record with **ƀ in positions 1-3. The remaining positions of the record may be used for comments. This table must follow the RPG II specification deck and file translation cards, if used. Figure 3 shows the arrangement of cards in an RPG II source deck.

Translation Table and Alternate Collating Sequence Coding Sheet

The Translation Table and Alternate Collating Sequence Sheet (Figure 7) can be used for coding an alternate collating sequence. It helps you to determine the entries needed for the alternate collating sequence table input records.

Causing Characters To Be Considered Equal

If you want a character to be considered the same as another character, both must hold the same position in the collating sequence. For example, you may want a blank to be considered a zero. Therefore, you need to define an alternate collating sequence in which the blank is the same as the zero because it holds the same position in the sequence. The alternate collating sequence input record looks like this:

Position	*Entry*
1-6	ALTSEQ
7-8	Blanks
9-12	40F0 (blank takes the zero's position)

Whenever a blank is read and used in a compare, it is considered as a zero. Thus, if you were comparing characters to 0036 to find an equal condition, 0036 and ƀƀ36 (where ƀ=blank) both compare equal to 0036.

IBM

International Business Machines Corporation

Form X21-9096
Printed in U.S.A.

TRANSLATION TABLE AND ALTERNATE COLLATING SEQUENCE CODING SHEET

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
00000000		00	
00000001		01	
00000010		02	
00000011		03	
00000100		04	
00000101		05	
00000110		06	
00000111		07	
00001000		08	
00001001		09	
00001010		0A	
00001011		0B	
00001100		0C	
00001101		0D	
00001110		0E	
00001111		0F	
00010000		10	
00010001		11	
00010010		12	
00010011		13	
00010100		14	
00010101		15	
00010110		16	
00010111		17	
00011000		18	
00011001		19	
00011010		1A	
00011011		1B	
00011100		1C	
00011101		1D	
00011110		1E	
00011111		1F	
00100000		20	
00100001		21	
00100010		22	
00100011		23	
00100100		24	
00100101		25	
00100110		26	
00100111		27	
00101000		28	
00101001		29	
00101010		2A	
00101011		2B	
00101100		2C	
00101101		2D	
00101110		2E	
00101111		2F	
00110000		30	
00110001		31	
00110010		32	
00110011		33	
00110100		34	
00110101		35	
00110110		36	
00110111		37	
00111000		38	
00111001		39	
00111010		3A	
00111011		3B	
00111100		3C	
00111101		3D	
00111110		3E	
00111111		3F	
01000000	Blank	40	
01000001		41	
01000010		42	
01000011		43	
01000100		44	
01000101		45	
01000110		46	
01000111		47	
01001000		48	
01001001		49	
01001010	¢	4A	
01001011	.	4B	
01001100	<	4C	
01001101	(	4D	
01001110	+	4E	
01001111	|	4F	
01010000	&	50	
01010001		51	
01010010		52	
01010011		53	
01010100		54	
01010101		55	
01010110		56	
01010111		57	
01011000		58	
01011001		59	
01011010	!	5A	
01011011	$	5B	
01011100	*	5C	
01011101	)	5D	
01011110	;	5E	
01011111	¬	5F	
01100000	-	60	
01100001	/	61	
01100010		62	
01100011		63	
01100100		64	
01100101		65	
01100110		66	
01100111		67	
01101000		68	
01101001		69	
01101010		6A	
01101011	,	6B	
01101100	%	6C	
01101101	–	6D	
01101110	>	6E	
01101111	?	6F	
01110000		70	
01110001		71	
01110010		72	
01110011		73	
01110100		74	
01110101		75	
01110110		76	
01110111		77	
01111000		78	
01111001		79	
01111010	:	7A	
01111011	#	7B	
01111100	@	7C	
01111101	'	7D	
01111110	=	7E	
01111111	"	7F	
10000000		80	
10000001		81	
10000010		82	
10000011		83	
10000100		84	
10000101		85	
10000110		86	
10000111		87	
10001000		88	
10001001		89	
10001010		8A	
10001011		8B	
10001100		8C	
10001101		8D	
10001110		8E	
10001111		8F	
10010000		90	
10010001		91	
10010010		92	
10010011		93	
10010100		94	
10010101		95	
10010110		96	
10010111		97	
10011000		98	
10011001		99	
10011010		9A	
10011011		9B	
10011100		9C	
10011101		9D	
10011110		9E	
10011111		9F	
10100000		A0	
10100001		A1	
10100010		A2	
10100011		A3	
10100100		A4	
10100101		A5	
10100110		A6	
10100111		A7	
10101000		A8	
10101001		A9	
10101010		AA	
10101011		AB	
10101100		AC	
10101101		AD	
10101110		AE	
10101111		AF	
10110000		B0	
10110001		B1	
10110010		B2	
10110011		B3	
10110100		B4	
10110101		B5	
10110110		B6	
10110111		B7	
10111000		B8	
10111001		B9	
10111010		BA	
10111011		BB	
10111100		BC	
10111101		BD	
10111110		BE	
10111111		BF	
11000000		C0	
11000001	A	C1	
11000010	B	C2	
11000011	C	C3	
11000100	D	C4	
11000101	E	C5	
11000110	F	C6	
11000111	G	C7	
11001000	H	C8	
11001001	I	C9	
11001010		CA	
11001011		CB	
11001100		CC	
11001101		CD	
11001110		CE	
11001111		CF	
11010000	}	D0	
11010001	J	D1	
11010010	K	D2	
11010011	L	D3	
11010100	M	D4	
11010101	N	D5	
11010110	O	D6	
11010111	P	D7	
11011000	Q	D8	
11011001	R	D9	
11011010		DA	
11011011		DB	
11011100		DC	
11011101		DD	
11011110		DE	
11011111		DF	
11100000		E0	
11100001		E1	
11100010	S	E2	
11100011	T	E3	
11100100	U	E4	
11100101	V	E5	
11100110	W	E6	
11100111	X	E7	
11101000	Y	E8	
11101001	Z	E9	
11101010		EA	
11101011		EB	
11101100		EC	
11101101		ED	
11101110		EE	
11101111		EF	
11110000	0	F0	
11110001	1	F1	
11110010	2	F2	
11110011	3	F3	
11110100	4	F4	
11110101	5	F5	
11110110	6	F6	
11110111	7	F7	
11111000	8	F8	
11111001	9	F9	
11111010		FA	
11111011		FB	
11111100		FC	
11111101		FD	
11111110		FE	
11111111		FF	

Figure 7. Translation Table and Alternate Collating Sequence Coding Sheet

Altering the Normal Collating Sequence

You can alter the normal collating sequence in several ways. You can insert a character between two existing characters, you can take a character out of the sequence, or you can change characters (put *A* where *Z* is, and *Z* where *A* is). Regardless of how you alter the sequence, you must specify every character to be changed by the alteration. For example, if you want the dollar sign ($) to be positioned in the collating sequence between *A* and *B* , the normal sequence is changed as follows:

Normal Sequence	*Altered Sequence*	*Normal Sequence*	*Altered Sequence*
A	A	F	E
B	$	G	F
C	B	H	G
D	C	I	H
E	D		I

On the Translation Table and Alternate Collating Sequence Coding Sheet, note that there are many characters between *I* and } , *R* and *S*, *Z* and *0*. These characters can be represented in the computer and on records by a certain code. However, they have no printable graphic symbol. Due to this particular arrangement of graphics, nongraphics, graphics, etc. in the collating sequence, a character, when inserted between *A* and *B*, changes only the position of graphics *B* through *I*. All other graphics are not affected. *B* through *I* all move down one position, causing the *I* to take the place of the non-graphic represented by hexadecimal CA. This does not matter, however, since the original character CA cannot be printed anyway. See Figure 8 for the entries on the Translation Table and Alternate Collating Sequence Coding Sheet.

The alternate sequence input record is constructed as follows (this record must be preceded by a record with **b̸ in positions 1-3):

International Business Machines Corporation

Form X21
Printed in

TRANSLATION TABLE AND ALTERNATE COLLATING SEQUENCE CODING SHEET

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
00110011		33	
00110100		34	
00110101		35	
00110110		36	
00110111		37	
00111000		38	
00111001		39	
00111010		3A	
00111011		3B	
00111100		3C	
00111101		3D	
00111110		3E	
00111111		3F	
01000000	Blank	40	
01000001		41	
01000010		42	
01000011		43	
01000100		44	
01000101		45	
01000110		46	
01000111		47	
01001000		48	
01001001		49	
01001010	¢	4A	
01001011	.	4B	
01001100	<	4C	
01001101	(	4D	
01001110	+	4E	
01001111	|	4F	
01010000	&	50	
01010001		51	
01010010		52	
01010011		53	
01010100		54	
01010101		55	
01010110		56	
01010111		57	
01011000		58	
01011001		59	
01011010	!	5A	
01011011	$	5B	C2 (B)
01011100	*	5C	
01011101	)	5D	
01011110	:	5E	
01011111	¬	5F	
01100000	-	60	
01100001	/	61	
01100010		62	
01100011		63	
01100100		64	
01100101		65	

$ takes B's position.

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
01100110		66	
01100111		67	
01101000		68	
01101001		69	
01101010		6A	
01101011	,	6B	
01101100	%	6C	
01101101	—	6D	
01101110	>	6E	
01101111	?	6F	
01110000		70	
01110001		71	
01110010		72	
01110011		73	
01110100		74	
01110101		75	
01110110		76	
01110111		77	
01111000		78	
01111001		79	
01111010	:	7A	
01111011	#	7B	
01111100	@	7C	
01111101	'	7D	
01111110	=	7E	
01111111	"	7F	
10000000		80	
10000001		81	
10000010		82	
10000011		83	
10000100		84	
10000101		85	
10000110		86	
10000111		87	
10001000		88	
10001001		89	
		8A	
		8B	
10001100		8C	
10001101		8D	
10001110		8E	
10001111		8F	
10010000		90	
10010001		91	
10010010		92	
10010011		93	
10010100		94	
10010101		95	
10010110		96	
10010111		97	
10011000		98	

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
10011001		99	
10011010		9A	
10011011		9B	
10011100		9C	
10011101		9D	
10011110		9E	
10011111		9F	
10100000		A0	
10100001		A1	
10100010		A2	
10100011		A3	
10100100		A4	
10100101		A5	
10100110		A6	
10100111		A7	
10101000		A8	
10101001		A9	
10101010		AA	
10101011		AB	
10101100		AC	
10101101		AD	
10101110		AE	
10101111		AF	
10110000		B0	
10110001		B1	
10110010		B2	
10110011		B3	
10110100		B4	
10110101		B5	
10110110		B6	
10110111		B7	
10111000		B8	
10111001		B9	
10111010		BA	
10111011		BB	
10111100		BC	
10111101		BD	
10111110		BE	
10111111		BF	
11000000		C0	
11000001	A	C1	
11000010	B	C2	C3 (C)
11000011	C	C3	C4 (D)
11000100	D	C4	C5 (E)
11000101	E	C5	C6 (F)
11000110	F	C6	C7 (G)
11000111	G	C7	C8 (H)
11001000	H	C8	C9 (I)
11001001	I	C9	CA
11001010		CA	
11001011		CB	

B takes C's position.

C takes D's position.

(no printable character)

Code	System/3 Graphic	Entry	Re By. Pla
11001100		CC	
11001101		CD	
11001110		CE	
11001111		CF	
11010000	}	D0	
11010001	J	D1	
11010010	K	D2	
11010011	L	D3	
11010100	M	D4	
11010101	N	D5	
11010110	O	D6	
11010111	P	D7	
11011000	Q	D8	
11011001	R	D9	
11011010		DA	
11011011		DB	
11011100		DC	
11011101		DD	
11011110		DE	
11011111		DF	
11100000		E0	
11100001		E1	
11100010	S	E2	
11100011	T	E3	
11100100	U	E4	
11100101	V	E5	
11100110	W	E6	
11100111	X	E7	
11101000	Y	E8	
11101001	Z	E9	
11101010		EA	
11101011		EB	
11101100		EC	
11101101		ED	
11101110		EE	
11101111		EF	
		F0	
		F1	
11110010	2	F2	
11110011	3	F3	
11110100	4	F4	
11110101	5	F5	
11110110	6	F6	
11110111	7	F7	
11111000	8	F8	
		9	
		A	
11111011		FB	
11111100		FC	
11111101		FD	
11111110		FE	
		FF	

Figure 8. Altering the Collating Sequence

Position	*Entry*
1-6	ALTSEQ
7-8	(blanks)
9-12	5BC2 ($ takes B's position)
13-16	C2C3 (B takes C's position)
17-20	C3C4 (C takes D's position)
21-24	C4C5 (D takes E's position)
25-28	C5C6 (E takes F's position)
29-32	C6C7 (F takes G's position)
33-36	C7C8 (G takes H's position)
37-40	C8C9 (H takes I's position)
41-44	C9CA (I takes a new position held by no other printable character)

COLUMNS 27-36

Columns 27-36 are not used by System/3.

COLUMN 37 (INQUIRY)

Entry	*Explanation*
Blank	The program cannot be interrupted (does not recognize an inquiry request).
B	The program can be interrupted (recognizes an inquiry request). RPG II 5704-RG2 does not support B-type programs.
I	The program is an inquiry program that can only be executed when an inquiry request is made.

System/3 allows certain programs to be interrupted while they are being processed. A request for interruption is called an inquiry request and is handled by the rollout/rollin support of the supervisor. (For Models 10 and 12, made by pressing the REQ key on the printer-keyboard; for Model 15 using SCP 5704-SC1, made by entering a ROLLOUT OCC command. See *IBM System/3 Model 15 System Control Programming Reference Manual,* GC21-5077).

An I-type program is usually read in only when a B-type program is interrupted. In this case, the I-type program does not recognize an inquiry request. However, if an I-type program is loaded in the normal manner (not because of a program interrupt), input from the console on the Model 10 is accepted after an inquiry request has been made. On the Model 15, input from the CRT/Keyboard is accepted after the PA1 key has been pressed. For Model 15, if CRT77 input is not specified, the program is loaded and executed as if column 37 was blank.

The RPG II inquiry request is outlined in these steps:

1. Only a B-type program recognizes an inquiry or rollout request.
2. When the program recognizes an inquiry or rollout request, a rollout routine moves the interrupted program from main storage to disk.
3. The program for which the interrupt was requested is processed. The interrupting program may be any type (blank, B, or I). This interrupting program cannot be interrupted.
4. After the interrupting program is executed, the interrupted program moves back into main storage using a rollin routine. The interrupted program resumes execution at the point of interruption and terminates in a normal manner.

On the Models 10 and 12 (in dual program mode) or Model 15 (with both partitions active), the same specifications apply. However, only level 1 programs can be interrupted and moved out of main storage by a rollout routine.

Note: An inquiry request can also be made by using IBM-written subroutines (SUBR95 (Models 10 and 12) or SUBR89 (Model 15) instead of rollout/rollin. For information on this method see Appendix J. When these subroutines are used, it is not necessary to supply a code in column 37 of the control card.

COLUMNS 38-40

Columns 38-40 are not used by System/3.

COLUMN 41 (1P FORMS POSITION)

Entry	*Explanation*
Blank	First 1P line is printed only once.
1	First 1P line can be printed repeatedly.

When forms are first inserted in the printer, they may not always be in perfect alignment. Sometimes several lines must be printed to determine the correct positioning of the form. Since you may not want to print several lines of a report before getting the forms positioned correctly, you have the option of repeatedly printing the first line conditioned by the first page (1P) indicator. Each time the 1P line is printed, the program halts so you may reposition the forms if needed. Forms positioning applies to the first 1P output line for the first printer file. Page count is not incremented until the forms are positioned correctly.

COLUMN 42 (INDICATOR SETTING)

Column 42 is not used.

COLUMN 43 (FILE TRANSLATION)

Entry	*Explanation*
Blank	No file translation is needed.
F	Input, output, update, or combined files are to be translated.

Use column 43 only when information contained in an input, output, combined, or update file is in a form which requires translation. When file translation is specified for an update or combined file, both the input and output portions of the file are translated. In this discussion, input and output characters are referred to as *external* characters; characters used for processing within System/3 are called *internal* characters.

An *F* in column 43 indicates either or both of the following:

1. The character code used in the input data (external character) must be translated into a form that can be used by your program (internal character).

2. The output data must be in a character code different from that used by your program.

FILE TRANSLATION

RPG II allows you to translate any character code into another character code. This capability is file translation.

A different character code used as input can be translated into the code used by System/3, and the code used by System/3 can be translated into a different code for output.

Specifications for File Translation

To indicate that there are files to be translated, enter an *F* in column 43 of the RPG II Control Card Specifications Sheet. File translate table records must also be used to specify how the translation is to be done. The following entries are needed for each file translation table record used:

Positions 1-6: Enter *FILES to indicate that *all* input, output, update, and combined files are to undergo translation (both the input and output portions of update and combined files will be translated). Then use the specifications listed below, beginning with positions 9-10. All files will be translated according to the translate table specified beginning in position 9.

If only *certain* files are to be translated, you need not specify *FILES, but you must name the files to be translated (in columns 1-8) as follows:

Positions 1-8: Enter the filename of the input, output, update, or combined file to be translated (both the input and output portions of update and combined files will be translated). Then use the following specifications, beginning with positions 9-10.

Positions 9-10: Enter the hexadecimal equivalent of the external character. This is the character in a different character code to be translated from input data or for output data.

Positions 11-12: Enter the hexadecimal equivalent of the internal character. This is the character in the System/3 code which internally represents the external input or output character.

Positions 13-16, 17-20, and 21-24, etc: These groups of positions are used the same way as positions 9-12. The first two positions of a group give the character to be translated into the character named in the last two positions of a group. All tables for one file must be kept together. The file translation table input records must be preceded by a record with **ƀ in positions 1-3. The remaining positions of this record may be used for comments. The file translation records must directly follow the RPG II specifications in the source program (Figure 3).

Example

Assume that a department store must process cards serving as sales slips for all items sold. Each card contains a punched and printed record of the actual, or wholesale, cost of its associated item along with a retail price.

Since wholesale cost is confidential, the store uses individual letters of a code name in place of wholesale cost figures.

A typical code name consists of a combination of letters that can be easily remembered by the store's personnel. The only restriction, however, is that the code name must contain ten different letters, one for each of the numbers zero through nine.

Using the code name BUCKINGHAM to represent numbers one through nine and zero, the letter B represents the number 1; letter U represents number 2, etc. Letter M represents zero. Individual letters are combined to represent each item's wholesale cost. Thus a wholesale cost of BBU.CC translates as 112.33; that is, one hundred twelve dollars and thirty-three cents.

In the following chart, hexadecimal equivalents of each letter in the word BUCKINGHAM are listed along with the hexadecimal equivalents of numbers one through nine and zero.

Letter in Code name	*Hexadecimal Equivalent*	*Number*	*Hexadecimal Equivalent*
B	C2	1	F1
U	E4	2	F2
C	C3	3	F3
K	D2	4	F4
I	C9	5	F5
N	D5	6	F6
G	C7	7	F7
H	C8	8	F8
A	C1	9	F9
M	D4	0	F0

Hexadecimal equivalents are merely a different way of representing the 8-bit code that the computer examines to recognize individual characters in your language.

See Figure 9. Note that if letters BBU were read and never translated, hexadecimal equivalents C2, C2, and E4 would be used by System/3. As a result, it would be impossible to perform an arithmetic operation involving the wholesale cost, BBU. Therefore, with the aid of file translation, the computer replaces the letters BBU with numbers.

File translation table input card specifications for letters in the word BUCKINGHAM are as follows:

Column	*Entry*
1-6	*FILES
7-8	Blank
9-12	C2F1
13-16	E4F2
17-20	C3F3
21-24	D2F4
25-28	C9F5
29-32	D5F6
33-36	C7F7
37-40	C8F8
41-44	C1F9
45-48	D4F0

Only the letters of the previous example will be specified for translation. All other characters will be handled in the normal manner. Figure 10 shows the entries made on the Translation Table and Alternate Collating Sequence Coding Sheet for the previous example.

COLUMN 44 (PUNCH MFCU ZEROS)

Entry	*Explanation*
Blank	Leading zeros are removed.
1	Leading zeros are used.

TRANSLATION TABLE AND ALTERNATE COLLATING SEQUENCE CODING SHEET

Replaced By/Takes Place Of

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
01100110		66	
01100111		67	
01101000		68	
01101001		69	
01101010		6A	
01101011	,	6B	
01101100	%	6C	
01101101	–	6D	
01101110	>	6E	
01101111	?	6F	
01110000		70	
01110001		71	
01110010		72	
01110011		73	
01110100		74	
01110101		75	
01110110		76	
01110111		77	
01111000		78	
01111001		79	
01111010	:	7A	
01111011	#	7B	
01111100	@	7C	
01111101	'	7D	
01111110	=	7E	
01111111	"	7F	
10000000		80	
10000001		81	
10000010		82	
10000011		83	
10000100		84	
10000101		85	
10000110		86	
10000111		87	
10001000		88	
10001001		89	
10001010		8A	
10001011		8B	
10001100		8C	
10001101		8D	
10001110		8E	
10001111		8F	
10010000		90	
10010001		91	
10010010		92	
10010011		93	
10010100		94	
10010101		95	
10010110		96	
10010111		97	
10011000		98	

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
10011001		99	
10011010		9A	
10011011		9B	
10011100		9C	
10011101		9D	
10011110		9E	
10011111		9F	
10100000		A0	
10100001		A1	
10100010		A2	
10100011		A3	
10100100		A4	
10100101		A5	
10100110		A6	
10100111		A7	
10101000		A8	
10101001		A9	
10101010		AA	
10101011		AB	
10101100		AC	
10101101		AD	
10101110		AE	
10101111		AF	
10110000		B0	
10110001		B1	
10110010		B2	
10110011		B3	
10110100		B4	
10110101		B5	
10110110		B6	
10110111		B7	
10111000		B8	
10111001		B9	
10111010		BA	
10111011		BB	
10111100		BC	
10111101		BD	
10111110		BE	
10111111		BF	
11000000		C0	
11000001	A	C1	
11000010	B	C2	
11000011	C	C3	
11000100	D	C4	
11000101	E	C5	
11000110	F	C6	
11000111	G	C7	
11001000	H	C8	
11001001	I	C9	
11001010		CA	
11001011		CB	

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
11001100		CC	
11001101		CD	
11001110		CE	
11001111		CF	
11010000	}	D0	
11010001	J	D1	
11010010	K	D2	
11010011	L	D3	
11010100	M	D4	
11010101	N	D5	
11010110	O	D6	
11010111	P	D7	
11011000	Q	D8	
11011001	R	D9	
11011010		DA	
11011011		DB	
11011100		DC	
11011101		DD	
11011110		DE	
11011111		DF	
11100000		E0	
11100001		E1	
11100010	S	E2	
11100011	T	E3	
11100100	U	E4	
11100101	V	E5	
11100110	W	E6	
11100111	X	E7	
11101000	Y	E8	
11101001	Z	E9	
11101010		EA	
11101011		EB	
11101100		EC	
11101101		ED	
11101110		EE	
11101111		EF	
11110000	0	F0	
11110001	1	F1	
11110010	2	F2	
11110011	3	F3	
11110100	4	F4	
11110101	5	F5	
11110110	6	F6	
11110111	7	F7	
11111000	8	F8	
11111001	9	F9	
11111010		FA	
11111011		FB	
11111100		FC	
11111101		FD	
11111110		FE	
11111111		FF	

E4, which if translated would represent the number 2, is the letter U in the code used by the System/3.

C2, which if translated would represent the number 1, is the letter B in the code used by the System/3.

Figure 9. Differences in Character Codes

This column applies only to output on the MFCU. If the column is left blank, all numeric output fields on the MFCU will be zero suppressed to the units position. Enter a *1* in column 44 when you wish to have leading zeros on fields punched or printed by the MFCU.

If an edit word or edit code is defined for fields to be printed or punched on the MFCU, the edit word or code will override column 44.

COLUMN 45 (NONPRINT CHARACTERS)

Entry	*Explanation*
Blank	Program halts if an unprintable character was in the last line printed.
1	No program halt for such unprintable characters.

International Business Machines Corporation

Form X21-9096
Printed in U.S.A.

TRANSLATION TABLE AND ALTERNATE COLLATING SEQUENCE CODING SHEET

de	System/3 Graphic	Entry	Replaced By/Takes Place Of
110011		33	
110100		34	
110101		35	
110110		36	
110111		37	
111000		38	
111001		39	
111010		3A	
111011		3B	
111100		3C	
111101		3D	
111110		3E	
111111		3F	
000000	Blank	40	
000001		41	
000010		42	
000011		43	
000100		44	
000101		45	
000110		46	
000111		47	
001000		48	
001001		49	
001010	¢	4A	
001011	.	4B	
001100	<	4C	
001101	(	4D	
001110	+	4E	
001111	|	4F	
010000	&	50	
010001		51	
010010		52	
010011		53	
010100		54	
010101		55	
010110		56	
010111		57	
011000		58	
011001		59	
011010	!	5A	
011011	$	5B	
011100	*	5C	
011101	)	5D	
011110	;	5E	
011111	¬	5F	
100000	-	60	
100001	/	61	
100010		62	
100011		63	
100100		64	
100101		65	

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
01100110		66	
01100111		67	
01101000		68	
01101001		69	
01101010		6A	
01101011	,	6B	
01101100	%	6C	
01101101	_	6D	
01101110	>	6E	
01101111	?	6F	
01110000		70	
01110001		71	
01110010		72	
01110011		73	
01110100		74	
01110101		75	
01110110		76	
01110111		77	
01111000		78	
01111001		79	
01111010	:	7A	
01111011	#	7B	
01111100	@	7C	
01111101	'	7D	
01111110	=	7E	
01111111	"	7F	
10000000		80	
10000001		81	
10000010		82	
10000011		83	
10000100		84	
10000101		85	
10000110		86	
10000111		87	
10001000		88	
10001001		89	
10001010		8A	
10001011		8B	
10001100		8C	
10001101		8D	
10001110		8E	
10001111		8F	
10010000		90	
10010001		91	
10010010		92	
10010011		93	
10010100		94	
10010101		95	
10010110		96	
10010111		97	
10011000		98	

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
10011001		99	
10011010		9A	
10011011		9B	
10011100		9C	
10011101		9D	
10011110		9E	
10011111		9F	
10100000		A0	
10100001		A1	
10100010		A2	
10100011		A3	
10100100		A4	
10100101		A5	
10100110		A6	
10100111		A7	
10101000		A8	
10101001		A9	
10101010		AA	
10101011		AB	
10101100		AC	
10101101		AD	
10101110		AE	
10101111		AF	
10110000		B0	
10110001		B1	
10110010		B2	
10110011		B3	
10110100		B4	
10110101		B5	
10110110		B6	
10110111		B7	
10111000		B8	
10111001		B9	
10111010		BA	
10111011		BB	
10111100		BC	
10111101		BD	
10111110		BE	
10111111		BF	
11000000		C0	
11000001	A	C1	F9
11000010	B	C2	F1
11000011	C	C3	F3
11000100	D	C4	
11000101	E	C5	
11000110	F	C6	
11000111	G	C7	F7
11001000	H	C8	F8
11001001	I	C9	F5
11001010		CA	
11001011		CB	

Code	System/3 Graphic	Entry	Replaced By/Takes Place Of
11001100		CC	
11001101		CD	
11001110		CE	
11001111		CF	
11010000	}	D0	
11010001	J	D1	
11010010	K	D2	F4
11010011	L	D3	
11010100	M	D4	FØ
11010101	N	D5	F6
11010110	O	D6	
11010111	P	D7	
11011000	Q	D8	
11011001	R	D9	
11011010		DA	
11011011		DB	
11011100		DC	
11011101		DD	
11011110		DE	
11011111		DF	
11100000		E0	
11100001		E1	
11100010	S	E2	
11100011	T	E3	
11100100	U	E4	F2
11100101	V	E5	
11100110	W	E6	
11100111	X	E7	
11101000	Y	E8	
11101001	Z	E9	
11101010		EA	
11101011		EB	
11101100		EC	
11101101		ED	
11101110		EE	
11101111		EF	
11110000	0	F0	
11110001	1	F1	
11110010	2	F2	
11110011	3	F3	
11110100	4	F4	
11110101	5	F5	
11110110	6	F6	
11110111	7	F7	
11111000	8	F8	
11111001	9	F9	
11111010		FA	
11111011		FB	
11111100		FC	
11111101		FD	
11111110		FE	
11111111		FF	

This is the hexadecimal equivalent of the character to be translated.

This is the hexadecimal equivalent of the System/3 character that will be substituted for the character that is to be translated.

Figure 10. Specifications for File Translation Input Cards

Column 45 is used to bypass machine halts for unprintable characters. This column applies to the printer and the printer keyboard. All characters are known to the system by a numeric code. If a numeric code is formed which is not known to your system (not in your character set) and that character is to be printed, the machine will halt after printing the line. The unprintable characters will have been replaced by blanks.

If you wish to bypass this halt, enter a one (1) in column 45. An unprintable character will be printed as a blank and no halt will occur. Note, however, that this option could make some types of output data meaningless.

COLUMNS 46-47

Columns 46-47 are not used.

COLUMN 48 (SHARED I/O AREA; EXTERNAL BUFFERS FOR PROGRAM NUMBER 5704-RG2 ONLY)

Entry	*Explanation*
1	All 5444 disk files share a single input/output area (Program Number 5702-RG1 only).
E	External buffers for disk devices (Program Number 5704-RG2 only).
Blank	All disk files use a separate input/output area.

For Model 10 5444 disk files, enter a 1 in this column to indicate that all disk files in the program share a single input/output area. For Program Number 5704-RG2, enter an E in this column to specify external buffers for disk devices. This option causes all disk I/O buffers to be outside the generated object program. This allows the user program to be up to 52K long excluding the I/O buffers. When the object program is executed, the user must allocate a partition large enough to contain the object program and the disk buffers. For more information about external buffers, see the appropriate SCP reference manual or overlay linkage editor manual listed under *Related Publications* in the Preface.

Normally an RPG II program uses one input/output area for each file. A 1 in column 48 allows all 5444 disk files to use one input/output area. By specifying a shared input/output area, you can reduce the amount of main storage needed to process a program. This is particularly important if a program is so large that it cannot run in the main storage you have available. However, the use of a shared input/output area increases the time required to process your program. Therefore, before you indicate that all disk files are to share one input/output area, be sure that the program would otherwise exceed the capacity of the system.

Note 1: A shared input/output area cannot be specified for multivolume files (entry greater than 01 in columns 68-69 of the File Description sheet).

Note 2: Additional input/output areas (entry in column 32 of the File Description Sheet) cannot be specified for disk files using a shared input/output area.

COLUMNS 49-74

Columns 49-74 are not used. Leave them blank.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

See Chapter 2.

File description specifications are required for every file used by a program. Write these specifications on the Control Card and File Description Sheet (Figure 11). Only one line is needed to describe a file. A maximum of 20 file description records are allowed per program.

At the end of this chapter is a series of charts showing all possible files that can be defined on the File Description Sheet. The charts are arranged by device, showing the basic entries for all possible disk, card, console, and printer files.

COLUMNS 1-2 (PAGE)

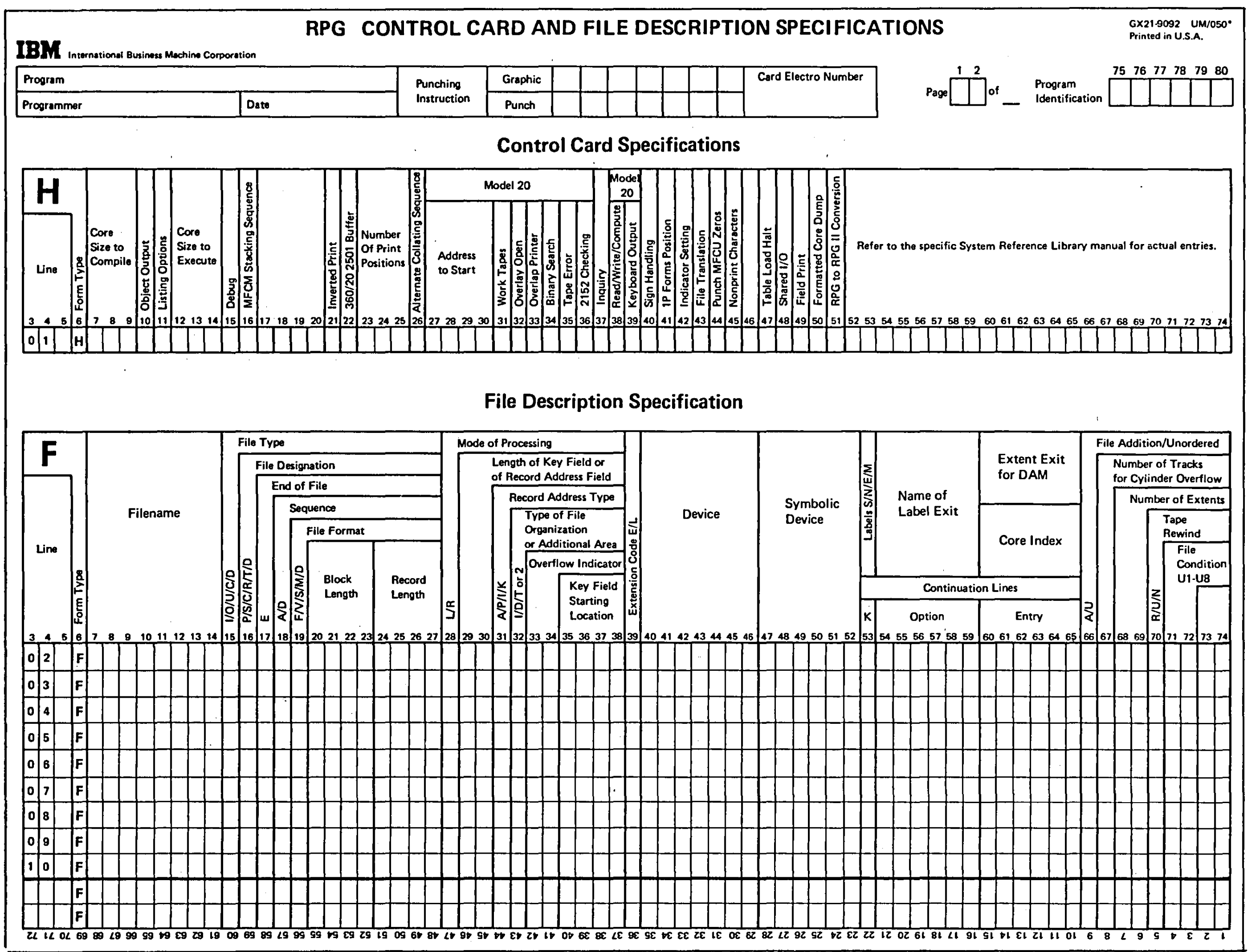
See Chapter 2.

COLUMNS 3-5 (LINE)

See Chapter 2.

COLUMN 6 (FORM TYPE)

An *F* must appear in column 6.


IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 UM/050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page of | Program Identification

Control Card Specifications

H | Line | Form Type | Core Size to Compile | Object Output | Listing Options | Core Size to Execute | Debug | MFCM Stacking Sequence | Inverted Print | 360/20 2501 Buffer | Number Of Print Positions | Alternate Collating Sequence | Model 20: Address to Start, Work Tapes, Overlap Open, Overlap Printer, Binary Search, Tape Error, 2152 Checking | Model 20: Inquiry, Read/Write/Compute, Keyboard Output | Sign Handling | 1P Forms Position | Indicator Setting | File Translation | Punch MFCU Zeros | Nonprint Characters | Table Load Halt | Shared I/O | Field Print | Formatted Core Dump | RPG to RPG II Conversion | Refer to the specific System Reference Library manual for actual entries.

0 1 H

File Description Specification

F | Line | Form Type | Filename | File Type: I/O/U/C/D | File Designation: P/S/C/R/T/D | End of File: E | Sequence: A/D | File Format: F/V/S/M/D | Block Length | Record Length | L/R | Mode of Processing | Length of Key Field or of Record Address Field | Record Address Type: A/P/I/K | Type of File Organization or Additional Area: I/D/T or 2 | Overflow Indicator | Key Field Starting Location | Extension Code E/L | Device | Symbolic Device | Labels S/N/E/M | Name of Label Exit | Extent Exit for DAM | Core Index | Continuation Lines: K, Option, Entry | A/U | File Addition/Unordered | Number of Tracks for Cylinder Overflow | Number of Extents | Tape Rewind: R/U/N | File Condition U1-U8

0 2 F
0 3 F
0 4 F
0 5 F
0 6 F
0 7 F
0 8 F
0 9 F
1 0 F
F
F

Figure 11. Control Card and File Description Sheet

COLUMNS 7-14 (FILENAME)

Use columns 7-14 to assign a unique filename to every file used in your program except compile time table and array files, which must not be named on the File Description Sheet. (Compile time tables and arrays are described on the Extension Sheet.) The filename can be from 1-8 characters long, must begin in column 7, and must be a valid RPG II name. The filename can be the same as a field name.

Pre-execution time table and array files are described on the File Description Sheet. More than one table or array file can be described for the same device (see columns 40-46 in this chapter). For the MFCU (but not for other devices), a single file may contain more than one table or array. In this case, the MFCU file would be named only once on the File Description Sheet, but each table or array within the file would be described separately on the Extension Sheet (see *Tables and Arrays* in Chapter 5).

COLUMN 15 (FILE TYPE)

Entry	*Explanation*
I	Input file
O	Output file
U	Update file
C	Combined file
D	Display file

Use column 15 to identify the way in which your program uses the file.

Input File

Input files are records that a program uses as a source of data. When input files are described in a program it indicates that records are to be read from the file. All input files except table and record address files must be further described on the Input Sheet. Table files and record address files must be further described in the Extension Sheet.

Output Files

Output files are records that are written, punched, or printed by a program. All output files, except table and array output files, must be further described on the Output Sheet.

Update Files

Update files are disk files from which a program reads a record, updates fields in the record, and writes the record back in the location from which it was read. Update files must be further described on both the Input Sheet and Output Sheet; only the fields to be updated must be described on the Output Sheet. A record in a update file can be updated only once during a cycle (see *output file output indicator description*). A chained file or a demand file may be updated at detail time or at total time or exception time. All other disk files should be updated only at detail time, during the same program cycle that reads the record, otherwise, results of the update will be unpredictable.

Model 15 Systems: The CRT/keyboard can be specified as an update file. This gives you the flexibility of displaying fields or constants on the CRT during output and responding to, altering, or adding to the fields or constants during input. See Figure 12 for an example and refer to *CRT/Keyboard Files (Model 15)* in Chapter 4 for a further description.

Combined Files

A combined file is both an input and an output file. For Models 10 and 12, a combined file can be assigned only to the MFCU or 1442 Card Read Punch. For Model 15, MFCU, MFCM, or 1442 Card Read Punch files can be combined files. A program reads records from a combined file and includes output data on the records in the file. The result is one file that contains both input and output data. Combined files must be further described on both the Input Sheet and Output Sheet.

Output data can be printed or punched on cards as they are read.

Do not condition output to a combined file such that more than one record can satisfy the output conditions during the same cycle. (This condition should not exist for the entire detail output cycle, even if output is going to more than one combined file.)

All output records to a combined file are stored in a hold area until another record is read from the combined file. The output record is printed or punched just prior to the time the new record is read.

Since each record stored in the hold area overlays and replaces any record previously stored in the hold area, only one record can be available as output to a combined file during any one RPG II cycle. This record will be the last record that satisfied output conditions during the cycle.

File Description Specification

Line	Form Type	Filename	File Type (I/O/U/C/D)	File Designation (P/S/C/R/T/D)	File Format (F/V/S/M/D)	Block Length	Record Length	Device
02	F	DISFLE	U	P	F	279	279	CRT77
03	F							
04	F							

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050*
Printed in U.S.A.

IBM International Business Machine Corporation

Program | Punching Instruction | Graphic | Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80
Programmer | Date

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Field Location From	Field Location To	Field Name
01	I	DISFLE	AA	01			
02	I				126	133	DATE
03	I				166	200	NAME
04	I				215	240	STREET
05	I				256	279	ADDRES

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050*
Printed in U.S.A.

IBM International Business Machine Corporation

Program | Punching Instruction | Graphic | Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80
Programmer | Date

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators	End Position in Output Record	Constant or Edit Word
01	O	DISFLE	D	1P		
02	O		OR	01		
03	O				24	'THIS IS AN EXAMPLE USING'
					48	'THE CRT/KEYBOARD AS AN U'
					59	'PDATE FILE.'
					100	'ENTER THE FOLLOWING:'
					133	'DATE-DD/MM/YY'
					165	'NAME-'
					214	'STREET NUMBER-'
					255	'CITY/STATE/ZIP-'

Edit codes: Commas / Zero Balances to Print / No Sign / CR / -: Yes Yes 1 A J; Yes No 2 B K; No Yes 3 C L; No No 4 D M. X = Remove Plus Sign; Y = Date Field Edit; Z = Zero Suppress.

The following is displayed on the CRT at output time.

```
THIS IS AN EXAMPLE USING THE CRT/KEYBOARD
AS AN UPDATE FILE.
ENTER THE FOLLOWING:
DATE—DD/MM/YY
NAME—
STREET NUMBER—
CITY/STATE/ZIP—
```

At input time the operator enters the information called for. Any data displayed can be changed at this time. All data displayed plus new data entered will be available for processing.

Figure 12. Coding a CRT file as an update file

Display Files

A display file is a collection of information from fields used by a program. The DSPLY operation code must be used on the Calculation Sheet in order to print a field or record directly from storage and/or key data into a field or record in storage. Display files need only be described on the File Description Sheet. The device associated with a display file for a Model 10 or 12 must be a printer-keyboard (CONSOLE). For a Model 15, it must be the CRT/keyboard (CRT77). See *Operation Codes, Display* in Chapter 8 for more information.

COLUMN 16 (FILE DESIGNATION)

Entry	*Explanation*
P	Primary file
S	Secondary file
C	Chained file
R	Record address file
T	Table or array file (pre-execution time tables or arrays)
D	Demand file

Use column 16 to further identify the use of input, update, combined, and chained output files. Leave the column blank for display files and all output files except chained output files (direct load).

Primary Files

A primary file is the main file from which a program reads records. In multifile processing the primary file is used to control the order in which records are selected for processing. It can be an input, update, or combined file. In programs that read records from only one file, that file is the primary file. Every program must have one, and only one, primary file.

Secondary Files

Secondary files apply to programs that do multifile processing. All of the files involved in multifile processing, except the primary file, are secondary files. A secondary file can be an input, update, or combined file. Secondary files are processed in the order in which they are written in the file description specifications.

Note that table, chained, record address, and demand files are not involved in record selection in multifile processing.

See *Multifile Processing* (columns 61-62) in Chapter 7 for more information on primary and secondary files.

Chained Files

A chained file is a disk file that is read randomly or loaded directly via the CHAIN operation code. A maximum of 15 chained and/or demand files are allowed per program.

A chained file can be an input, output, or update file. See *Column 28 (Mode of Processing), Random* in this chapter, and *Operation Codes, CHAIN* in Chapter 8.

Record Address Files

A record address file is an input file that indicates which records are to be read from a disk file and the order in which the records are to be read from the disk file. You cannot use more than one record address file in a program. All record address files must be further defined in extension specifications.

Record address files contain either record key limits or relative record numbers in binary format. Record address files that contain record key limits can be disk files, card files, tape files, diskette files or can be entered from the printer-keyboard (Models 10 and 12) or CRT/keyboard (Model 15).

Record address files that contain binary relative record numbers can only be disk or tape files. Those files that contain limits are used with indexed files only. See *Column 28 (Mode of Processing), Sequential Within Limits* in this chapter for more information.

Record address files on disk that contain binary relative record numbers are called ADDROUT (address output) files. They are produced by the Disk Sort Program and can be used with any type of disk file. See *Column 28 (Mode of Processing), By ADDROUT File* in this chapter for more information.

Table or Array Files

A table or array file is a sequential input file that contains table or array entries. The entries can be read into the program during compilation or immediately before execution of the program. Only pre-execution time tables or arrays are described on the File Description Sheet. However, both pre-execution and compile time tables and arrays must be described in the Extension Sheet.

A table or array output file (written or punched after LR output) is defined as a normal output file and does not require an entry in column 16.

Table and array files are not involved in record selection and processing. They are only a means of supplying entries for tables or arrays used by the program. When table or array files are read during the execution of the program, the program reads all the entries from the table or array files before it begins record processing. See *Tables and Arrays* in Chapter 5 for additional information.

Demand Files

Demand files can be input, update, or combined files. The READ operation code must be used on the Calculation Sheet in order to read from a demand file. Demand files can be processed either sequentially by key or consecutively. A maximum of 15 demand and/or chained files are allowed per program. See *Operation Codes, READ* in Chapter 8 for a discussion of processing demand files. If a demand file is assigned to the same device from which the OCL is read, the last OCL record may be placed in the wrong stacker.

COLUMN 17 (END OF FILE)

Entry	*Explanation*
E	All records from the file must be processed before the program can end.
Blank	1. The program can end whether or not all of the records from the file have been processed. 2. If column 17 is blank for all of the files, all records from every file must be processed before the program can end.

Column 17 applies to programs that perform multifile processing. Use it to indicate whether or not the program can end before all of the records from the file are processed. It applies only to input, update, and combined files that are used as primary, secondary, or record address files.

A program that performs multifile processing could reach the end of one file before reaching the end of the others. It therefore needs some indication of whether it is to continue reading records from the other files or end the program. An entry in column 17 in the descriptions of the files provides that indication.

If the records from all the files must be processed, column 17 must be blank for all files, or contain *E*'s for all files.

End-of-File Processing

By specifying an *E* in column 17 of the File Description Sheet, you indicate that the job is to end after all records are processed from the file for which you specified the *E*. In most cases, the job will end at the time all records from that file are processed. However, under certain conditions additional records may be processed after all records from the file with the *E* designation are processed. The exceptional situation is in matching records when an *E* is designated for the primary file and all records from that file have been processed. The job will end only after all secondary records that match the last primary record have been processed or the first secondary record without a match field has been encountered.

Figure 13 shows the records that will be processed for various end-of-file situations.

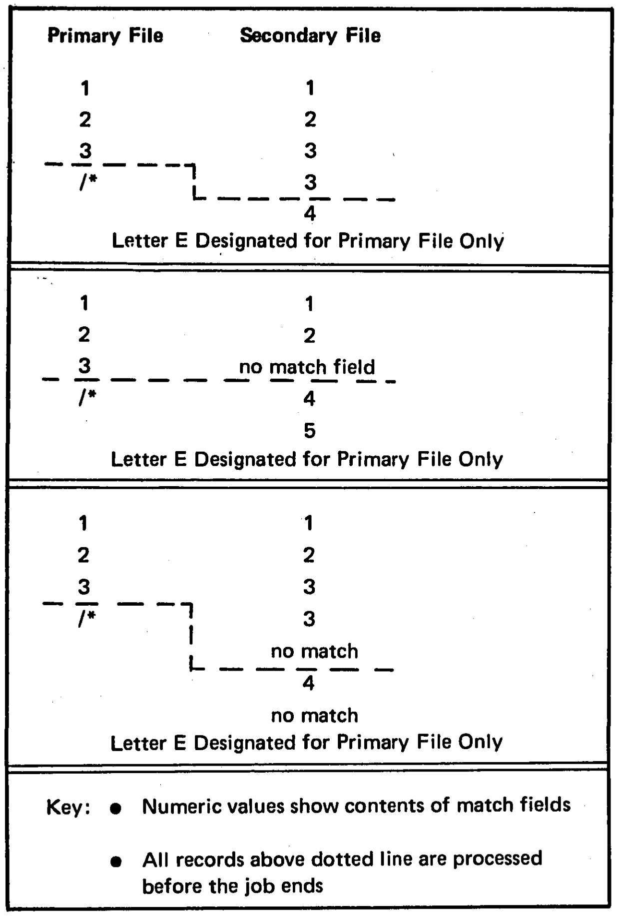


Figure 13. End-of-File Processing

COLUMN 18 (SEQUENCE)

Entry	*Explanation*
A	Sequence checking is to be done. Records in the file are in ascending order.
D	Sequence checking is to be done. Records in the file are in descending order.
Blank	No sequence checking is to be done.

Column 18 applies to update files, combined files, and all input files except table, array, chained, demand, and record address files. Leave column 18 blank for output, display, record address, table or array files, and chained files. Use it to indicate whether or not the program is to check the sequence of the records. Use columns 61-62 on the Input Sheet to identify the matching fields containing the sequence information.

Sequence checking is required when match fields are used in the records from the file. When a record from a matching input file is out of sequence, the program halts, and the operator has three options:

1. Bypass the record out of sequence and read the next record from the same file.

2. Bypass the record out of sequence, turn on the LR indicator, and perform all end-of-job and final total procedures.

3. Cancel the entire program.

COLUMN 19 (FILE FORMAT)

Entry	*Explanation*
F (or blank)	Fixed length records
V	Variable length records (EBCDIC tape files only)
D	Variable length records (ASCII tape files only)

Column 19 may contain an F, V or D entry. An F entry indicates all records in the file are the same length. A V entry indicates records in a tape file are variable length EBCDIC. A D entry indicates records in a tape file are variable length ASCII. A blank defaults to fixed length records (F).

COLUMNS 20-23 (BLOCK LENGTH)

Entry	*Explanation*
1-9999	1. Multiple of record length or record length for disk BSCA or SPECIAL files. 2. Record length for MFCU, MFCM, DISKET, CONSOLE, PRINTER, PRINT84, READ42, CRT77, and READ01 files.
18-9999	Tape file block length.
Blank	Block length for this file is the same as record length.

Columns 20-23 have a different use depending on the device named for the file. If an entry is specified, the entry must end in column 23, and leading zeros can be omitted (Figure 16).

Block Length for Disk Records

Disk block length must be a number equal to record length or a multiple of record length. The maximum block length is 9999.

Block length does not affect the way records are written on disk. Its function is to specify the amount of main storage to use for input/output area.

If a value equal to the disk record length is entered in these columns, RPG II will assign an efficient block length. See Table D-7 in Appendix D for block lengths computed by RPG II for various disk files and record lengths.

Block Length for Tape Records

The block length for fixed length tape records must be a multiple of the record length plus the length of the buffer offset and the total length must be from 18 to 9999 characters. When figuring the block length, remember to allow space for:

1. The number of records to be in a block
2. The length of the buffer offset (block prefix). This applies to ASCII files only.

The block length for variable length tape records need not be an exact multiple of the record lengths, but the following factors should be considered when calculating the approximate block length:

1. Constants of four bytes per block and four bytes per record should be added.
2. The minimum block size would be the maximum record length plus eight.

For a discussion of buffer offset, see *Columns 54-59 (Continuation Line Option).*

Device (Columns 40-46)	Block Length (Columns 20-23)	Record Length (Columns 24-27)	Maximum Record Length	System Supported
Blank (Device independent files)	Record length or multiple of record length	Record length	9999	Model 15
DISK and DISK45 (Models 10, 12, and 15), or DISK40 (Models 12 and 15 only)	Record length or a multiple of record length.	Record length	9999	Models 10, 12, and 15
MFCU1 or MFCU2	Record length	Record length	96	Models 10, 12, and 15
CONSOLE (printer-keyboard)	Record length	Record length	125	Models 10 and 12
PRINTER	Record length	Record length	96, 120, or 132 (number of print positions)	Models 10, 12, and 15
PRINTR2	Record length	Record length	96, 120, or 132 (number of print positions)	Models 10 and 12
PRINT84	Record length	Record length	132	Model 15
TAPE	Record length or a multiple of record length plus the buffer offset.	Record length	9999	Models 10, 12, and 15
READ42	Record length	Record length	80	Models 10, 12, and 15
CRT77 (CRT/keyboard)	Record length	Record length	120 – Input or Output files 35 – Display files 279 – Update files	Model 15
READ01 (2501 Card Reader)	Record length	Record length	80	Model 15
MFCM1 and MFCM2	Record length	Record length	80	Model 15
BSCA	Record length or multiple of record length	Record length	9999	Models 10, 12, and 15
SPECIAL	Record length or multiple of record length	Record length	9999	Models 10, 12, and 15
DISKET (3741 Data Station or Programmable Work Station)	Record length	Record length	128	Models 10, 12, and 15

Figure 14. Block Length and Record Length Entries

COLUMNS 24-27 (RECORD LENGTH)

Entry	*Explanation*
1-9999	The number of characters in each record (limited by the device used).
18-9999	Record length for tape files.

Columns 24-27 are used to indicate the length of records in all files except variable length tape files. All of the records in one file must be the same length. (For update files, the length of a record after it is updated must be the same as before it was updated.) The maximum record length allowed and the size of the I/O area assigned depends upon the device assigned to the file (Figure 14). For printer and card devices, an I/O area equal to the maximum record length is assigned. The record length specified, however, may be shorter than the maximum length for the device.

If no entry is placed in columns 24-27, the program defaults to the maximum record length for the device. For disk files, the default is 256; for SPECIAL, the default is 9999.

The entry you place in these columns must end in column 27. Leading zeros can be omitted.

The record length for tape must specify the size of the data records to be processed by this program. The record length for printer and card devices will be the maximum for the device (Model 15 only).

If variable length records are being used with a tape file, the record length must be the length of the largest record.

COLUMN 28 (MODE OF PROCESSING)

Entry	*Explanation*
L	Sequential within limits.
R	1. Random by relative record number.
	2. Random by key.
	3. By ADDROUT file.
	4. Direct file load (random load).
Blank	1. Sequential by key.
	2. Consecutive.

Use column 28 to indicate the method by which records are to be read from the file or to indicate that a direct file load (random load) is to take place.

For disk files specified as primary, secondary, or chained files, the possible methods depend upon the organizations of the files (Figure 15). For the other types of files, consecutive processing is the only possible method.

Column 31 is used to further identify the method for the program. See *Column 31 (Record Address Type)* in this chapter.

PRIMARY, SECONDARY, OR DEMAND FILES	
Organization	Possible Methods
Sequential	1. Consecutively 2. By ADDROUT file
Direct	1. Consecutively 2. By ADDROUT file
Indexed	1. Consecutively 2. By ADDROUT file 3. Sequentially by key 4. Sequentially within limits

CHAINED FILES	
Organization	Possible Methods
Sequential	Randomly by relative record number
Direct	Randomly by relative record number
Indexed	Randomly by key or by relative record number

Figure 15. Possible Record Retrieval Methods for Disk Files

Consecutive

The consecutive method applies to all sequential and direct files. It may also be used with indexed input files. During consecutive processing, records are read in the order in which they physically appear in the file. The contents of spaces left for missing records in direct files are read as though the records were there. (When a direct file is loaded, such spaces are filled with blanks.) You should allow for these blank records in your program.

The program reads records from the file until either the end of that file is reached or the program ends due to the end-of-file condition of another file. See *Column 17 (End of File)* in this chapter for more information about the second condition.

By ADDROUT File

An ADDROUT (address output) file is a record address file produced by the Disk Sort Program. It is a file of 3-byte disk records or 18-byte tape records containing binary relative record numbers of records in a disk file. Each tape record contains six binary relative record numbers. The binary relative record number is converted to a disk address and the record at that address in the original disk file is located and read. Records are read in this manner until either the end of the ADDROUT file is reached or the program ends due to the end-of-file condition of another file (see *Examples, Example 1*). See *Column 17 (End of File)* in this chapter for more information about the second condition.

Sequential By Key

The sequential by key method of processing applies to indexed files that are used as primary, secondary, or demand files.

Records are read in ascending key sequence (the order in which the record keys are arranged in the index portion of the file). The program reads records until all records in the file are processed or the program ends due to the end of file condition of another file. See *Column 17 (End of File)* for more information about the second condition.

Sequential Within Limits

The sequential within limits method of processing can be accomplished by using either: (1) a record address file containing limit records, or (2) the SETLL operation code during calculations.

The first method applies only to indexed disk files used as primary and secondary files and demand files. A limits record consists of the lowest record key and the highest record key of the records in the indexed disk file which are to be read. Limits records are contained in a record address file. The record address file can be located on disk, diskette, tape, punched on cards, or entered via the keyboard.

The second method applies only to indexed disk files used as demand files and sets the lower limit only. The program defaults to the address of the last record in the file for the upper limit. The lower limit may be reset before end of file is reached.

To process sequentially within limits, the program reads:

1. A limits record from the record address file, or the SETLL operation is used during calculations.

2. Records with keys greater than or equal to the low record key and less than or equal to the high record key (end of file when using SETLL).

The program repeats these two steps until either the end of the record address file is reached or the program ends due to the end-of-file condition of another file. See *Column 17 (End of File)* in this section for more information about the second condition.

The format of the records in a record address file containing limits must conform to these rules:

1. Only one set of limits is allowed per record in the record address file.

2. The length of a record in a record address file must be at least twice the length of the record key.

3. The low record key must begin in position 1 of the record. The high record key must follow the low record key. A record key can be from 1-29 characters in length.

4. Both the low record key and the high record key must be equal in length to the key field length specified in columns 29-30. Therefore, leading zeros may be necessary in specifying numeric record keys.

5. An alphameric record key may contain blanks.

Files containing limits and files being processed by limits can have keys in different formats. (For example, one file can have packed keys and the other unpacked.) During execution time the format of the key from the file containing limits will be changed to the format of the file being processed by limits. The format of the keys on each file must be indicated by an A or P in column 31. Also, the unpacked key length must be twice the packed length, minus either 1 or 2. See *Packed Decimal Format (P)* for more information concerning this calculation.

Note: A key may not contain any X'FF'.

The same set of limits can appear in more than one record address record. Data records, therefore, can be processed as many times as you wish.

The two record keys in a limits record can be equal. In this case, however, only one data record will be read.

The SETLL operation code method of limits processing applies to any indexed disk file designated as a demand file (D in column 16 and L in column 28 of File Description Sheet). You cannot, however, process an indexed demand file with SETLL if you are using a record address file to set the limits of the file.

The maximum number of files which may be processed using SETLL is limited by the number of demand files permitted (a maximum of 15 demand and/or chained files are allowed per program). See Example 2, Figure 17 for an example of SETLL. For additional information on how to set limits using the SETLL operation code, see *Operation Codes* in Chapter 8.

Random

The two methods, random by relative record number and random by key, apply to chained files only. They require the use of the CHAIN operation code. The records of a file to be read or written must be processed by the CHAIN operation code. The records are read or written only when the CHAIN statements that identify them are executed.

For sequential and direct files, relative record numbers must be used to identify the records (see *Examples, Example 3*). Relative record numbers identify the positions of the records relative to the beginning of the file. For example, the relative record numbers of the first, fifth, and seventh records in a file are 1, 5, and 7 respectively. (See *Operation Codes, CHAIN* in Chapter 8 for a description and example of direct file loading.)

For indexed files, record keys must be used to identify the records (see *Examples, Example 4*). A record key is the information from the key field of a record. The information is used in the index portion of the file to identify the record. Indexed files may also be processed randomly by relative record number if they are input files.

Records are read during the calculation phase of the program. Therefore, fields from these records can be used during detail or total calculations. Note then, that fields of records read from chained update files can be read and altered during total calculations and the records can be updated (written back on the file with alterations) during total output; the same also applies to detail calculations and detail output (see *Examples, Example 5*).

Examples

Example 1

Figure 16 shows processing a sequential disk file by an ADDROUT file. The record address file, ADRTFILE, defined as an ADDROUT disk file, consists of 3-byte binary relative record numbers which correspond to locations of records on the input disk file, MASTER. As each record is read from ADRTFILE, the indicated record from MASTER is located and read. For each record read from MASTER (indicator 01 is on), a detail line is printed on the printer output file, PRINTER.

Since end of file (*E* in column 17 of the File Description Sheet) is specified for the ADDROUT file, processing continues until all records in ADRTFILE have been read.

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

IBM International Business Machines Corporation — GX21-9092 UM/050* Printed in U.S.A.

Program — Programmer — Date — Punching Instruction: Graphic / Punch — Card Electro Number — Page 01 of — Program Identification (75–80)

Control Card Specifications

Line (3–5)	Form Type (6)
01	H

File Description Specification

Line	Form Type	Filename (7–14)	File Type (15)	File Designation (16)	End of File (17)	File Format (19)	Block Length (20–23)	Record Length (24–27)	L/R (28)	Length of Key Field or of Record Address Field (29–30)	Record Address Type (31)	Type of File Organization (32)	Extension Code (39)	Device (40–46)	File Addition/Unordered: Number of Extents (68–69)
02	F	ADRTFILE	I	R	E	F	3	3		3	I	T	E	DISK	01
03	F	MASTER	I	P		F	256	64	R		I			DISK	01
04	F	PRINTER	O			F	96	96						PRINTER	
05	F														
06	F														

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

IBM International Business Machines Corporation — Form X21-9091 Printed in U.S.A.

Program — Programmer — Date — Punching Instruction: Graphic / Punch — Card Electro Number — Page 02 of — Program Identification (75–80)

Extension Specifications

Line	Form Type	From Filename (11–18)	To Filename (19–26)
01	E	ADRTFILE	MASTER
02	E		

Figure 16 (Part 1 of 2). Processing a Sequential Disk File with an ADDROUT File

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 03 of | Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes	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
01	I	MASTER	NS			01											
02	I							1	8		ACCT						
03	I							9	64		NAMADR						

Lines 04–15 are blank except for the preprinted form type I.

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 04 of | Program Identification

Line	Form Type	Filename	Type (H/D/T/E)	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name	Edit Codes	B/A/C/1-9/R	End Position in Output Record	P/B/L/R	Constant or Edit Word
01	O	PRINTER	D					01						
02	O								ACCT			8		
03	O								NAMADR			70		

Lines 04–17 are blank except for the preprinted form type O.

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Figure 16 (Part 2 of 2). Processing a Sequential Disk File with an ADDROUT File

Example 2

In Figure 17, the input disk file, MASTER, described as an indexed file to be processed by record keys is to be processed within the limits contained on the record address file, LIMITS. The LIMITS file, which is further described on the Extension Sheet, is to be read from the primary MFCM hopper.

Each set of limits read from LIMITS will consist of the low and high account numbers to be processed. Since the account number key field (ACCT) is eight positions long, each set of limits will include two 8-position keys. These consist of two 5-position packed fields.

As MASTER is processed within each set of limits, the corresponding records are written out on the printer output file, PRINTER. Processing is complete when all sets of limits have been processed.

File SMASTER is processed by the SETLL operation code. It is characterized by having no extension specifications, and its filename appears in factor 2 of the SETLL operation code. In this example the first record read from file SMASTER would be the one whose key is equal to or the next higher than the literal 'AAAAAAAA'. Records are read sequentially to end of file unless the cycle is interrupted by additional SETLL operations.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 UM/050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 01 of | Program Identification (75–80)

Control Card Specifications

Refer to the specific System Reference Library manual for actual entries.

Line (3–5)	Form Type (6)	7–74
01	H	

File Description Specification

Line (3–5)	Form Type (6)	Filename (7–14)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D (16)	End of File E (17)	Sequence A/D (18)	File Format F/V/S/M/D (19)	Block Length (20–23)	Record Length (24–27)	Mode of Processing L/R (28)	Length of Key Field or of Record Address Field (29–30)	Record Address Type A/P/I/K (31)	Type of File Organization or Additional Area I/D/T or 2 (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/E/M (53)	Name of Label Exit (54–59)	Extent Exit for DAM / Core Index (60–65)	A/U (66)	File Addition/Unordered (67)	Number of Tracks for Cylinder Overflow / Number of Extents (68–69)	Tape Rewind R/U/N (70)	File Condition U1-U8 (71–72)	73–74
02	F	LIMITS	I	R			F	80	80		5	P				E	MFCM1										
03	F	MASTER	I	P			F	256	64	L	8	A	I		1		DISK							01			
04	F	PRINT	O				F	96	96					OF			PRINTER										
05	F	SMASTER	I	D			F	256	64	L	8	A	I		1		DISK										

IBM International Business Machine Corporation

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

Form X21-9091 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 02 of | Program Identification (75–80)

Extension Specifications

Line (3–5)	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 or Array Name (27–32)	Number of Entries Per Record (33–35)	Number of Entries Per Table or Array (36–39)	Length of Entry (40–42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46–51)	Length of Entry (52–54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58–74)
01	E			LIMITS	MASTER													
02	E																	

Figure 17 (Part 1 of 2). Processing an Indexed File Sequentially Within Limits

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 03 of | Program Identification

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Field Location From	Field Location To	Field Name
01	I	MASTER	NS	01			
02	I				1	8	ACCT
03	I				9	64	NAMADR
04	I	SMASTER	NS	02			
05	I				1	8	ACTT
06	I				9	64	NAME
07	I						
08	I						
09	I						

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 03 of | Program Identification

Line	Form Type	Factor 1	Operation	Factor 2
01	C			
02	C	ACCT	SETLL	SMASTER
03	C		READ	SMASTER
04	C			
05	C			

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 04 of | Program Identification

Line	Form Type	Filename	Type (H/D/T/E)	Space After	Output Indicators	Field Name	End Position in Output Record	Constant or Edit Word
01	O	PRINT	D	1	01			
02	O					ACCT	8	
03	O					NAMADR	70	
04	O		D	1	02			
05	O						16	'SMASTER NO. = '
06	O					ACTT	25	
07	O					NAME	90	
08	O							

Figure 17 (Part 2 of 2). Processing an Indexed File Sequentially Within Limits

Example 3

In Figure 18, the direct update file, MASTER, is to be processed randomly by relative record numbers. The account number (ACCT) from the primary MFCU file, CHANGE, is used as the relative record number.

As each record is read from CHANGE, the MASTER record corresponding to the account number is read during calculation time by the CHAIN operation code. At detail output time, the data in the NEW field replaces the original data in the NAMADR field and the updated MASTER record is output to its original relative record location on the disk file.

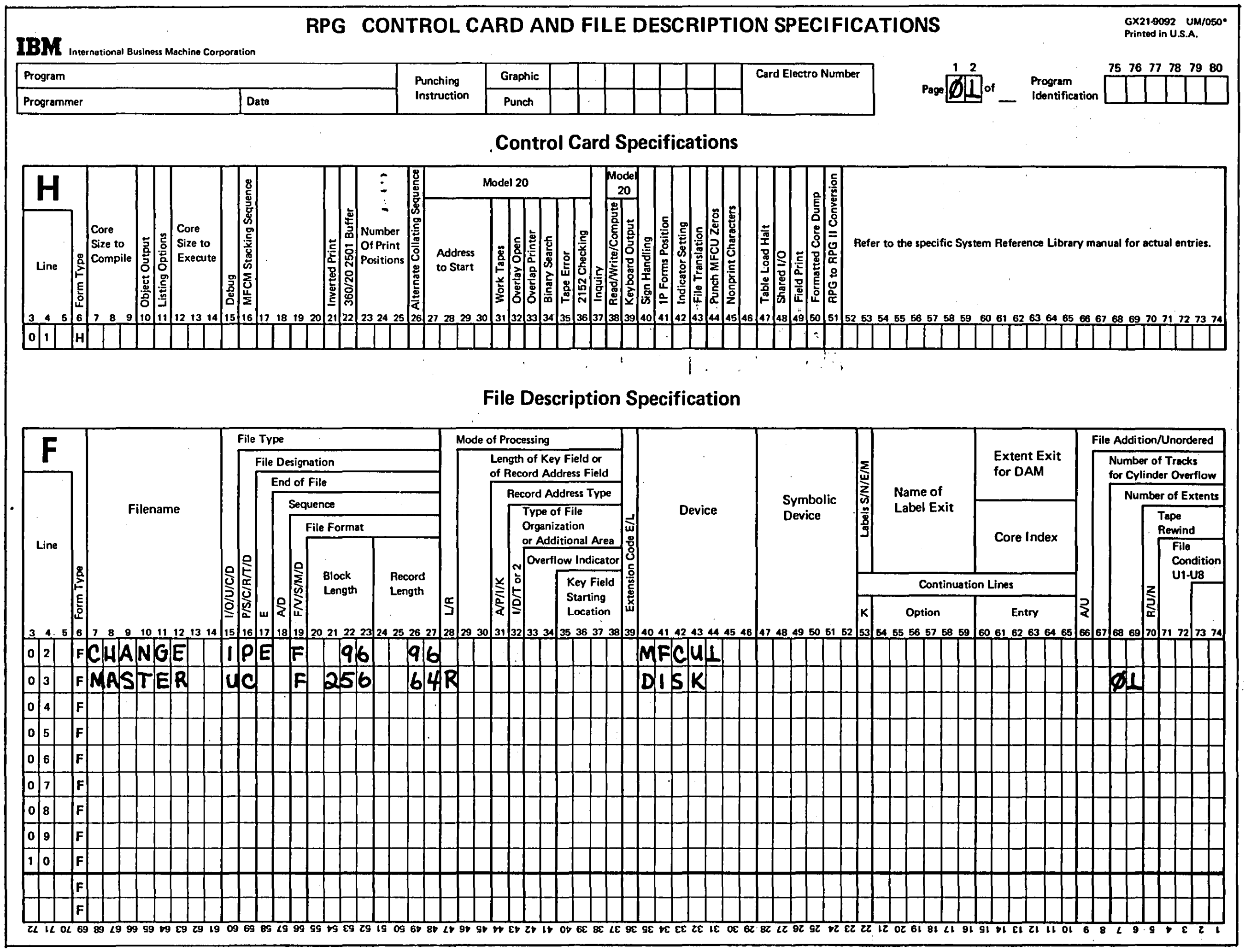


Figure 18 (Part 1 of 2). Random Processing of a Direct File by Relative Record Number

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 U/M 050* Printed in U.S.A.

Program: | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page 1 2: Ø2 of __ | Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Field Location From	Field Location To	Field Name
01	I	MASTER	NS	Ø1			
02	I				1	8Ø	ACCT
03	I				9	64	NAMADR
04	I	CHANGE	NS	Ø2			
05	I				1	8Ø	ACCT
06	I				9	64	NEW
07	I						
08	I						
09	I						

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093 Printed in U.S.A.

Program: | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page 1 2: Ø3 of __ | Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators	Comments
01	C		Ø2	ACCT	CHAIN	MASTER						
02	C											
03	C											

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050* Printed in U.S.A.

Program: | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page 1 2: Ø4 of __ | Program Identification 75 76 77 78 79 80

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators	Field Name	End Position in Output Record	Constant or Edit Word
01	O	MASTER	D	Ø1			
02	O				NEW	64	
03	O						

Figure 18 (Part 2 of 2). Random Processing of a Direct File by Relative Record Number

Example 4

Figure 19 shows random processing by key of an indexed file. MASTER, a chained update file, is described on the File Description Sheet as an indexed file to be processed by keys. As each record is read from the input card file, CHANGE, the account number (ACCT) is used as the key to chain to the corresponding record in MASTER at calculation time. At detail output time, the data in the NEW field of CHANGE replaces the original data in the NAMADR field. The updated MASTER record is then written on its original disk location. See *Column 32* in this chapter for a description of indexed file organization.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 UM/050* Printed in U.S.A.

Program		Punching Instruction	Graphic	Card Electro Number	Page 01 of	Program Identification
Programmer	Date		Punch			

Control Card Specifications

Line	Form Type	Entries (7–74)
01	H	

Refer to the specific System Reference Library manual for actual entries.

File Description Specification

Line	Form Type	Filename (7–14)	File Type (15)	File Designation (16)	End of File (17)	Sequence (18)	File Format (19)	Block Length (20–23)	Record Length (24–27)	Mode of Processing (28)	Length of Key Field or of Record Address Field (29–30)	Record Address Type (31)	Type of File Organization or Additional Area (32)	Overflow Indicator (33–34)	Key Field Starting Location (35–38)	Extension Code (39)	Device (40–46)	Symbolic Device (47–52)	Labels (53)	Name of Label Exit (54–59)	Core Index (60–65)	A/U (66)	Number of Extents (68–69)
02	F	CHANGE	I	P	E		F	96	96								MFCU1						
03	F	MASTER	U	C			F	256	64	R	8	A	I		1		DISK						01
04	F																						
05	F																						
06	F																						
07	F																						
08	F																						
09	F																						
10	F																						
	F																						
	F																						

Figure 19 (Part 1 of 2). Random Processing of an Indexed File by Key

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page Ø2 of | Program Identification

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Field Location From	Field Location To	Field Name
01	I	MASTER	NS	Ø1			
02	I				1	8	ACCT
03	I				9	64	NAMADR
04	I	CHANGE	NS	Ø2			
05	I				1	8	ACCT
06	I				9	64	NEW
07	I						
08	I						
09	I						

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page Ø3 of | Program Identification

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field
01	C	Ø2	ACCT	CHAIN	MASTER	
02	C					
03	C					
04	C					

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page Ø4 of | Program Identification

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators	Field Name	End Position in Output Record
01	O	MASTER	D	Ø1		
02	O				NEW	64
03	O					
04	O					

Figure 19 (Part 2 of 2). Random Processing of an Indexed File by Key

Example 5

Figure 20 shows the updating of an indexed file. The indexed file, MASTER, is described as a chained update file to be processed by keys. The key field in MASTER is ITEMNO, in positions 1-10. The index will be sorted into ascending sequence when processing is complete.

As each record is read from TRANS, the input transaction file, the ITEMNO field is used as the key to chain to MASTER during calculations. If the character 2 is in position 64 of the transaction record, the quantity in ADJUST is added to the ONHAND field of MASTER. If the character 3 is in position 64, ADJUST is subtracted from ONHAND. If the character 1 appears in location 64 of the MASTER record, the updated ONHAND field is written out on its original location in the MASTER record at detail output time.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 UM/050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 01 of __ | Program Identification (75-80)

Control Card Specifications

Line (3-5)	Form Type (6)	Columns 7-74
01	H	

Refer to the specific System Reference Library manual for actual entries.

File Description Specification

Line	Form Type	Filename (7-14)	File Type (15)	File Designation (16)	End of File (17)	Sequence (18)	File Format (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing (28)	Length of Key Field or of Record Address Field (29-30)	Record Address Type (31)	Type of File Organization or Additional Area (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code (39)	Device (40-46)	Symbolic Device (47-52)	Number of Extents (68-69)
02	F	MASTER	U	C			F	256	64	R	1Ø	A	I		1		DISK		Ø1
03	F	TRANS	I	P			F	96	96								MFCU1		
04	F																		
05	F																		
06	F																		
07	F																		
08	F																		
09	F																		
10	F																		

Figure 20 (Part 1 of 2). Updating an Indexed File

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 02 of | Program Identification

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Position (1)	C/Z/D	Character	Field Location From	To	Decimal Positions	Field Name
01	I	MASTER	NS	01	64	C	1				
02	I							1	10		ITEMNO
03	I							11	50		DESCR
04	I							51	56		PRICE
05	I							57	63	0	ONHAND
06	I	TRANS	NS	02	64	C	2				
07	I	OR		03	64	C	3				
08	I							1	10		ITEMNO
09	I							11	15	0	ADJUST
10	I										
11	I										

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 03 of | Program Identification

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length
01	C		ITEMNO	CHAIN	MASTER		
02	C	02	ONHAND	ADD	ADJUST	ONHAND	
03	C	03	ONHAND	SUB	ADJUST	ONHAND	
04	C						
05	C						

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 04 of | Program Identification

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators	Field Name	End Position in Output Record
01	O	MASTER	D	01		
02	O				ONHAND	63
03	O					
04	O					
05	O					

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign; Y = Date Field Edit; Z = Zero Suppress

Figure 20 (Part 2 of 2). Updating an Indexed File

COLUMNS 29-30 (LENGTH OF KEY FIELD OR RECORD ADDRESS FIELD)

Entry	Explanation
Number	Length of record key or ADDROUT file record

Columns 29-30 apply only to indexed disk files and record address files. Enter:

1. The length of the record keys in indexed files and record address files that contain limits.

2. The length of the records in ADDROUT files.

3. The length of record keys in packed format.

All of the key fields in the records in an indexed file must be the same length. The maximum is 29 bytes; 8 bytes are for record keys in packed format. All of the records in an ADDROUT file have a length of three. A leading zero is not required for entries of 1-9.

COLUMN 31 (RECORD ADDRESS TYPE)

Entry	Explanation
A	Record keys in unpacked format are used in processing and loading indexed files.
I	The file is being processed by means of an ADDROUT file or the file is an ADDROUT file.
P	Record keys in packed format are used in processing and loading indexed files.
Blank	1. Relative record numbers are used in processing sequential and direct files. 2. A sequential or direct file is being loaded. 3. Records are read consecutively. 4. The file is not a record address file.

PRIMARY AND SECONDARY FILES		
Method	Column 28 (Mode of Processing)	Column 31 (Record Address Type)
Consecutive	Blank	Blank
By ADDROUT	R	I
Record address	Blank	A or P
Sequential By Key	Blank	A or P
Sequential Within Limits	L	A or P

CHAINED FILES		
Method	Column 28 (Mode of Processing)	Column 31 (Record Address Type)
Random By Relative Record Number	R	Blank
Random By Key	R	A
Direct File Load (Random Load)	R	Blank*

* A direct file load requires an *O* in column 15 and a *C* in column 16.

Figure 21. Specifications Identifying Methods for Retrieving Records or Loading a Direct File

Column 31 applies to files specified as input, update, or chained output files. It indicates the way in which records in the file are identified (Figure 21). Together, columns 28 and 31 indicate:

1. The method by which records are read from the file.

2. A direct file load.

For ADDROUT files, column 31 must contain an *I*.

Note: When building a file with packed keys (P in column 31), you must specify the key field as packed in output specifications.

COLUMN 32 (FILE ORGANIZATION OR ADDITIONAL I/O AREA)

Entry	*Explanation*
I	Indexed file.
T	ADDROUT file.
1-9	Sequential file or direct file. Use two input/output areas for the file. (The digit two is preferred because a maximum of two input/output areas are allowed.)
Blank	Sequential file or direct file. Use one input/output area for the file.

Use column 32 to:

- Identify the organization of all files except ADDROUT files.
- Identify ADDROUT files.
- Indicate whether one or two input/output areas are to be used for sequential files or direct files.

File Organization

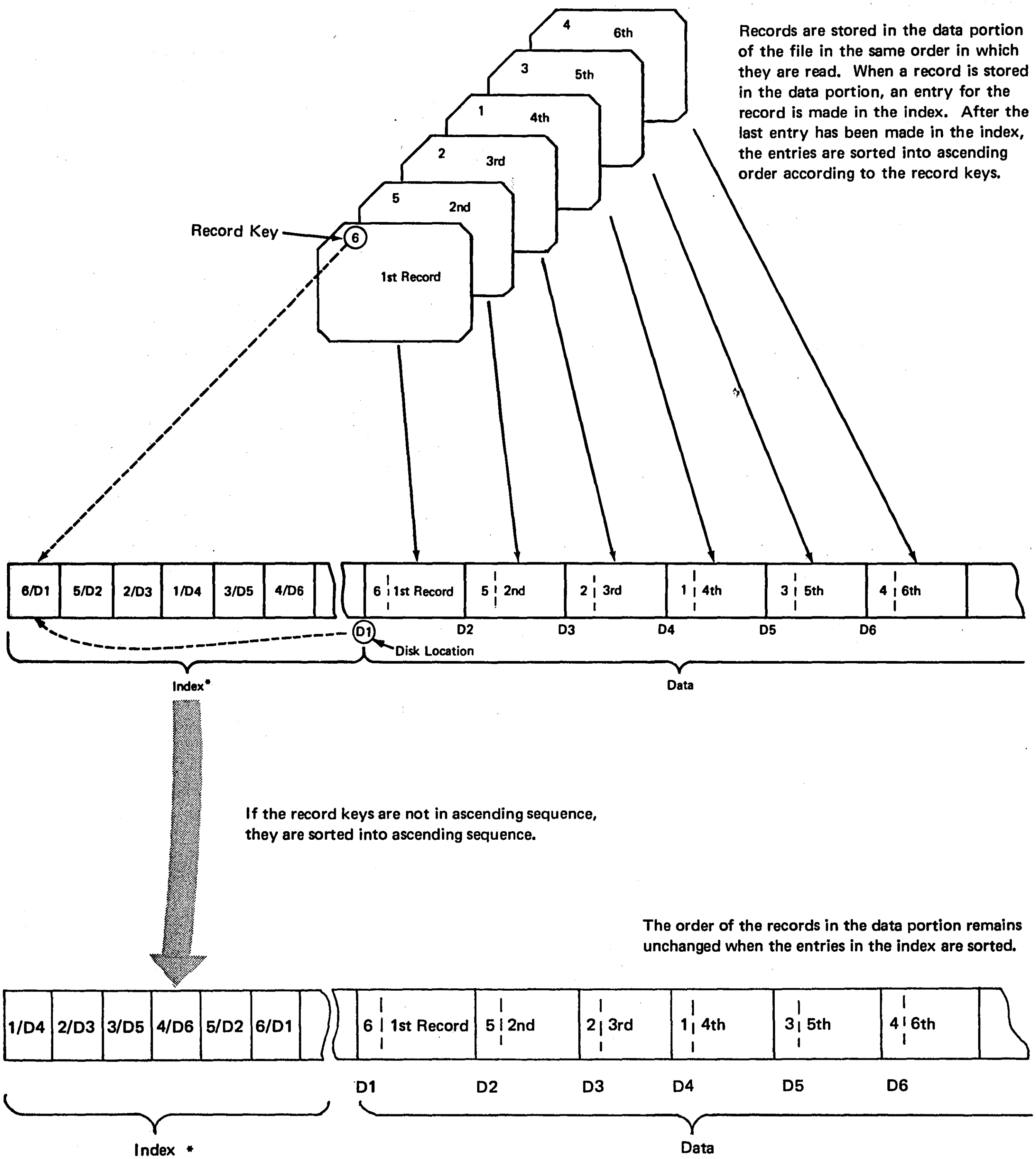
File organization is the arrangement of records in a file. The three types are indexed, direct, and sequential. Files organized in these ways are called indexed files, direct files, and sequential files, respectively.

Indexed Files

An indexed file is a disk file in which the location of records is recorded in a separate portion of the file called an index. The index and its associated file occupy adjacent positions on disk. The index contains the record key and record location of every record (Figure 22).

A record key is the information from the key field of a record. The record key can be used to identify the records of an indexed file. Record keys are always required in an indexed file. Indexed files may be loaded with the keys in ascending sequence or keys in non-ascending sequence. After a file is loaded in non-ascending key sequence, the keys in the index are sorted into ascending sequence. See Column 66 of the File Description Sheet for a definition of the unordered load function.

Note: An indexed file in a simulation area on a 3340 or 3344 disk file cannot be processed as an indexed file. However, it can be processed as a sequential file. In this case, the index is not used.



*Entries are of the form record-key/disk-location (D1=1st disk location, D2=2nd disk location, and so on)

Figure 22. Indexed File Organization

Direct Files

Direct files are disk files in which records are assigned specific record positions. Regardless of the order in which the records are put in the file, they always occupy a specific position (a specific disk address). Relative record numbers identify the relative position of a record within the file.

Before a direct file is loaded the entire disk area (a minimum of one track is allocated) for the direct file is cleared to blanks. Spaces are reserved in a direct file for records not available at the time the file is loaded (Figure 23). You should handle these blank records in your program.

Sequential Files

Sequential files are files in which the order of the records is determined by the order in which the records are put in the file. For example, the tenth record put in the file occupies the tenth record position.

Files other than disk files are always sequential files.

Additional Input/Output Area

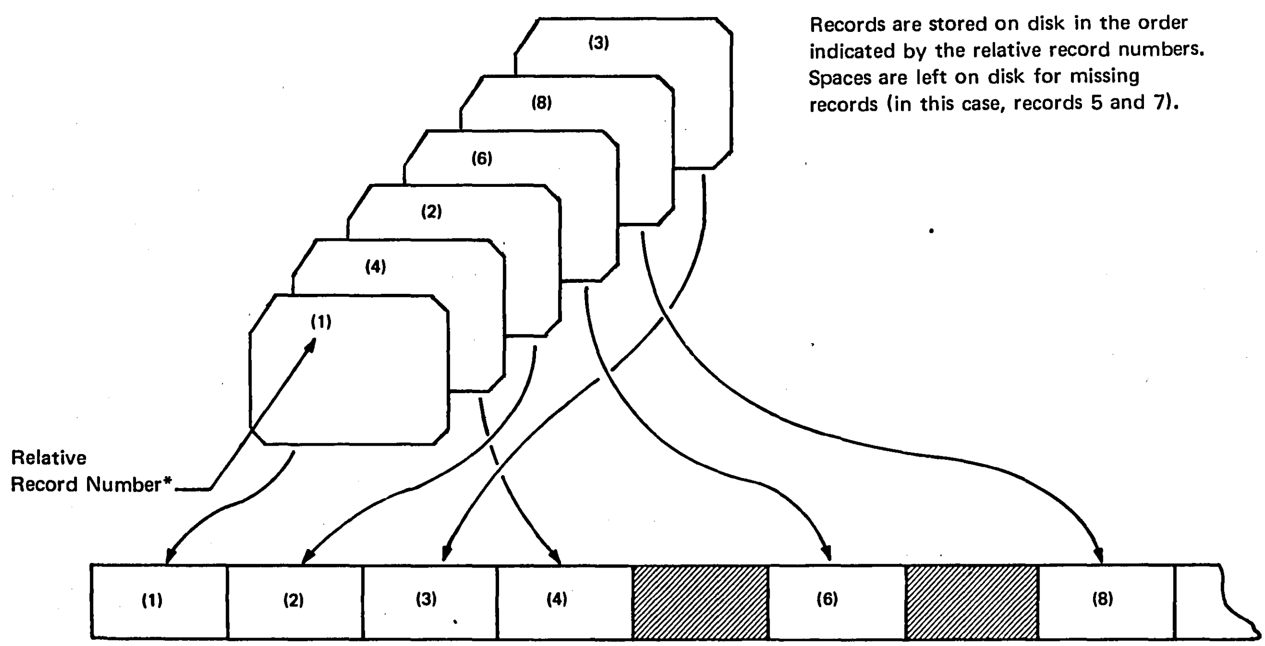
Normally the program uses one input/output area for each file. A second area, however, can be used for sequential and direct disk files and non-disk files, specified as input or output files in column 15. Additional input/output areas cannot be used for console files, table files, demand files, or record address files, or for disk files using a shared I/O area (Models 10 and 12 only). The devices associated with these files can be the disk, MFCU, DISKET, READ42, MFCM, and device-independent files for input or output files; READ01 for input files only; the printer for output files only. If you want two areas to be used for a card file, do not specify stacker selection for the records in the file. Stacker selection is described under *Column 42, Stacker Select* in Chapter 7.

The use of two I/O areas may increase the size of the program. Therefore, before you indicate that two areas are to be used for a file, be sure that the increase in size will not make your program exceed the capacity of your system.

Additional I/O area cannot be specified for disk files with a shared input/output area (column 48 of the Control Card Specifications Sheet). If both additional I/O and shared input/output areas are specified, additional I/O is dropped, and a warning message is given (Models 10 and 12). On the Model 15 using Program Numbers 5704-SC2 and 5704-RG2, an additional I/O area cannot be specified if file sharing is used.

ADDROUT Files

When describing an ADDROUT file, you must place a *T* in column 32. The ADDROUT file must be a disk or tape file. See *Column 28, Mode of Processing* for a description and example of ADDROUT processing.



* The programmer usually derives relative record numbers from information in the records.

Figure 23. Direct File Organization

COLUMNS 33-34 (OVERFLOW INDICATOR)

Entry	*Explanation*
OA-OG, OV	An overflow indicator is used to condition records in the file. The indicator specified is the one used.
Blank	No overflow indicator is used.

Columns 33-34 apply to output files assigned to the printer. Use these columns to indicate that you are using an overflow indicator to condition records being printed in the file. Any overflow indicators used in a program must be unique for each output file assigned to the printer. Note that only one overflow indicator can be assigned to a file. Do not assign overflow indicators to a console file.

Overflow Indicators

Overflow indicators are used only with printer files, primarily to condition the printing of heading lines. If you intend to use an overflow indicator to condition output lines on the printer, you must assign an overflow indicator to the printer file on the File Description Sheet (columns 33-34). The same indicator must be used to condition all lines that are to be written only when overflow occurs.

If the destination of a space/skip or print operation is a line beyond the overflow line, the overflow indicator is turned on and remains on until all overflow lines are printed. However, if a skip or space is specified that advances the form past the overflow line to the first line or past the first line on a new page, the overflow indicator does not turn on.

If an overflow indicator is used as a conditioning indicator, it indicates that output is to be performed at overflow time. This applies whether or not the line conditioned by the indicator is in an AND or OR relationship with other indicators. However, to get output, all indicator conditions specified in an AND relationship must be met.

The overflow indicator may be set by the SETON or SETOF operation code. After all total records have been written, however, the indicator is set as it normally is in accord with the overflow line.

USING OVERFLOW

When the printer has reached the end of a printed page, RPG II language allows you to do one of three things:

1. Advance to the top (line 6) of the next page and continue printing.
2. Ignore the fact that the end of the page has been reached and keep right on printing.
3. Print special lines at the bottom of the page and at the top of the new page.

You automatically get the first option by not assigning an overflow indicator. You get the second by assigning an overflow indicator and never using it to condition output lines. You get the third by assigning and using overflow indicators. These three possibilities are described as follows:

1. For every job you do you must determine how many lines will be printed on each page or form. You can indicate this by line counter specifications. From these specifications RPG II determines which line is the overflow line. (The overflow *area* includes the first line past the overflow line to the end of the form.) When the overflow line is sensed, an overflow indicator automatically turns on and the following steps occur:
   a. Detail lines are printed (if this part of the program cycle has not already been completed).
   b. Total lines are printed if required.
   c. Forms advance to a new page.
   d. The overflow indicator turns off.

2. If you are not concerned about pages or skipping to new pages and want one continuous listing, you must make an entry that will cause the automatic handling of overflow and advancing of forms to be discontinued. To cause overflow to be ignored, assign an overflow indicator to the printer file in columns 33-34 of a file description specification line and do not reference it on output specifications.

3. If you are concerned about pages and want certain lines to appear on each page, assign an overflow indicator to the printer file in columns 33-34 of the File Description Sheet (Figure 24). Use this same indicator to condition those lines which you want printed on every page. Usually these lines are total lines which must be printed at the bottom of every page, or heading lines which must be printed at the top of each new page.

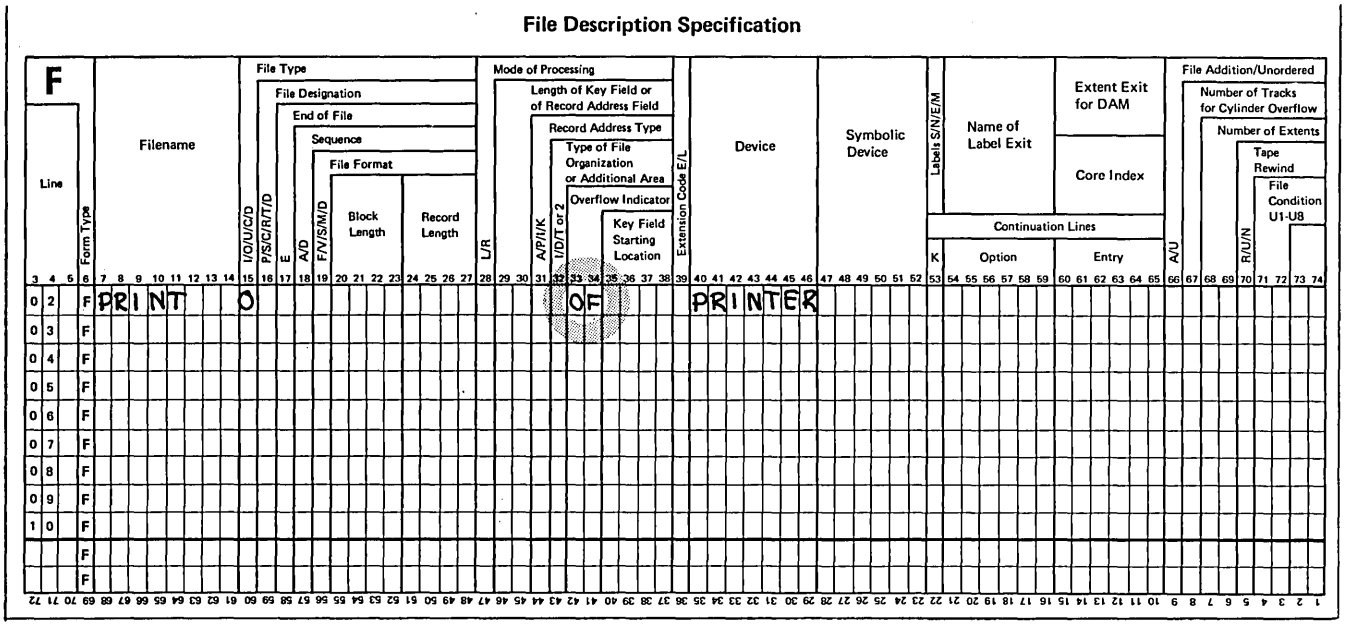


Figure 24. Assigning an Overflow Indicator

When an overflow indicator is assigned and used, forms do not automatically advance to a new page. You have to specify a skip to the first printing line on a new page. This skip is usually specified on the first heading line you want printed on the new page (Figure 25).

In the case where you have specified an overflow indicator and are using it to condition output lines, the following steps occur when the overflow line (end of page) has been sensed:

a. Detail lines are printed (if that part of the program cycle has not already been completed).
b. Total lines are printed (except at LR time).
c. Total overflow lines are printed if conditioned by the overflow indicator.
d. Forms advance to the next page if indicated by the skip specification on a heading line or total line.
e. Headings and detail lines are printed, if conditioned by overflow indicators.

Writing Specifications Using Overflow Indicators

Often you want each page to contain information from only one control group. (Information from one group may require several printed pages, however.) You might also wish each page to have headings identifying the type of information on the page. For these cases you need to use both the control level indicators *and* the overflow indicators. Together they condition when headings and/or group information are to be printed.

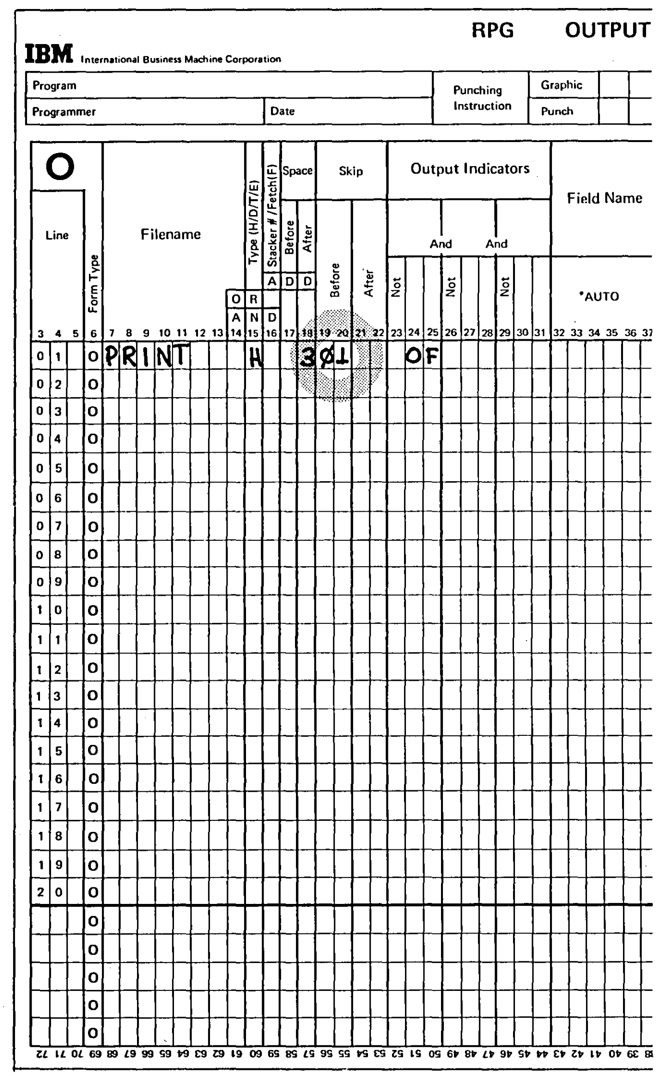


Figure 25. Advance Forms to New Page

A new page should advance either when the overflow line has been reached (the overflow indicator you assigned is on) or when there is a change in a control field (L indicator is on). You must specify that each indicator causes a new page to be advanced by specifying a skip to the first printing line on a page. If the control level has changed and the overflow condition has occurred at the same time, it is possible to duplicate an output line (one called for by the overflow indicator, the other by the control level indicator). A blank page can also appear in your report as a result.

Figure 26 shows the coding necessary for printing headings on every page: first page, every overflow page, and each new page to be started because of a change in control fields (L2 is on). Line 01 allows the headings to be printed at the top of a new page (skip to 01) only when an overflow occurs (OV is on and L2 is not on).

Line 02 allows printing of headings on the new page only at the beginning of a new control group (L2 is on). This way, duplicate headings caused by both L2 and OV being on at the same time do not occur. Line 02 allows headings to be printed on the first page after the first record is read. This is true because the first record always causes a control break (L2 turns on), if control fields are specified on the record. (If the first record did not have a control field, another OR line would be necessary with a 1P entry in columns 24-25.)

Figure 27 shows the necessary coding for the printing of certain fields on every page: a skip to 01 (first line on new page) is done either on an overflow condition or on a change in control level (L2). The NL2 indicator in line 01 prevents the line from printing and skipping twice in the same cycle.

Fetching The Overflow Routine

When the overflow line is reached, the same sequence of events always takes place. These were described previously. Briefly, remaining detail lines, total lines, and total overflow lines (lines conditioned by the overflow indicator) are printed on the page even after overflow has occurred. Therefore, you must leave enough room between the overflow line and the actual end of page to have room for all these lines to print.

However, you can run into problems when you do this. For example, if a different number of detail or total lines can be printed each time, you may not have allowed enough room between the overflow line and the end of page to take care of all total lines which will print before the forms advance. Therefore, printing is done on the perforation. You may also have to allow so much room between the overflow line and the end of page that often only half a page is used.

To take care of these problems, you may call for the printing of overflow lines and a forms advance any time after the overflow line has been reached. Causing overflow lines to be printed ahead of the usual time is known as *fetching overflow*. When overflow is caused in this way, the following events occur:

1. All total lines conditioned by the overflow indicator are printed.

2. Forms advance to new page when a skip to 01 has been specified in a line conditioned on an overflow indicator.

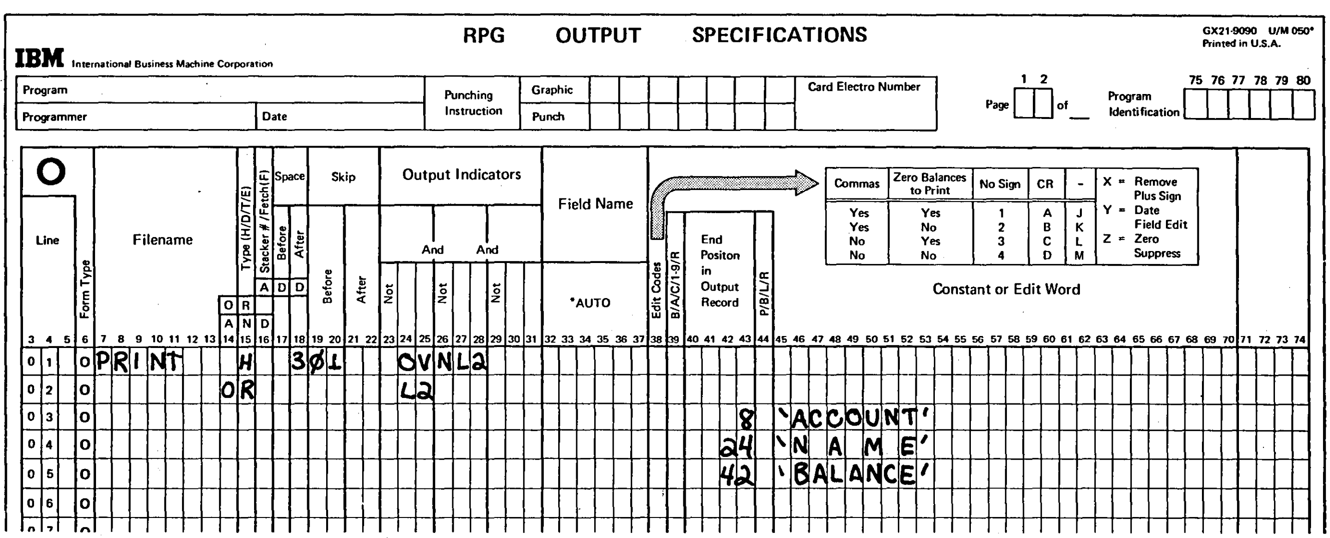


Figure 26. Printing Headings on Every Page

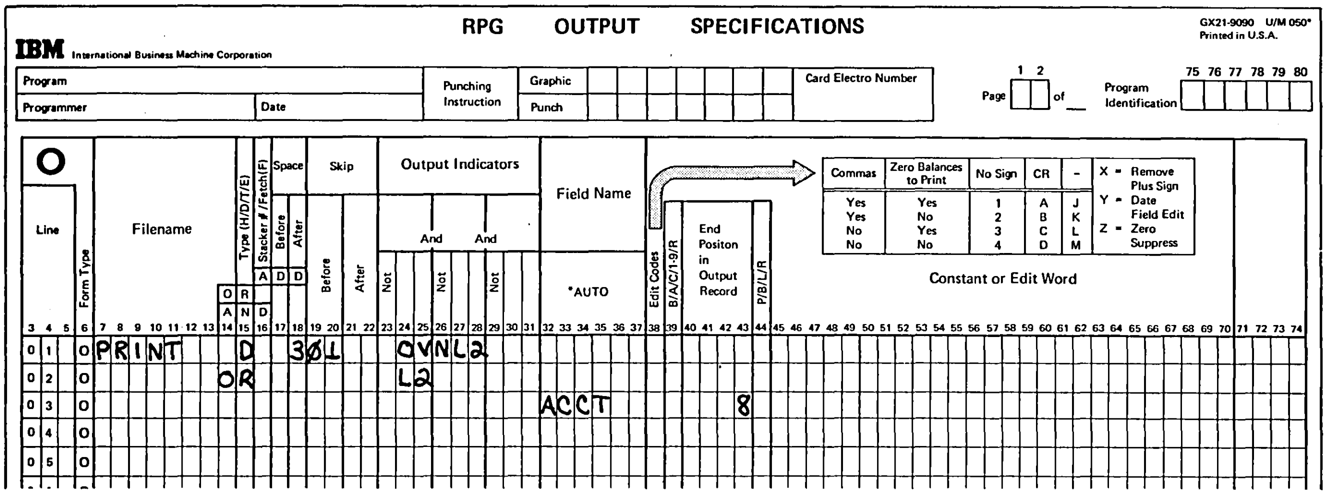


Figure 27. Printing Fields on Every Page

3. Heading and detail lines conditioned by the overflow indicator are printed.

4. The line that fetched overflow is printed.

5. Any detail and/or total lines left to be printed for that program cycle are printed.

For the printer file, an *F* in column 16 on the Output Sheet specifies that the overflow routine will be fetched. An *F* can be specified for any total, detail line, or exception line except those conditioned by an overflow indicator.

Figure 28 shows the use of a fetched overflow routine (*F* in column 16). The total lines 03, 09, and 11 can fetch the overflow routine. They do this, however, only if the overflow line has been sensed prior to the printing of one of these lines. If the overflow indicator is turned on before the output line specified in line 03 is printed and if control level indicator L1 is on, forms advance to the new page as specified by the skip entry in the heading line. The heading line and all total lines are printed on the new page. If, however, the printing of the line specified in 03 caused the overflow indicator to turn on, the following happens:

1. The line specified in 05 prints on the same page.

2. The line specified in 07 prints on the same page.

3. The line specified in 09 fetches an overflow (*F* in column 16) and causes the heading line and all total lines (09, 11, 13, and 15) to print on the new page.

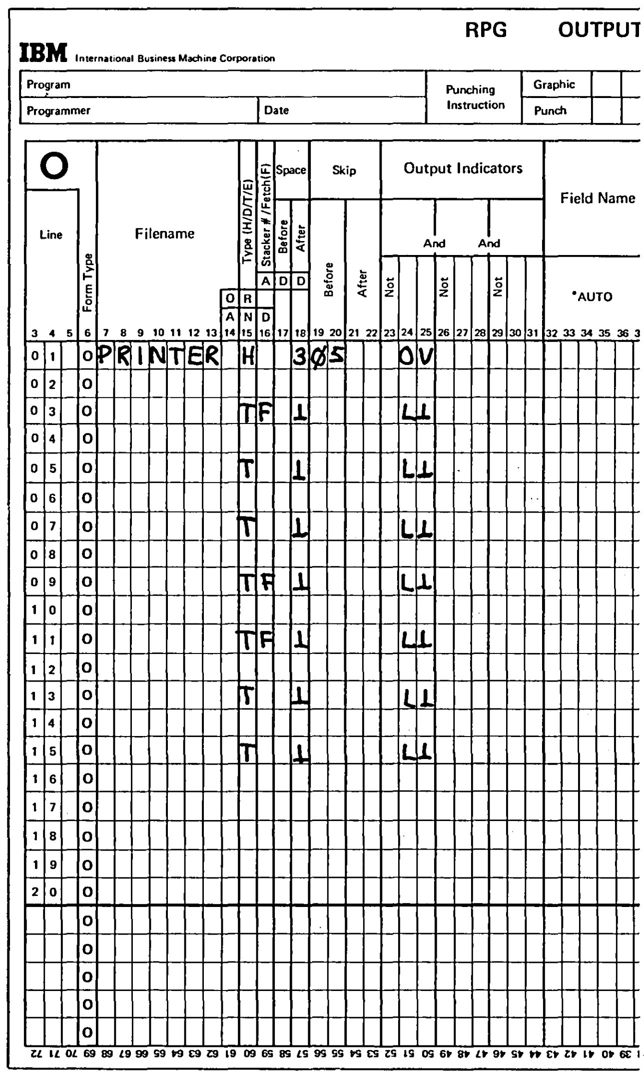


Figure 28. Uses of Fetch

If the output lines specified in 09 fetched overflow, line 11 does not fetch a new page again because an internal overflow indicator was turned off after line 09 fetched overflow. (The external overflow indicator is not turned off at this time.) Setting off the internal overflow indicator prevents overflow processing from occurring twice for a single physical overflow. (Remember, a line can fetch overflow only when the internal overflow indicator is on.) Line 11 fetches overflow only if the output line specified in 09 causes the overflow indicator to turn on.

You should fetch the overflow routine (*F* in column 16) only when you feel that (1) this line, when printed, could cause overflow and (2) if it did, there would not be enough room left on the page to print the remaining detail and/or total output lines plus lines conditioned by the overflow indicator.

When more than one printer file is used, fetch overflow applies only to the overflow lines associated with the file containing the record that specified fetch.

Note: Fetch overflow cannot be specified when an overflow indicator is specified in columns 23-31 on the same specification line. If this condition does occur, fetch overflow is not performed.

Overflow Printing with EXCPT Operation Code

Overflow indicators cannot condition an exception line, but can condition fields within an exception line. The use of the EXCPT operation code with exception lines (E in column 15 of the Output Sheet) causes only exception lines to be printed during calculation time. If the overflow line is sensed when an exception line is printed, the overflow indicator turns on as usual, but overflow processing does not occur until another exception line conditioned to print (with fetch overflow specified) is encountered.

The actual overflow output lines (totals and/or headings) must be coded as H, D, or T types. The use of fetch overflow will cause the H, D, or T overflow output lines to be printed if the overflow indicator is on. The overflow output lines are printed prior to the printing of the line on which fetch overflow is specified. The user may also force overflow by issuing a SETON of the appropriate overflow indicator prior to the EXCPT operation code, provided fetch overflow has been specified.

General Considerations

When using the overflow indicator to condition overflow printing, remember:

1. Overflow indicators may be turned on and off by the operation codes SETON and SETOF.
2. Spacing past the overflow line causes the overflow indicator to turn on.
3. Skipping past the overflow line to any line on the new page does not turn the overflow indicator on.
4. Skipping past the overflow line to a line on the same page causes the overflow indicator to turn on.
5. A skip to a new page specified on a line not conditioned by an overflow indicator causes the overflow indicator to turn off.

Figure 29 shows the setting of overflow indicators during the normal overflow routine and during a fetched overflow routine for both normal output and exception output. The left-hand portion of the graph shows when the indicators are on or off in relation to the general program cycle. For example, if, during normal output, a detail line is printed on the line number specified as the overflow line, the overflow indicator turns on. It remains on until the end of the next program cycle. The solid blank lines indicate that the indicator is on. The dashes are used to show a connection between the end of one cycle and the start of the next.

COLUMNS 35-38 (KEY FIELD STARTING LOCATION)

Entry	*Explanation*
1-9999	Record position in which the key field begins.

Columns 35-38 apply to indexed disk files only. An entry *must* be made in these columns for an indexed disk file. Use them to identify the record position in which the key field begins. The key field of a record is the field that contains the information that identifies the record. The information is used in the index portion of the file. The key field must be in the same location in all of the records in the file.

The number you place in these columns must end in column 38. Leading zeros can be omitted.

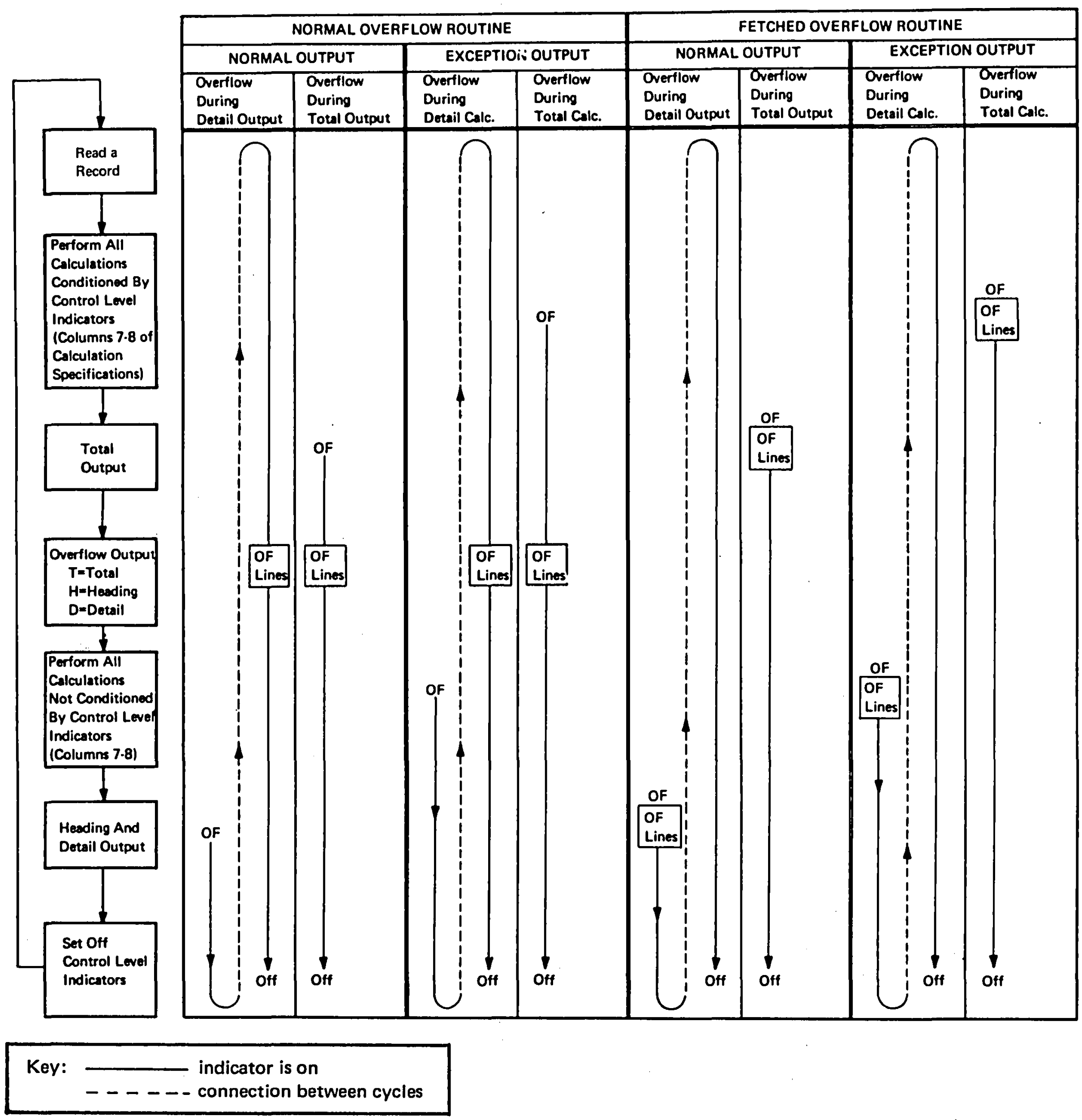


Figure 29. Overflow Printing: Setting of the Overflow Indicator

Key fields cannot contain any X'FF' characters. Therefore, if the key field is a binary field or is generated by the RPG Program you must be certain no X'FF' characters appear in the key field.

COLUMN 39 (EXTENSION CODE)

Entry	*Explanation*
E	Extension specifications further describe the file.
L	Line counter specifications further describe the file.

Column 39 applies to (1) table and array files that are to be read during program execution, (2) record address files, and (3) output files that are assigned to the printer. Output files that are assigned to the printer can be described on the Line Counter Sheet. Table, array, and record address files must be described on the Extension Sheet.

COLUMNS 40-46 (DEVICE)

See Figure 30 for the various devices supported.

Use columns 40-46 to identify the input/output device to be used for the file. All entries must begin in column 40. The devices that can be used depend upon the form of the records (Figure 31).

Figure 33 shows the columns that cannot be used for the devices named. The shaded columns must be blank for the device named in the specification line. (MFCU is MFCU1 or MFCU2; MFCM is MFCM1 or MFCM2; PRINTER is PRINTER, PRINTR2, or PRINT84; DISK is DISK, DISK40, or DISK45; a blank entry is for device independent input or output files on Model 15.)

For discussions of RPG II support for the IBM 1442 Card Read Punch, see *IBM System/3 80-96 Conversion Program and RPG II Support for the IBM 1442 Card Read Punch Reference Manual*, SC21-7518. For information about the RPG II Telecommunications feature (BSCA), see the *IBM System/3 RPG II Telecommunications Programming Reference Manual*, SC21-7507.

Since the Model 15 support is identical to the Models 10 and 12 support, information concerning the use of the 3881 can be found in *IBM System/3 Model 10 Disk System IBM 3881 Optical Mark Reader Model 1 Program Reference and Logic Manual*, GC21-5103.

For the Model 15, information about the 1255 and 1419 Magnetic Character Readers is found in the *IBM System/3 Model 15 1255/1419 Magnetic Character Reader Reference and Program Logic Manual*, GC21-5132.

For a description of the IBM 3340 Disk Storage Drive, see the *IBM System/3 3340 Disk Storage Reference Manual*, GA26-1619.

CONSOLE (Printer-Keyboard, Model 10 Systems)

Figure 31 shows the file types that can be assigned to the printer-keyboard (CONSOLE). More than one printer-keyboard file may be described in a program.

Records entered from a printer-keyboard file will be treated as any other records. Every character to be entered must be keyed in. Key the information into the fields as you would into a card. Fields must be properly right-justified and left-justified by you. You must space where blanks appear in a record. The END key must be pressed after all characters have been keyed into a record.

Entry	Explanation	System supported On
MFCU1	5424 Multi-function Card Unit. The cards are in the primary hopper.	Models 10, 12, and 15
MFCU2	5424 Multi-function Card Unit. The cards are in the secondary hopper.	Models 10, 12, and 15
PRINTER	5203 Printer (Models 10 and 12) 1403 Printer (Models 10, 12, and 15). On the 5203, if the dual feed carriage feature is present, this entry refers to the left carriage.	Models 10, 12, and 15
PRINTR2	5203 Printer (Models 10 and 12). If the dual feed carriage is present, this entry refers to the right carriage.	Models 10 and 12
PRINT84	3284 Printer	Model 15
CONSOLE	Printer-Keyboard	Models 10 and 12
DISK	5444 Disk Storage Drive	Models 10, 12, and 15
DISK45	5445 Disk Storage (Equivalent to DISK40 on models with 3340 or 3344 Direct Access Storage facilities.)	Models 10, 12, and 15
TAPE	3410-3411 Magnetic Tape Unit	Models 10, 12, and 15
SPECIAL	Used for devices not supported by RPG language.	Models 10, 12, and 15
CRT77	CRT/Keyboard	Model 15
READ01	2501 Card Reader	Model 15
MFCM1	2560 Multi-function Card Machine. The cards are in the primary hopper.	Model 15
MFCM2	2560 Multi-function Card Machine. The cards are in the secondary hopper.	Model 15
READ42	1442 Card Read Punch	Models 10, 12, and 15
BSCA	Binary Synchronous Communications Adapter	Models 10, 12, and 15
No Entry	Device independent input or output file	Model 15
DISKET	3741 Data Station or Programmable Work Station (directly attached)	Models 10, 12, and 15
DISK40	3340 Direct Access Storage Facility (Models 12 and 15) or 3344 Direct Access Storage Facility (Program Number 5704-RG2 only)	Models 12 and 15

Figure 30. Devices supported

File	Form	Possible Devices
Primary or Secondary Input Files	Cards	MFCU1 or MFCU2, READ42, MFCM1 or MFCM2, READ01
	Disk	DISK, DISK40, or DISK45
	Tape	TAPE
	Keyed in by operator	CONSOLE or CRT77
	SIOC	SPECIAL
	TP lines	BSCA
	Diskette	DISKET
	Device independent	
Record Address Containing Record-Key Limits	Cards	MFCU1 or MFCU2, READ42, MFCM1 or MFCM2, or READ01
	Disk	DISK, DISK40, or DISK45
	Tape	TAPE
	Keyed in by operator	CONSOLE or CRT77
	Diskette	DISKET
	Device independent	
ADDROUT Files	Disk	DISK or DISK45
	Tape	TAPE
Demand Files	Cards	MFCU1 or MFCU2, READ42, MFCM1 or MFCM2, READ01
	Disk	DISK, DISK40, or DISK45
	Tape	TAPE
	Keyed in by operator	CONSOLE or CRT77
	Diskette	DISKET
	SIOC	SPECIAL
	TP lines	BSCA
	Device independent	

Figure 31 (Part 1 of 2). Device Assignment

File	Form	Possible Devices
Table Files	Cards	MFCU1 or MFCU2, READ42, MFCM1 or MFCM2, or READ01
	Disk	DISK, DISK40, or DISK45
	Tape	TAPE
	Diskette	DISKET
	Keyed in by operator	CONSOLE or CRT77
	Device independent	
Chained Input Files	Disk	DISK, DISK40, or DISK45
Update Files (primary, secondary, or chained)	Disk	DISK, DISK40, or DISK45
	Displayed output	CRT77
	SIOC	SPECIAL
Combined Files (primary or secondary)	Cards	MFCU1 or MFCU2, READ42, MFCM1 or MFCM2,
	SIOC	SPECIAL
	TP lines	BSCA (only for conversational reply)
Output Files	Cards	MFCU1 or MFCU2, READ42, MFCM1 or MFCM2
	Disk	DISK, DISK40, or DISK45
	Tape	TAPE
	Displayed output	CRT77
	Printed pages	PRINTER, PRINTR2, PRINT84, or CONSOLE
	SIOC	SPECIAL
	Diskette	DISKET
	TP lines	BSCA
	Device independent	
Display Files	Printed pages	CONSOLE
	Displayed output	CRT77

Figure 31 (Part 2 of 2). Device Assignment

If the operator presses CNCL (cancel), those characters of the record already accepted will be erased; the keying element will return to column 1, and the operator may begin to key the record in again. If the operator keys in more characters than are specified for a record, the record is automatically cancelled and the operator is notified to key it in again.

For use of the printer-keyboard in the display operation, see *Operation Codes, DSPLY*, in Chapter 8.

Note: When the printer-keyboard is used as an input device, it is suggested that some output to the printer-keyboard occur before input data is to be keyed in. This provides a visual indication in addition to the PROCEED light that data can be entered on the printer-keyboard.

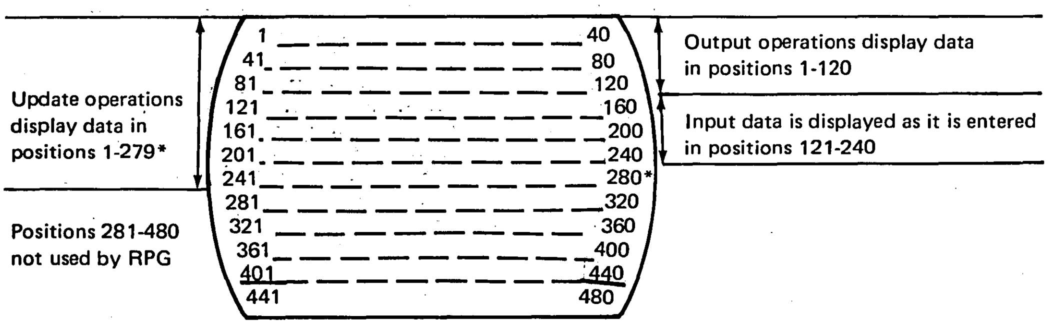
CRT/Keyboard Files (Model 15)

Figure 31 shows the file types that can be assigned to the CRT/keyboard. More than one CRT/keyboard file may be described in a program.

A record entered from the keyboard or displayed on the CRT will be treated as any other record. The data is displayed on the CRT as follows:

Every character to be entered must be keyed in. Key the information into the fields as you would into a card. Fields must be properly right-justified and left-justified by you. You must space where blanks appear in a record. Data may be entered when the message ENTER DATA Pn(n=the partition number) appears on the last line of the CRT. After all the data is entered, press ENTER to enter the data into the system. If an error is made while entering data, pressing ERASE INPUT causes those characters already entered to be erased and the cursor to reposition itself to position 121 for input files or to position 1 for update files. If more characters than are specified for a record are entered, the keyboard locks. Pressing RESET restores the keyboard, allowing you to correct and re-enter the record.

CAUTION
Only characters in the CRT character set can be displayed. Nondisplayable characters not in the X'80' through X'BF' range may be changed to displayable characters. Nondisplayable characters in the X'80' through X'BF' range are control characters that control the display of fields on the CRT. For a further explanation refer to the *IBM 3270 Information Display System Component Description,* GA27-2749. When the CRT is used as an update file, any characters changed during the output operation are returned during input as modified displayable characters. For example, if the units digit of a negative number is a zero, the internal representation is X'D0', which is not displayable. This non displayable character is changed to X'50' that is displayed as the & (ampersand). During an update operation this modified displayable character is reread from the CRT.



*Position 280 reserved for system use.

Note: CRT input data is specified as positions 1-120 on input specifications. See *Input Specifications*, chapter 7.

For use of the CRT/Keyboard with the DSPLY operation, see *Operation Codes, DSPLY,* in chapter 8.

Printer Files With Dual Feed Carriage Feature (Models 10 and 12 Only)

The dual carriage feature allows you to produce two separate printer output files in one program. The two output devices assigned to the printer must be named PRINTER and PRINTR2. The forms used for the two files are special forms such as checks or invoices that are narrower than the regular form for your printer. One form is controlled by the left carriage of the printer (device name PRINTER) and the other form is controlled by the right carriage of the printer (device name PRINTR2). The two printer files are considered as separate output files and must be described as such. A minimum of 17 print positions are lost between the two forms. Care must be taken, therefore, when describing the location (end position) of output fields, to avoid printing in positions where there is no form. Numbering of print positions is not affected when dual carriages are used; the first print position for PRINTR2 depends on where the forms are physically located on the carriage.

SPECIAL Device Support

You can process files using devices not supported by RPG II. To do this, you must indicate that the file will be handled by a SPECIAL device (SPECIAL in columns 40-46 of the File Description Sheet). You must also supply a subroutine to perform the I/O operations required to transfer data between the SPECIAL device and main storage (subroutine name in columns 54-59 of the File Description Sheet).

For a discussion of the file description specifications necessary for SPECIAL device support, see Appendix F.

Device Independent Input Files (Model 15)

Device Independent Input Files allow you to assign input devices during program execution and change input devices without recompiling the program. Input devices are assigned by a // FILE OCL statement. See Figure 32 for an example.

```
// FILE NAME-INPUT,UNIT-MFCU2
// FILE NAME-OUTPUT,UNIT-F1,PACK-F1F1F1,TRACKS-10,RETAIN-T
```

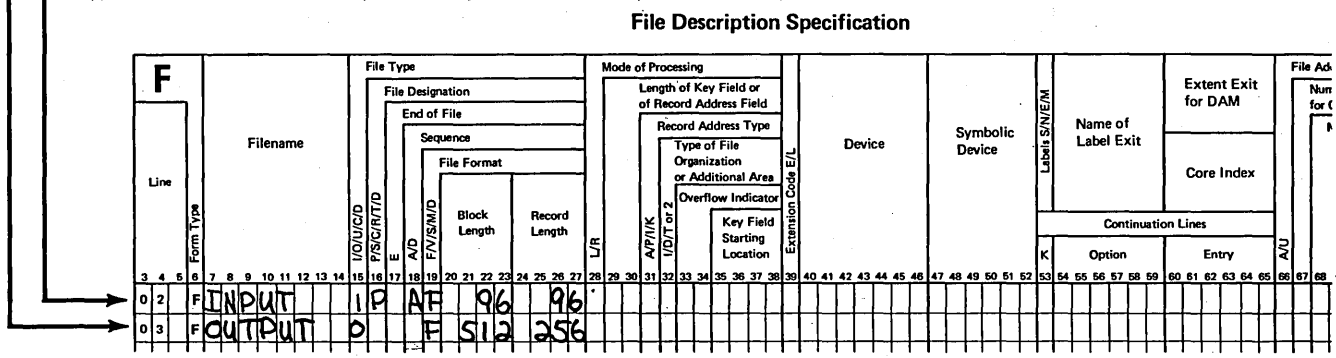


Figure 32. Coding Device Independent Input and Output Files

The following rules apply to these files:

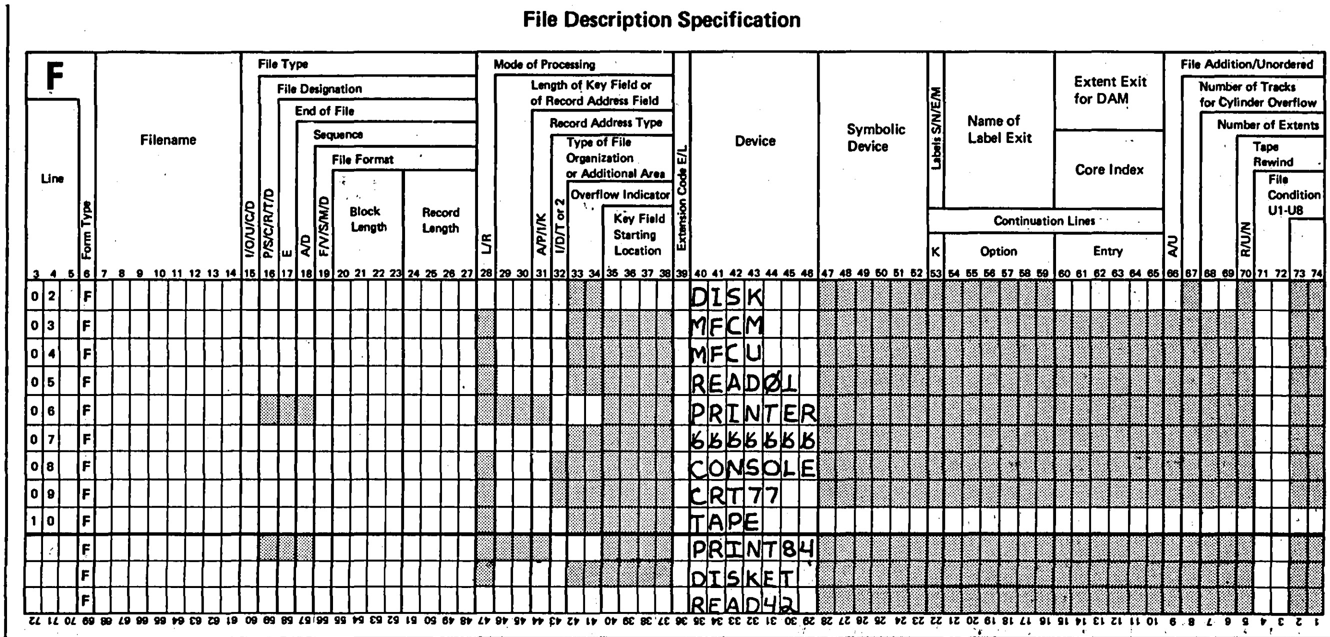
1. File Description Specifications (see Figures 33 and 52 for possible entries)
   - The DEVICE entry must be blank (column 40-46).
   - Record length can be 1-9999.
   - Block length can be a multiple of record length.
2. Double buffering is allowed.
3. Stacker select entries should not be used.
4. The device you assign to the device independent input file when you run your program must be different from other unit record input devices used in your program.
5. Packed or binary input is allowed from supported devices.
6. FROM file for table load or record address is allowed.
7. Variable length records are not allowed.
8. Multivolume disk files cannot be specified.
9. Only sequential file processing is allowed.

Device Independent Output Files (Model 15)

Device independent output files allow you to assign output devices during program execution and change output devices without recompiling the program. Output devices are assigned by a // FILE OCL statement. See Figure 32 for an example.

The following rules apply to these files:

1. File Description Specifications (See Figures 33 and 52 for possible entries)
   - The DEVICE entry must be blank (columns 40-46).
   - Record length can be 1-9999.
   - Block length can be a multiple of record length.
2. A Device independent output file can be specified as the TO file for table/array output or as factor 2 of a DEBUG operation.
3. Stacker select, space, skip, or fetch, specifications should not be used.
4. Packed or binary output is allowed on devices that support this type of data.
5. Card interpretation on the MFCU and MFCM is not allowed.
6. The device you assign to the device independent output file when you run your program must be different from other unit record output devices used in your program.
7. Multivolume disk files cannot be specified.
8. Record updating or record addition cannot be specified.
9. Variable length records are not allowed.
10. Only sequential file processing is allowed.



● Figure 33. Columns That Do Not Apply to Device Named

Shaded Columns Must be Blank

The following chart shows the possible devices that may be used for device independent input or output files:

	Device Independent Input Files	Device Independent Output Files
Device Type	Name in Unit Parameter of FILE Statement	Name in Unit Parameter of FILE Statement
5424	MFCU1 or MFCU2	MFCU1 or MFCU2
2560	MFCM1 or MFCM2	MFCM1 or MFCM2
2501	2501	
1442	1442	1442
5444	R1, R2, F1, or F2	R1, R2, F1, or F2
5445	D1, D2, D3, or D4	D1, D2, D3, or D4
3410/3411	T1, T2, T3, or T4	T1, T2, T3, or T4
1403		1403
3284		3284
System Input Device	READER	
System Punch Device		PUNCH
System Printer Device		PRINTER
3741	3741	3741
*3340	F1, F2, R1, or R2 D1, D2, D3, or D4	F1, F2, R1, or R2 D1, D2, D3, or D4

*With Model 15 SCP 5704-SC2, these codes can also be specified: D31, D32, D33, D34, D41, D42, D43, and D44.

COLUMNS 47-52

Columns 47-52 are not used.

COLUMNS 53-65 (CONTINUATION LINES)

Column 53

Entry	*Explanation*
K	Continuation record

Continuation records provide additional information about the TAPE file or SPECIAL file being defined. One or two continuation records can be specified for each tape file, and one continuation record can be specified for each SPECIAL file. When specifying a continuation record, columns 54-59 (Continuation Line Option) must be coded and columns 60-65 (Continuation Line Entry) may also need to be coded. Figure 34 shows an example of the coding necessary on the File Description Sheet for a continuation record.

A continuation record for DISK, DISK40, or DISK45 will provide an additional amount of main storage for the index buffer (Model 15 only).

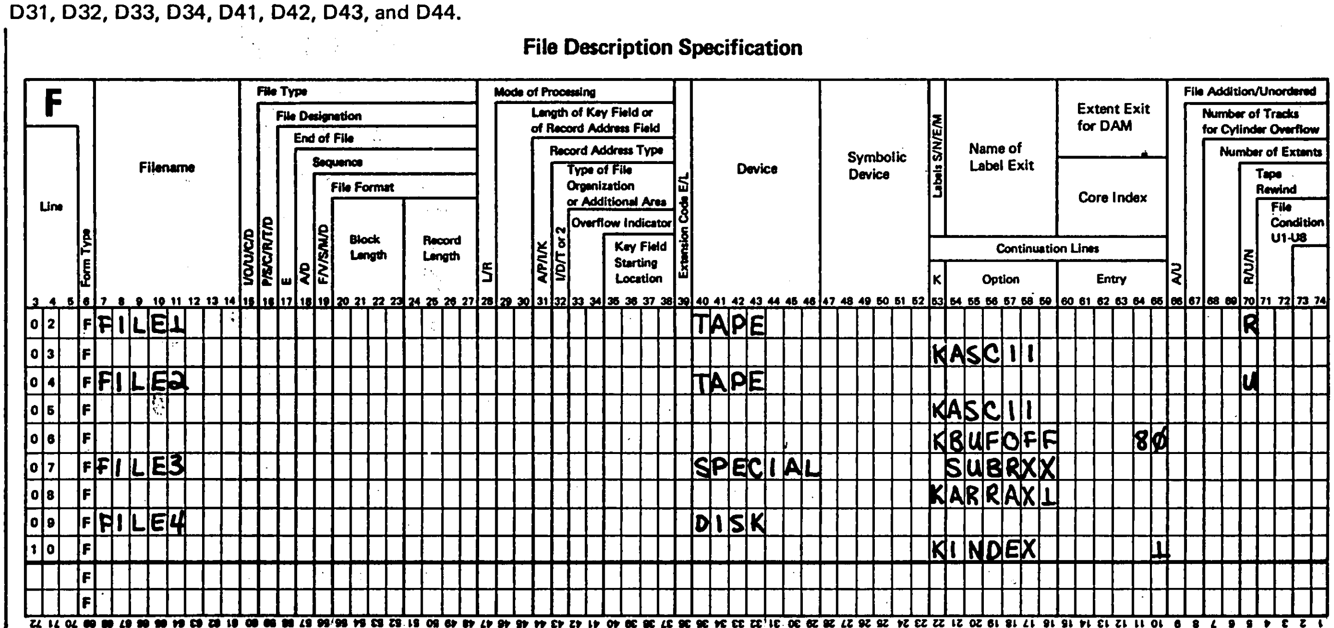


Figure 34. Continuation Record

COLUMNS 54-59 (CONTINUATION LINES OPTION)

Entry	*Explanation*
ASCII	Tape file defined as an ASCII (American Standard Code for Information Interchange) file.
BUFOFF	Tape input file contains a block prefix.
Table/ array name	Name of table/array to be used by user-written IOS subroutine. The array name cannot be ASCII, BUFOFF, or INDEX.
INDEX	Provide expanded index buffer, in main storage for index files (Model 15).

BUFOFF can only be used for ASCII files. Therefore, if BUFOFF is entered, ASCII must also be entered. BUFOFF also requires an entry in columns 60-65 (Continuation Lines Entry). Array name can only be used if SPECIAL device support is specified for the file.

If INDEX is specified the amount of main storage provided for the index buffer can be specified in columns 60-65 (Model 15 only). This entry applies to all 5444 and 5445 indexed files except for the index random input or update files with no ADD specified, and the index output with no ADD specified.

COLUMNS 60-65 (CONTINUATION LINE ENTRY)

Entry	*Explanation*
0-99	Length of the block prefix in an ASCII tape input file that specifies BUFOFF.
1-9	Amount of storage, in sector increments, to be provided for the index buffer. (Example: 1 sector = 256 bytes, 2 sectors = 512 bytes, . . . , 9 sectors = 2304 bytes.)

An entry must be specified in these columns if BUFOFF has been specified in columns 54-59. This entry cannot be specified for tape output files. The entry must end in column 65 (right justified).

COLUMN 53 LABELS

Column 53 for labels is not used.

COLUMN 54-59 NAME OF LABEL EXIT

Entry	*Explanation*
SUBRxx	Name of the user-written subroutine which will perform the I/O operation for a SPECIAL device. (x = any alphabetic character.)
SRyzzz	Name of the IBM written subroutine (six-character name in library is $$yzzz which will perform the I/O operation for a device supported by SPECIAL. (y = any of the following 15 characters: B, C, D, F, G, H, I, L, M, O, P, R, S, T, or U; z = any of the following 16 characters: A, B, C, D, F, G, H, I, L, M, O, P, R, S, T, or U.)
Blank	No SPECIAL device is being used.

Note: Subroutines of the type SRyzzz are overlayable. Modifications within the subroutine code may or may not be present the next time the subroutine is used.

Columns 54-59 must contain an entry for each data file assigned to a SPECIAL device. These columns are used to specify the subroutine which will perform the input/output operations for a file assigned to a SPECIAL device. The subroutine name entered in columns 54-59 can be from four to six characters long. The first four characters must be SUBR; the remaining characters can be any alphabetic characters.

COLUMNS 60-65 (CORE INDEX)

Entry	*Explanation*
6-9999	Number of bytes reserved for the core index and highest added key.
Blank	No core index will be built.

Columns 60-65 apply only to indexed files processed randomly using the CHAIN operation code. Core index cannot be specified in shared I/O. Entries must be right-justified. Leading zeros are not required. You can specify up to 9999 bytes for the core index. This will usually provide for faster retrieval of records.

The core index is a table containing entries for tracks in the index portion of a data file. Each entry contains a track address and the lowest key field associated with the next track. Figure 35 shows the layout on disk of the index for the indexed file, INDEXT, which contains 1000 records. Since all index entries are contained on three tracks, the core index for INDEXT shown in Figure 36 contains only three entries, one per track. Each core index entry contains the low key on the next track and the track address.

Use of the core index can significantly reduce the amount of time needed to process an indexed file because it enables the system to go more directly to the specific record you want. With the core index, the system can find a specific record by searching only a small part of the file index.

Without the core index all index entries that precede the record you want must be searched. Using the core index shown in Figure 35 record 767 can be found in this manner:

1. Search the core index until the first key field higher than record 767 is located. In this instance the key is 769, on track *C*. Since 769 is the low key on track *C*, key 767 must reside on track *B*.

2. Search track *B* in the file index until key 767 is located.

3. Chain directly to the associated data record.

In columns 60-65 you specify the number of storage positions (bytes) you wish reserved for the core index. Using the amount of core storage you specify, the system builds the most efficient core index it can. The core index is built immediately before your RPG II program is executed.

For single volume indexed random add on a 5445 disk, you must add one key length to the size of the core index you specify. The extra key length will contain the highest added key. Use of the save area for the highest key provides a significant performance increase, especially when the keys being added are in ascending sequence. The RPG II program does not need to search the previously added keys for duplicates if the key being added is higher than the highest added key in the save area.

For multivolume indexed random processing on a 5445 disk, RPG II automatically reserves the minimum space required to provide two core index entries. In addition, for multivolume indexed random add on a 5445 disk, RPG II automatically reserves one high key save area per online volume.

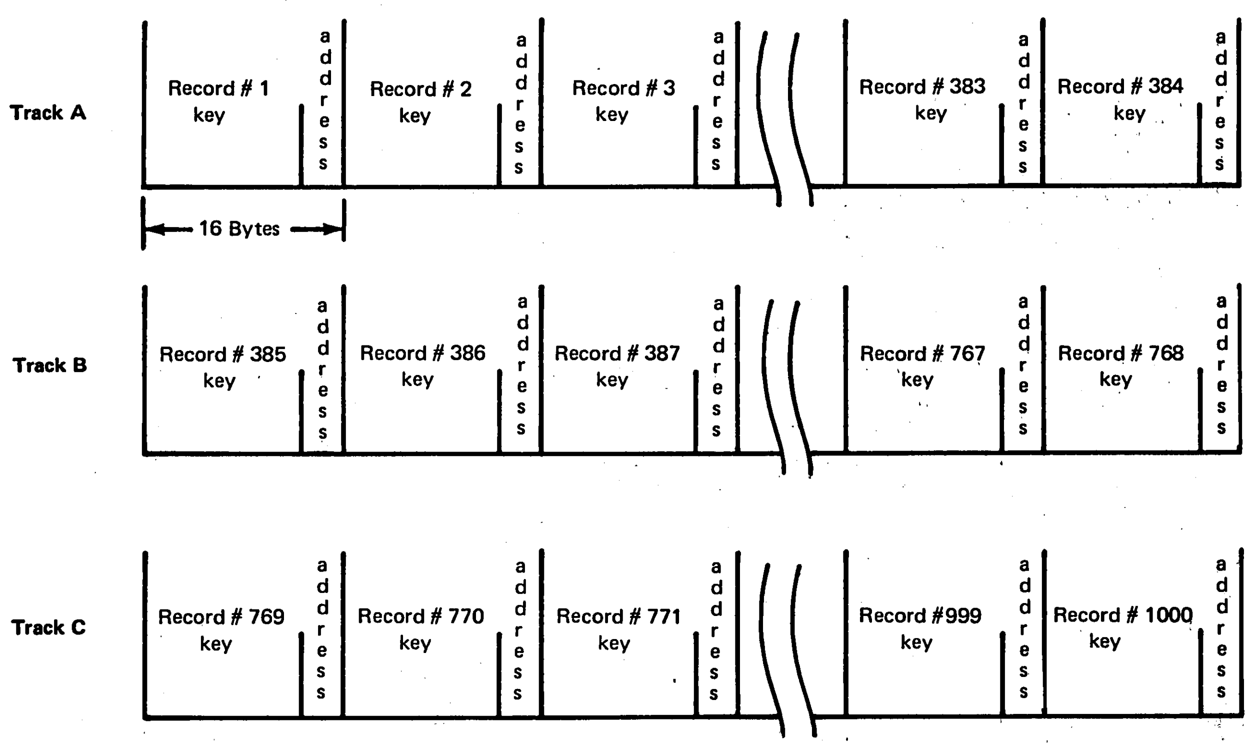


Figure 35. Disk Layout of the Index for INDEXT

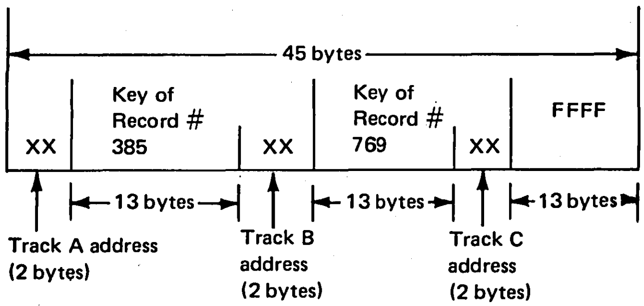


Figure 36. Core Index for INDEXT

For most efficient processing, the core index should be large enough to contain one entry (low key and track number) for each track of index in the data file. Therefore, the most efficient size for the core index is equal to key field length plus 2, multiplied by the number of tracks in the file index (plus one key length per online volume if a 5445 indexed random add). For the indexed file, INDEXT, in Figures 35 and 36, the entry in columns 60-65 would be 45:

```
   13 (key length)
+   2 (length of a track address)
   15 (length of a core index entry)
x   3 (number of file index tracks)
   45 (size of core index for 5444)
+  13 (if 5445 indexed random add, for highest added key)
   58 (size of core index for 5445 indexed random add)
```

For 5444 disk, if the storage space you specify in columns 60-65 is not large enough to contain one entry for each track of file index, RPG II may construct a core index containing one entry for every cylinder of the file index or, perhaps, for every other cylinder. However, the entry in columns 60-65 will be ignored if there is insufficient space for at least two index entries.

For a 5445 disk, if the storage space you specify in columns 60-65 is not large enough to contain one entry for each track of file index, RPG II may construct a core index containing one entry for every other track, every fourth track, etc. However, for a single volume file, the entry in columns 60-65 will be ignored if there is insufficient space for at least one index entry (after reserving space for one high key save area, if adding records). For multivolume files, the entry in columns 60-65 is added to the minimum space automatically reserved by the RPG II program. For a more detailed description of performance considerations, see the *IBM System/3 Disk Concepts and Planning Guide,* GC21-7571.

COLUMN 66 (FILE ADDITION)

Entry	*Explanation*
A	New records will be added to the file.
U	Records for an indexed file are to be loaded in unordered sequence.

Column 66 applies to sequential and indexed disk files. This column indicates:

1. The program is to add new records to the file (see *Examples, Example 1*).

2. Records are to be loaded in an unordered sequence (see *Examples, Example 2*).

Records added to a sequential file are added at the end of the file. To add records to a sequential file, the file must be an output file (O in column 15 of the File Description Sheet).

Records added to an indexed file are added at the end of the file and entries for the new records are made in the index. The index is then reorganized so that the record keys (including the new ones) are in ascending order.

Neither unordered sequence nor file addition in column 66 can be specified for indexed files from which records are read using the sequential-within-limits-method. For more efficient operation, records added to an indexed file should be in ascending sequence. New records may be added to a direct file by specifying the file as an update file processed consecutively or by the CHAIN operation code.

After a file has been loaded on disk, it may be necessary to add records to the file. Records can be added at detail, total, or exception time during the program cycle. When records are to be added to an indexed file randomly, the records to be added may:

1. Contain keys that are above the highest presently in the file. In this case, the records constitute an extension of the file.

2. Contain keys that are either lower than the lowest presently in the file, or fall between those already in the file.

If records are to be added to an indexed file sequentially:

1. The key to be added must be lower than the key retrieved and higher than the preceding key, or

2. The file must be at end of file, for single volume files only.

If neither of the above conditions exists a halt occurs; otherwise, the record is added. (See Appendix A for a discussion of halts and operator options.)

Unordered Load (U in column 66) is specified when an indexed file is to be built from records in an unordered sequence. After records have been loaded and an index built in the unordered sequence, the index is sorted into ascending sequence.

In Figure 37, combinations of entries in file type (column 15) and file addition (column 66) show the functions that can be performed for indexed files (*I* in column 32).

If a large number of records are to be added to the file, the time required for the index sort can be decreased by allocating a special work file. This requires no special RPG II coding but does require a special OCL statement. For additional information and an example, see the appropriate SCP reference manual listed under *Related Publications* in the Preface.

Examples

Example 1

Figure 38 shows how records can be added to an indexed disk file. The new records are contained in a card file, CARDIN. The file INDEXED is the existing disk file to which new records will be added. A printer file, PRINT, will provide a report showing all the records in CARDIN, with an indication of which records were added to INDEXED and which records were not added.

On the File Description Sheet, an *A* must appear in column 66 for the file INDEXED, and on the Output Sheet ADD must appear in columns 16-18 for the new record to be added.

As defined on the Input Sheet, all the cards in CARDIN should have an *A* in position 80. The code identifies a record to be added to the disk file, and this record type is assigned indicator 01. On the Output Sheet, notice that when 01 is on, the data from the card is written on the disk file INDEXED and is also printed on the file PRINT to keep a visual report of new records.

Column 15	Column 66	Function
O	Blank	Load records in ascending key sequence to an indexed file.
O	U	Load records in unordered key sequence to an indexed file.
O	A*	Add records to an existing indexed file.
I	Blank	Read records of an indexed file without adding new records or updating records.
I	A*	Read records of an indexed file and add new records to the file that are not presently there. No updating is performed.
U	Blank	Update records of an indexed file without adding new records.
U	A*	Update records of an indexed file and add new records to the file.

*An A in column 66 requires an ADD entry in columns 16-18 of the Output-Format Sheet.

Figure 37. Various Functions Performed on Indexed Files

File Description Specification

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 or Additional Area	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels S/N/E/M	Name of Label Exit	File Addition/Unordered	Number of Tracks for Cylinder Overflow
02	F	CARDIN	I	P	E		F	96	96								MFCU1					
03	F	INDEXED	O				F	256	64		10	A	I		1		DISK				A	01
04	F	PRINT	O				F	132	132					OA		L	PRINTER					
05	F																					
06	F																					

Figure 38 (Part 1 of 2). File Addition

Line Counter Specifications

Line (3-5)	Form Type (6)	Filename (7-14)	1: Line Number (15-17)	1: FL or Channel Number (18-19)	2: Line Number (20-22)	2: OL or Channel Number (23-24)	3–12: Line Number / Channel Number (25-74)
11	L	PRINT	66	FL	6Ø	OL	
12	L						
	L						

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9094 U/M 050* Printed in U.S.A.

Program		Punching Instruction	Graphic		Card Electro Number	Page (1-2)	Program Identification (75-80)
Programmer	Date		Punch			Ø3 of ___	

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2 (28-34)	3 (35-41)	Stacker Select (42)	P/B/L/R (43)	Field Location From (44-47)	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)	(71-74)
01	I	CARDIN	AA			Ø1	8Ø		C	A															
02	I	OR				Ø2	8Ø	N	C	A															
03	I														1	1Ø		KEY							
04	I														12	2Ø		FIELD1							
05	I														21	3Ø		FIELD2							
06	I														31	64		MORE							
07	I																								

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9090 U/M 050* Printed in U.S.A.

Program		Punching Instruction	Graphic		Card Electro Number	Page (1-2)	Program Identification (75-80)
Programmer	Date		Punch			Ø4 of ___	

Commas	Zero Balances to Print	No Sign	CR	-	
Yes	Yes	1	A	J	X = Remove Plus Sign
Yes	No	2	B	K	Y = Date Field Edit
No	Yes	3	C	L	Z = Zero Suppress
No	No	4	D	M	

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T/E) (15)	Stacker #/Fetch (F) (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators (23-31)	Field Name (32-37) *AUTO	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40-43)	P/B/L/R (44)	Constant or Edit Word (45-70)	(71-74)
01	O	INDEXED	D	A	D	D			Ø1							
02	O									KEY			1Ø			
03	O									FIELD1			2Ø			
04	O									FIELD2			3Ø			
05	O									MORE			64			
06	O	PRINT	H		3		Ø3		1P							
07	O	OR							OA							
08	O												84		'SUMMARY OF RECORDS ADDED'	
09	O												1ØØ		' TO INDEXED FILE'	
10	O		D						Ø1							
11	O	OR							Ø2							
12	O									KEY			5Ø			
13	O									FIELD1			6Ø			
14	O									FIELD2			7Ø			
15	O									MORE			1Ø4			
16	O								Ø2				125		'RECORD NOT ADDED'	

Figure 38 (Part 2 of 2). File Addition

There may be records in CARDIN that do not belong in that file, or some records may have a keypunch error. These records are identified on the Input Sheet as not having the character *A* in position 80. These records will turn on indicator 02, and are not to be added to the disk file INDEXED. On the Output Sheet, the constant RECORD NOT ADDED is printed only on indicator 02, indicating a record that was not added to the disk file. In this manner, there will be a printed report of all records in CARDIN, and the records not added to INDEXED are identified by the constant RECORD NOT ADDED.

Example 2

Figure 39 shows the unordered loading of an indexed disk file from an unsequenced input card file. The output file, MASTER, is described as an indexed file to be loaded and processed by record keys. The *U* in column 66 of the File Description Sheet indicates that an unordered load is to be done. The input file, CARDS, is described on the Input Sheet as being without sequence.

The keys from which the index is to be built appear as the first eight positions of the output record. As the disk file is loaded, the key is extracted from the record and an index entry is built including the location of the record on disk. After the entire file has been loaded and an index entry has been constructed for each record, the index entries are sorted into ascending sequence.

COLUMN 67

Column 67 is not used.

COLUMNS 68-69 (NUMBER OF EXTENTS)

Entry	*Explanation*
Blank	Single volume file
1-50	Number of volumes that contain the file.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 UM/050* Printed in U.S.A.

Program | Punching Instruction | Graphic | Punch | Card Electro Number | Page ØI of | Program Identification (75 76 77 78 79 80)

Programmer | Date

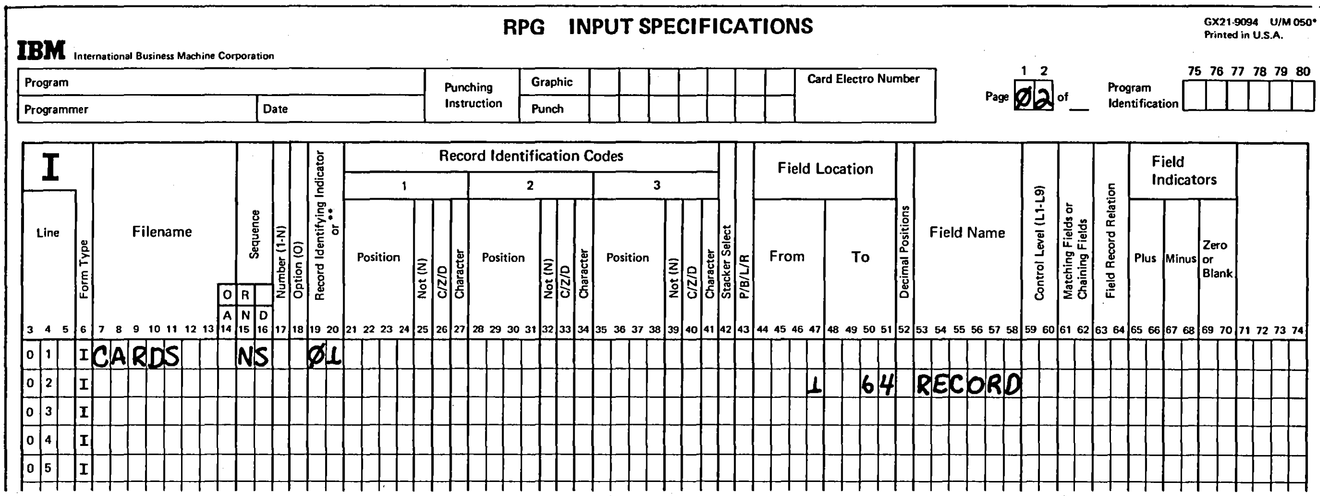
Control Card Specifications

Line	Form Type	Core Size to Compile	Object Output	Listing Options	Core Size to Execute	Debug	MFCM Stacking Sequence	Inverted Print	360/20 2501 Buffer	Number Of Print Positions	Alternate Collating Sequence	Address to Start (Model 20)	Work Tapes	Overlay Open	Overlap Printer	Binary Search	Tape Error	2152 Checking	Inquiry	Read/Write/Compute (Model 20)	Keyboard Output	Sign Handling	1P Forms Position	Indicator Setting	File Translation	Punch MFCU Zeros	Nonprint Characters	Table Load Halt	Shared I/O	Field Print	Formatted Core Dump	RPG to RPG II Conversion	Refer to the specific System Reference Library manual for actual entries.
3-5	6	7-9	10	11	12-14	15	16	21	22	23-25	26	27-30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	47	48	49	50	51	52-74
01	H																																

File Description Specification

Line	Form Type	Filename	File Type (I/O/U/C/D)	File Designation (P/S/C/R/T/D)	End of File (E)	Sequence (A/D)	File Format (F/V/S/M/D)	Block Length	Record Length	L/R	Length of Key Field or of Record Address Field	Record Address Type (A/P/I/K)	Type of File Organization or Additional Area (I/D/T or 2)	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels S/N/E/M	Name of Label Exit / Continuation Lines K, Option	Extent Exit for DAM / Core Index / Entry	File Addition/Unordered (A/U)	Number of Extents	Tape Rewind (R/U/N)	File Condition U1-U8
3-5	6	7-14	15	16	17	18	19	20-23	24-27	28	29-30	31	32	33-34	35-38	39	40-46	47-52	53	54-59	60-65	66	68-69	70	71-72
02	F	CARDS	I	P	E		F	96	96								MFCU1								
03	F	MASTER	O				F	256	64		8	A	I		1		DISK					U	Ø1		
04	F																								
05	F																								
06	F																								
07	F																								

Figure 39 (Part 1 of 2). Unordered Loading of an Indexed File



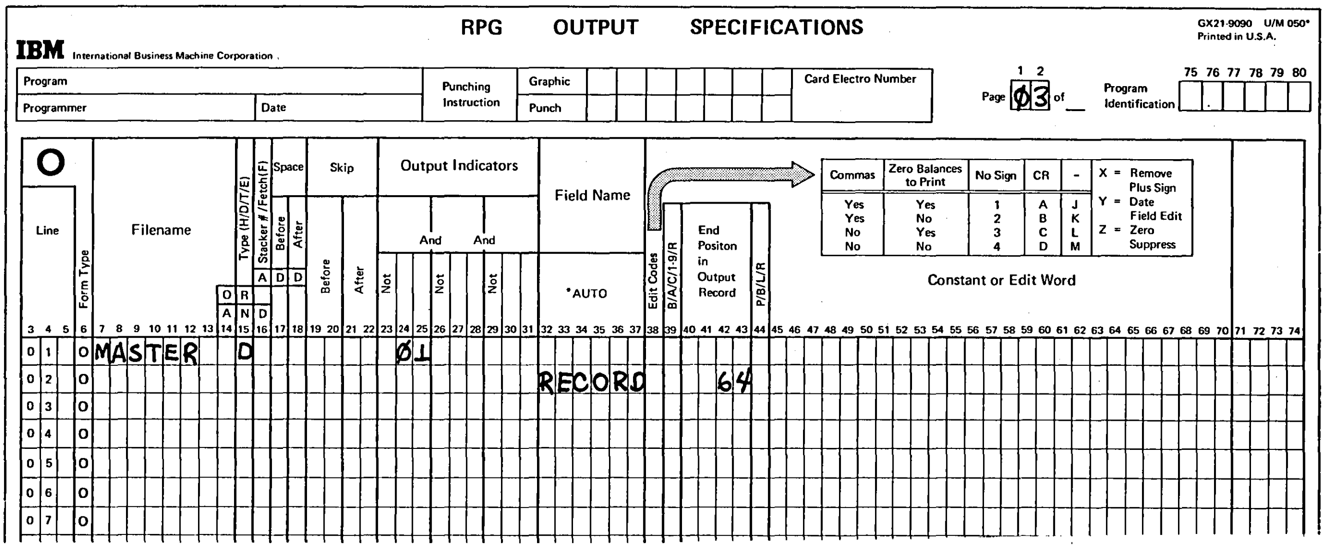


Figure 39 (Part 2 of 2). Unordered Loading of an Indexed File

The entry must end in column 69. These columns define the number of volumes (disks) on which the disk file is located. A disk file must occupy consecutive cylinders on each volume. For instance, a disk file could not occupy cylinders 20-30 and 41-50 on one volume. The file could occupy cylinders 20-40 on that volume, or the data in cylinders 41-50 could be placed on another volume.

The number of volumes you can use depends on the mode of processing and number of drives used. For single volume files the entry in columns 68-69 must be 01 or blank. The Number of Extents entry in columns 68-69 must not be greater than 01 when sequential processing within limits (L in column 28 and A in column 31) or an unordered load (U in column 66) is specified for the file.

For multivolume files, determine the entry as follows:

1. *Consecutive Processing.* A disk file to be processed consecutively can be located on a fixed disk, a removable disk, or both if the entire file is online during processing. However, when portions of the file are offline during processing, the file must be located on removable disks only. To determine the entry in columns 68-69 for a multivolume file on 5444 disks, see Figure 40. For a multivolume file on 5445 disks, the entry in columns 68-69 must be 02 (04 on Model 15) if the entire file is online or 02-50 if portions of the file are offline. The maximum number of offline, sequential volumes is 40. When the system is running under SCP 5704-SC2, the maximum number of offline sequential volumes is 192.

2. *Indexed Sequential or Random Processing by Keys.* A disk file to be processed indexed sequentially or randomly by keys can be located on a fixed disk, a removable disk, or both if the entire file is online during processing. However, when portions of the file are offline during processing, the file must be located on removable disks only. To determine the entry in columns 68-69 for a multivolume file on 5444 disks, see Figure 40. For a multivolume file on 5445 disks, the entry in columns 68-69 must be 02 (04 on Model 15) if the entire file is online or 02 or greater if portions of the file are offline. When the system is using SCP 5704-SC2, the maximum number of offline indexed volumes is 96.

   Note: For indexed random processing, the volumes must be processed sequentially. However, the records on each volume may be processed randomly.

3. *Random Processing by Relative Record Number.* A disk file to be processed randomly by relative record number can be located on a fixed disk, a removable disk, or both. To process a multivolume disk file randomly by relative record number, the entire file must be available to the system at any given time. Therefore, the entire file must be online. To determine the entry in columns 68-69 for a multivolume file on 5444 disks, see Figure 40. For a multivolume file on 5445 disks, the entry in columns 68-69 must be 02.

Multivolume processing cannot be used with shared I/O. Additional information on creating and processing multivolume files, including Operation Control Language statements is contained in the appropriate SCP reference manual listed under *Related Publications* in the Preface.

COLUMN 70 (TAPE REWIND)

Entry	*Explanation*
R	Rewind tape at end of file.
U	Unload tape at end of file.
N	Leave tape at end of file.

Column 70 is used only with tape files to control the rewinding and unloading of tapes. This entry specifies what the system should do with the tape after the tape files have been processed. These entries may be overridden by the END parameter on the FILE statement.

If column 70 is left blank, the tape rewind information specified at program execution time is assumed.

COLUMNS 71-72 (FILE CONDITION)

Entry	*Explanation*
U1-U8	The file is conditioned by the specified external indicator.
Blank	The file is not conditioned by an external indicator.

	ONE DRIVE	TWO DRIVES
	Maximum number of volumes allowed	Maximum number of volumes allowed
Consecutive processing, or indexed sequential or random processing by keys (offline—removable disks only)	50	50
Consecutive processing, or indexed sequential or random processing by keys (online—removable or fixed disks)	2	4
Random processing by relative record number (online—removable or fixed disks)	2	4

Figure 40. Number of Volumes Allowed for Multivolume Files (5444 Only)

Columns 71-72 apply to primary and secondary input (excluding table input files), update, output, display, and combined files. A record address file may be conditioned by an external indicator which is off, it will be in end of file status. Chained and demand files may be conditioned by external indicators. If an output file is conditioned by an external indicator which is off, records will not be written on that file. Any calculation operations which should not be done when the file is not in use should also be conditioned by the same indicator. When the indicator is off, the file is treated as though the end of file had been reached (that is, no records can be read from or written in the file). If a disk file is conditioned by an external indicator which is off, the FILE OCL statement for that file should be removed.

Note: Information on setting external indicators (*SWITCH* OCL statement) can be found in the appropriate SCP reference manual listed under *Related Publications* in the Preface.

U1-U8 (External Indicators)

Indicators U1-U8 are external indicators. This means they are set prior to processing by Operation Control Language. Their setting cannot be changed during processing. Thus, the program has no control over them.

You may use these indicators as file conditioning indicators. They tell whether or not a certain file is to be used for a job. For example, you may have a job which one time requires the use of two output (or input) files and another time the use of only one. Instead of writing two different programs (one using one file, the other two), you can condition a file (in the file description specifications) by an external indicator. When the indicator is on, the file is used; when it is off, the file is not used.

If a file is conditioned by an external indicator, output data handled by the file can also be conditioned by the same indicator. If an input file is conditioned by an external indicator which is off it will be in end of file status. If an output file is conditioned by an external indicator which is off records will not be written on that file. Any calculation operations which should not be done when the file is not in use should also be conditioned by the same indicator.

In addition to using these indicators as file conditioning indicators, you may use them:

1. To condition calculation operations.
2. To condition output operations.
3. As field record relation indicators (columns 63-64 of Input Specifications Sheet).

COLUMNS 73-74

Columns 73-74 are not used.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

See Chapter 2.

FILE DESCRIPTION CHARTS

The File Description Charts in the following pages (Figures 41-54) are for:

1. *Disk* files, presented by disk file organization and processing method.
2. *MFCU, Console,* and *Printer* files.
3. *Tape* files.
4. *MFCM,* 2501, 1442, Device independent, and CRT/Keyboard files.
   - The *entries* in the chart must be made for the processing method and type of file described on that line.
   - The *shaded columns* must be blank for the file described on that line.
   - The *other columns* may be required or optional, but cannot be indicated on the chart because the entries represent information that changes from program to program.
5. *Diskette* files

If you are updating an indexed disk file using the CHAIN operation code, look at the chart for *indexed* disk files, random processing by CHAIN operation code. Then choose the chained update file with or without record addition.

The entries on the chart must be made for the file you are describing. The shaded columns must be blank for that file.

The remaining columns represent information that changes from program to program. For instance, in this example these columns are required but may change from one program to another: Filename, Record Length, Length of Key Field, and Key Field Starting Location. Optional entries are: End of File, Sequence, File Condition, Line, Block Length, Number of Extents, and Core Index.

DISK FILES

Figure 41. Processing Methods for *Indexed* Disk Files

File Description Specification

Type of Processing		Line (3-5)	Form Type (6)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D (16)	File Format F/V/S/M/D (19)	L/R (28)	A/P/I/K (31)	I/D/T or 2 (32)	Device (40-46)	A/U (66)
Sequential*	by Key, no ADD	02	F	I	P	F		A	I	DISK	
	by Key, no ADD	03	F	I	S	F		A	I	DISK	
	by Key, with ADD	04	F	I	P	F		A	I	DISK	A
	by Key, with ADD	05	F	I	S	F		A	I	DISK	A
	by Key, no ADD	06	F	U	P	F		A	I	DISK	
	by Key, no ADD	07	F	U	S	F		A	I	DISK	
	by Key, with ADD	08	F	U	P	F		A	I	DISK	A
	by Key, with ADD	09	F	U	S	F		A	I	DISK	A
	by Limits	10	F	I	P	F	L	A	I	DISK	
	by Limits	11	F	I	S	F	L	A	I	DISK	
	by Limits	12	F	U	P	F	L	A	I	DISK	
	by Limits	13	F	U	S	F	L	A	I	DISK	
Sequential* (demand files processed by the READ operation code)	by Key, no ADD	14	F	I	D	F		A	I	DISK	
	by Key, with ADD	15	F	I	D	F		A	I	DISK	A
	by Key, no ADD	16	F	U	D	F		A	I	DISK	
	by Key, with ADD	17	F	U	D	F		A	I	DISK	A
	by Limits †	18	F	I	D	F	L	A	I	DISK	
	by Limits †	19	F	U	D	F	L	A	I	DISK	
Random	by CHAIN, no ADD	20	F	I	C	F	R	A	I	DISK	
	by CHAIN, with ADD	21	F	I	C	F	R	A	I	DISK	A
	by CHAIN, no ADD	22	F	U	C	F	R	A	I	DISK	
	by CHAIN, with ADD	23	F	U	C	F	R	A	I	DISK	A
		24	F								
	by ADDROUT	25	F	I	P	F	R	I	I	DISK	
	by ADDROUT	26	F	I	S	F	R	I	I	DISK	
	by ADDROUT	27	F	U	P	F	R	I	I	DISK	
	by ADDROUT	28	F	U	S	F	R	I	I	DISK	
		29	F								
Load	Unordered	30	F	O		F		A	I	DISK	U
	Ordered	31	F	O		F		A	I	DISK	
		32	F								
Add records only	ADD only	33	F	O		F		A	I	DISK	A

* Sequential processing by key or limits must use the file index, which is always arranged in ascending sequence. When an indexed file is processed record by record from beginning to end, the file is processed through the index using the sequential by key method.

† If no record address file is associated with a limits file it is assumed to be processed using the SETLL operation code.

Notes:

1. DISK, DISK40, or DISK45 can be specified as the device (columns 40-46).
2. Read-only operations are allowed on consecutively processed indexed disk files.

DISK FILES

File Description Specification

Type of Processing	Line	Form Type	File Type (15)	File Designation (16)	File Format (19)	Mode of Processing L/R (28)	A/P/I/K (31)	Extension Code E/L (39)	Device (40-46)	A/U (66)
Consecutive (The entire file is read from beginning to end.) by READ	02	F	I	P	F				DISK	
	03	F	I	S	F				DISK	
	04	F	I	T	F			E	DISK	
	05	F	U	P	F				DISK	
	06	F	U	S	F				DISK	
	07	F	I	D	F				DISK	
	08	F	U	D	F				DISK	
	09	F								
Random by CHAIN	10	F	I	C	F	R			DISK	
Random by CHAIN	11	F	U	C	F	R			DISK	
	12	F								
Random by ADDROUT	13	F	I	P	F	R	I		DISK	
Random by ADDROUT	14	F	I	S	F	R	I		DISK	
Random by ADDROUT	15	F	U	P	F	R	I		DISK	
Random by ADDROUT	16	F	U	S	F	R	I		DISK	
	17	F								
Load (The file is written on disk as entered)	18	F	O		F				DISK	
	19	F								
Load or Add records	20	F	O		F				DISK	A
		F								
		F								

Note: DISK, DISK40, or DISK45 can be specified as the device (columns 40-46).

Figure 42. Processing Methods for *Sequential* Disk Files

DISK FILES

File Description Specification

Type of Processing		Line	Form Type (6)	Filename (7-14)	File Type (15) I/O/U/C/D	File Designation (16) P/S/C/R/T/D	End of File (17)	Sequence (18)	File Format (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing (28) L/R	Length of Key Field or of Record Address Field (29-30)	Record Address Type (31) A/P/I/K	Type of File Organization or Additional Area (32)	Device (40-46)
Consecutive (The entire file is read from beginning to end.)	by READ	02	F		I	P			F							DISK
		03	F		I	S			F							DISK
		04	F		U	P			F							DISK
		05	F		U	S			F							DISK
		06	F		I	D			F							DISK
		07	F		U	D			F							DISK
		08	F													
Random	by CHAIN	09	F		I	C			F			R				DISK
	by CHAIN	10	F		U	C			F			R				DISK
		11	F													
	by ADDROUT	12	F		I	P			F			R		I		DISK
	by ADDROUT	13	F		I	S			F			R		I		DISK
	by ADDROUT	14	F		U	P			F			R		I		DISK
	by ADDROUT	15	F		U	S			F			R		I		DISK
		16	F													
Load	Disk addresses are developed for each record entered.	17	F		O	C			F			R				DISK

* Records are inserted or changed in a direct file by defining the file as an update processed consecutively, or an update file processed randomly by the CHAIN operation code.

Note: DISK, DISK40, or DISK45 can be specified as the device (columns 40-46).

Figure 43. Processing Methods for *Direct* Disk Files

Figure 44. Record Address Files Located on Disk

DISK FILES

File Description Specification

Record Address Files *

1. ADDROUT Files
2. Record Key Limits

Line (3-5)	Form Type (6)	Filename (7-14)	File Type (15)	File Designation (16)	End of File (17)	Sequence (18)	File Format (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing (28)	Length of Key Field or of Record Address Field (29-30)	Record Address Type (31)	Type of File Organization or Additional Area (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)
02	F		I	R			F		3		3	I	T			E	DISK
03	F																
04	F		I	R			F									E	DISK
05	F																
06	F																
07	F																
08	F																
09	F																
10	F																
	F																
	F																

* ADDROUT files may be associated with indexed, or direct disk files or sequential disk or tape files.

* Record address files containing record key limits may only be associated with indexed disk files, but may be a disk, 1442, 2501, CRT77, tape, MFCM, MFCU, console, or DISKET file (see charts for MFCU and console files).

Note: DISK, DISK40, or DISK45 can be specified as the device (columns 40-46).

MFCU FILES

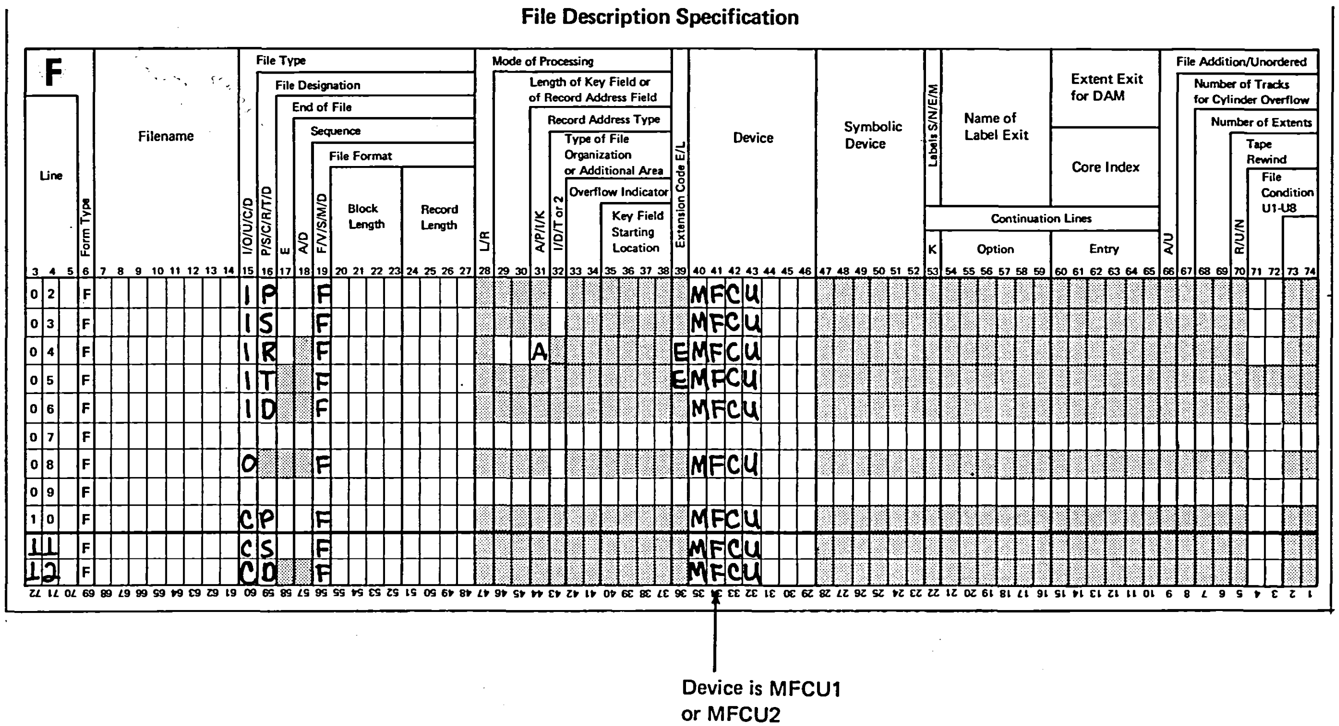


Figure 45. MFCU Files

CONSOLE FILES

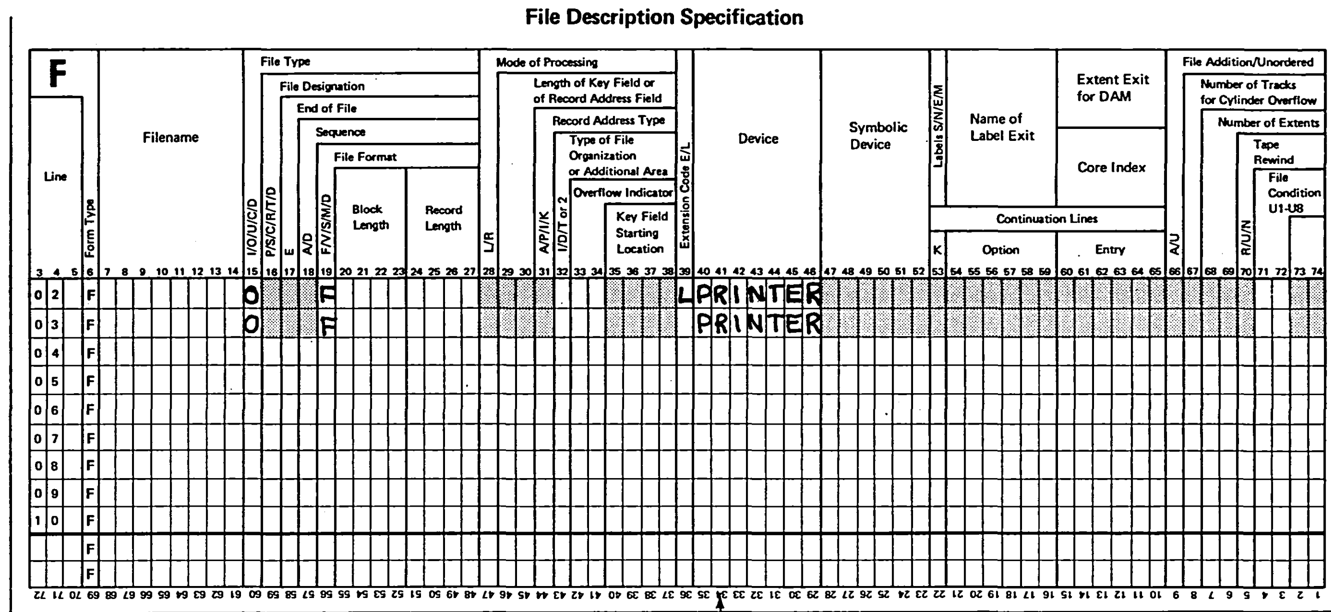
File Description Specification

Line (3–5)	Form Type (6)	Filename (7–14)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D (16)	End of File E (17)	Sequence A/D (18)	File Format F/V/S/M/D (19)	Block Length (20–23)	Record Length (24–27)	Mode of Processing L/R (28)	Length of Key Field or of Record Address Field (29–30)	Record Address Type A/P/I/K (31)	Type of File Organization or Additional Area I/D/T or 2 (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/E/M (53)	Name of Label Exit (54–59)	Extent Exit for DAM / Core Index (60–65)	A/U (66)	File Addition/Unordered (66)	Number of Tracks for Cylinder Overflow (67)	Number of Extents (68–69)	Tape Rewind R/U/N (70)	File Condition U1-U8 (71–72)	(73–74)
02	F		I	P			F										CONSOLE											
03	F		I	S			F										CONSOLE											
04	F		I	R			F									E	CONSOLE											
05	F		I	T			F									E	CONSOLE											
06	F		I	D			F										CONSOLE											
07	F																											
08	F		O				F										CONSOLE											
09	F																											
10	F		D				F										CONSOLE											
	F																											
	F																											

Figure 46. Console Files (Printer/Keyboard, Models 10 and 12)

Figure 47. Printer Files

PRINTER FILES



Device can be
PRINTER (Models 10, 12, and 15),
PRINTR2 (Models 10 and 12), or
PRINT84 (Model 15).

TAPE FILES

File Description Specification

Line (3-5)	Form Type (6)	File Type (15)	File Designation (16)	File Format (19)	Record Length (24-27)	Length of Key Field or of Record Address Field (29-30)	Record Address Type (31)	Type of File Organization or Additional Area (32)	Extension Code E/L (39)	Device (40-46)	K (53)	Option (54-59)
02	F	I	P	F						TAPE		
03	F	I	S	F						TAPE		
04	F	I	R	F	L8	Ø3	/	T	E	TAPE		
05	F	I	R	F					E	TAPE		
06	F	I	T	F					E	TAPE		
07	F	I	D	F						TAPE		
08	F	O		F						TAPE		
09	F										K	ASCII
10	F										K	BUFOFF
	F											
	F											

Note: Column 19 may be V if Column 16 is P, S, O, or D

Figure 48. Tape Files

File Description Specification

Line (3-5)	Form Type (6)	Filename (7-14)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D (16)	End of File E (17)	Sequence A/D (18)	File Format F/V/S/M/D (19)	Block Length (20-23)	Record Length (24-27)	L/R (28)	Length of Key Field or of Record Address Field (29-30)	Record Address Type A/P/I/K (31)	Type of File Organization or Additional Area I/D/T or 2 (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/E/M (53)	Name of Label Exit (54-59)	Extent Exit for DAM / Core Index (60-65)	A/U (66)	Number of Tracks for Cylinder Overflow (67-68)	Number of Extents (69)	Tape Rewind R/U/N (70)	File Condition U1-U8 (71-72)
02	F		I	P													CRT77									
03	F		I	S													CRT77									
04	F		U	P													CRT77									
05	F		U	S													CRT77									
06	F		I	D													CRT77									
07	F		U	D													CRT77									
08	F		I	T												E	CRT77									
09	F		I	R									A			E	CRT77									
10	F		O														CRT77									
	F		D														CRT77									
	F																									

Figure 49. File Description Entries for 3277 CRT/Keyboard (Model 15)

File Description Specification

Line (3-5)	Form Type (6)	Filename (7-14)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D (16)	End of File E (17)	Sequence A/D (18)	File Format F/V/S/M/D (19)	Block Length (20-23)	Record Length (24-27)	L/R (28)	Length of Key Field or of Record Address Field (29-30)	Record Address Type A/P/I/K (31)	Type of File Organization or Additional Area I/D/T or 2 (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/E/M (53)	Name of Label Exit (54-59)	Extent Exit for DAM / Core Index (60-65)	A/U (66)	Number of Tracks for Cylinder Overflow (67-68)	Number of Extents (69)	Tape Rewind R/U/N (70)	File Condition U1-U8 (71-72)
02	F		I	P			F										READ42									
03	F		I	S			F										READ42									
04	F		C	P			F										READ42									
05	F		C	S			F										READ42									
06	F		I	D			F										READ42									
07	F		C	D			F										READ42									
08	F		I	R			F									E	READ42									
09	F		I	T			F									E	READ42									
10	F		O				F										READ42									
	F																									
	F																									

Figure 50. File Description Entries for 1442 Card Read Punch

File Description Specification

Line	Form Type	Filename (7–14)	File Type (15)	File Designation (16)	File Format (19)	Extension Code E/L (39)	Device (40–46)
02	F		I	P	F		READØ1
03	F		I	S	F		READØ1
04	F		I	R	F	E	READØ1
05	F		I	D	F		READØ1
06	F		I	T	F	E	READØ1
07	F						
08	F						
09	F						
10	F						
	F						
	F						

Figure 51. File Description Entries for 2501 Card Reader (Model 15)

File Description Specification

Line	Form Type	Filename (7–14)	File Type (15)	File Designation (16)	File Format (19)	Extension Code E/L (39)	Device (40–46)
02	F		I	P	F		MFCM
03	F		I	S	F		MFCM
04	F		C	P	F		MFCM
05	F		C	S	F		MFCM
06	F		I	D	F		MFCM
07	F		C	D	F		MFCM
08	F		I	R	F	E	MFCM
09	F		I	T	F	E	MFCM
10	F		O		F		MFCM
	F						
	F						

Device is MFCM1 or MFCM2

Figure 52. File Description Entries for 2560 MFCM (Model 15)

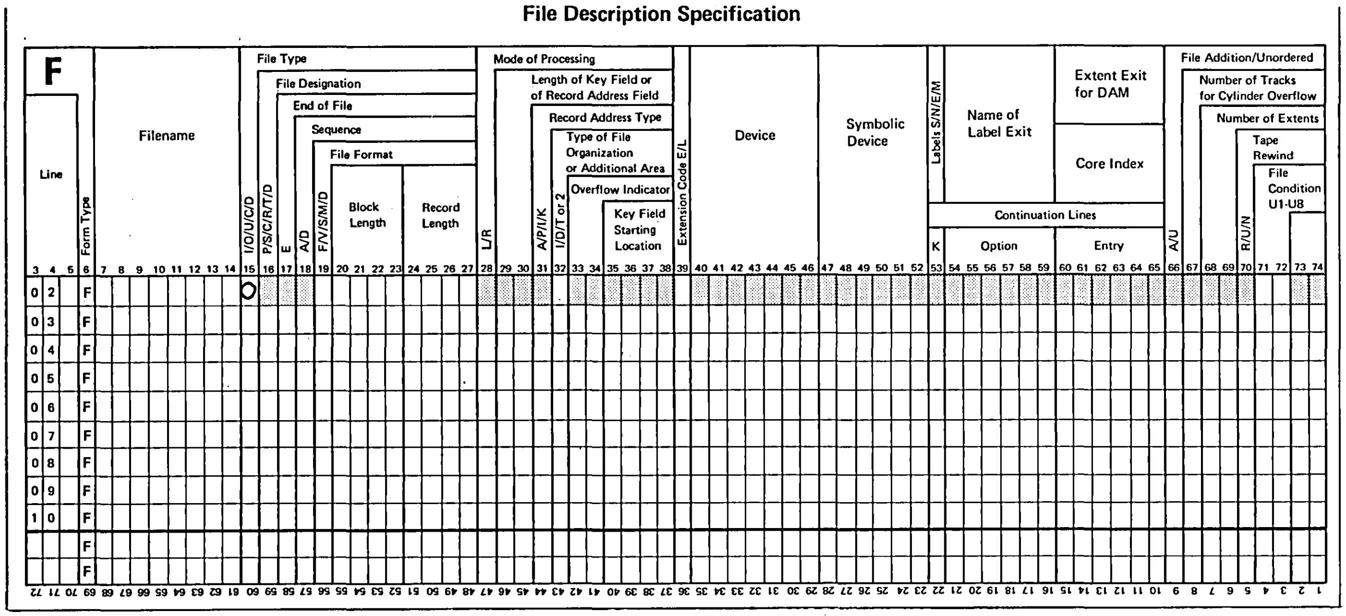


Figure 53 (Part 1 of 2). File Description Specifications for Device Independent Output File (Model 15)

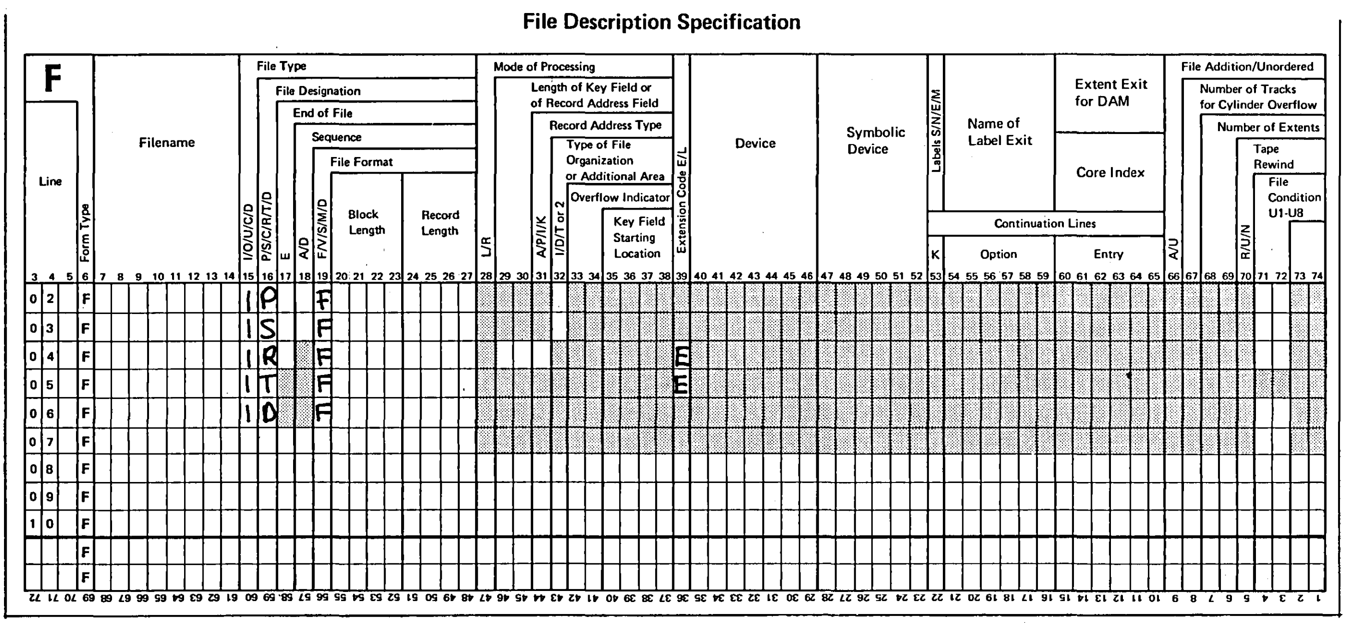


Figure 53 (Part 2 of 2). File Description Specifications for Device Independent Input File (Model 15)

File Description Specification

Line (3–5)	Form Type (6)	Filename (7–14)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D (16)	End of File E (17)	Sequence A/D (18)	File Format F/V/S/M/D (19)	Block Length (20–23)	Record Length (24–27)	Mode of Processing L/R (28)	Length of Key Field or of Record Address Field (29–30)	Record Address Type A/P/I/K (31)	Type of File Organization or Additional Area I/D/T or 2 (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/E/M (53)	Name of Label Exit (54–59)	Extent Exit for DAM / Core Index (60–65)	A/U (66)	Number of Tracks for Cylinder Overflow (67–68)	Number of Extents (69–70)	Tape Rewind R/U/N (70)	File Condition U1-U8 (71–72)	File Addition/Unordered (73–74)
02	F		I	P			F										DISKET										
03	F		I	S			F										DISKET										
04	F		I	R			F					A				E	DISKET										
05	F		I	T			F									E	DISKET										
06	F		I	D			F										DISKET										
07	F		O				F										DISKET										
08	F																										
09	F																										
10	F																										
	F																										
	F																										

Figure 54. Diskette Files

Extension specifications are needed to describe the record address files, tables, and arrays you may use in your job. Enter these specifications on the Extension and Line Counter Sheet (Figure 55).

See *Tables and Arrays* at the end of the column descriptions in this chapter for a complete description of tables and arrays including definitions of terms used in this chapter and examples of tables and arrays.

Pre-execution time tables and arrays are described in columns 11-45. Compile time tables and arrays are described in columns 19-45. If an alternating table or array is to be specified with another table or array, it is described in columns 46-57 of the same line as the first. A maximum of 63 tables and arrays can be used per program. Only 60 of these tables or arrays may be compile-time tables or arrays.

Record address files require entries on the Extension Sheet in columns 11-26.

Figure 58 is a chart showing possible Extension Sheet entries.

COLUMNS 1-2 (PAGE)

See Chapter 2.

COLUMNS 3-5 (LINE)

See Chapter 2.

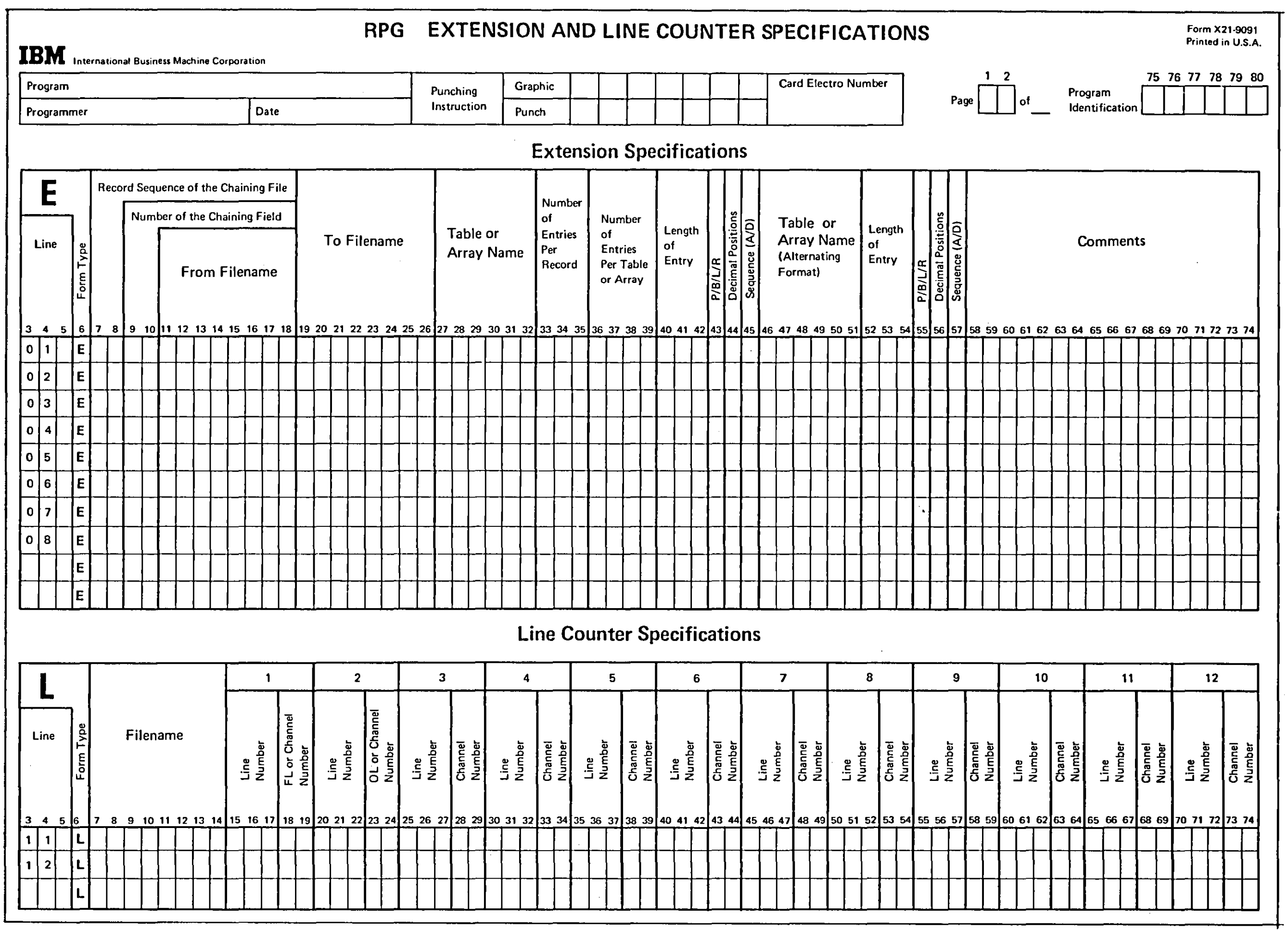


Figure 55. Extension and Line Counter Sheet

COLUMN 6 (FORM TYPE)

An E must appear in column 6.

COLUMNS 7-10

Columns 7-10 are not used.

COLUMNS 11-18 (FROM FILENAME)

Entry	*Explanation*
Record Address Filename	The name of the record address file defined on the File Description Specifications Sheet.
Table or Array Filename	Table or array file loaded at pre-execution time.
Blank	1. Table or array loaded at compilation time if an entry appears in Number of Entries per Record (columns 33-35). 2. Array loaded at execution time (loaded via input or calculations specifications) if there is no entry in Number of Entries per Record (columns 33-35).

Columns 11-18 are used to name a table file, array file, or record address file. Filenames must begin in column 11.

Leave columns 11-18 blank for compile time tables or arrays or for arrays loaded via input or calculations specifications (execution time array). These columns must contain the table or array filename of every pre-execution time table or array used in your program. More than one pre-execution time table or array can be read from the same MFCU or diskette file; therefore the From Filename might be the same for more than one table or array (this is true only for MFCU and diskette files).

COLUMNS 19-26 (TO FILENAME)

Entry	*Explanation*
Name of an input or update file	The file processed via the record address file named under From Filename.
Name of an output file	The output file on which a table or array is to be written at end of job.

Columns 19-26 define the relationship between a file named in these columns and a file named in columns 11-18. Filenames must begin in column 19.

If a record address file is named under From Filename, columns 11-18, the name of the primary or secondary file that contains the data records to be processed must be entered in To Filename, columns 19-26.

If you wish a table or array to be written or punched, use columns 19-26 to enter the filename of the output file you will use to do this. This output file must have been previously named in the file description specifications. Execution time tables/arrays cannot be written at end of job. Leave columns 19-26 blank for execution time tables/arrays or if you do not want the table or array written or punched.

If a table or array is to be written or punched, it is automatically written or punched at the end of the job after all other records have been written or punched.

Since the table or array will be written or punched in the same format in which it was entered, you may want to rearrange the output table or array through output-format specifications. You may format table or array output by using exception lines to write out one item at a time (see *Operation Codes, Exception* in Chapter 8). Tables or arrays will be written or punched under RPG II control only after all records have been processed (Last Record indicator is on).

Note: If a table or array is to be written to a printer file at the end of a job, the last Output specification should be a space or skip to the line at which table or array output should begin.

COLUMNS 27-32 (TABLE OR ARRAY NAME)

Entry	*Explanation*
Table or Array name	Name of a table or array used in the program.

Use columns 27-32 to name your table or array. No two tables or arrays may have the same name. The name can be from one to six characters long and must begin in column 27, and must be a valid RPG II name. If alternating tables or arrays are being described, this must name the table or array whose entry is first on the input record (see *Example*).

Table Name

Every table used in your program must be given a name from three to six characters long beginning with the letters TAB. Any name in these columns which does not begin with TAB is considered an array name. This table name is used throughout the program. However, different results can be obtained depending upon how the table name is used. Factor 2 on the Calculation Sheet can contain the name of a table to be searched and the result field can contain the name of another table from which an associated function is to be obtained. When the table name is used in Factor 2 or Result Field (on the Calculation Sheet) with LOKUP operation, it refers to the entire table. When the table name is used with any other operation code, it refers to the table item last selected from the table by a LOKUP operation. If the table name is used before any successful look-ups are performed, the first table item is referenced. See *Operation Codes, Lookup* in Chapter 8 for more information.

Tables are processed in the same order as they are specified on the Extension Sheet. Therefore, if you have more than one table, remember the tables are to be loaded in the same order as they appear on the sheet.

Tables cannot be used with an index (see *Tables and Arrays, Array Name and Index* in this chapter).

Array Name

Every array used in your program must be given a name from one to six characters long. An array name cannot begin with the letters TAB. This array name is used throughout the program. See *Tables and Arrays* after the column description in this chapter for complete information.

Example

Figure 56, insert A, shows two related tables (TABA and TABB) described in alternating form on a table input card. An item for TABA appears first. Thus, in insert B, TABA is named in columns 27-32 of the Extension Sheet: TABB is named in columns 46-51.

COLUMNS 33-35 (NUMBER OF ENTRIES PER RECORD)

Entry	*Explanation*
1-999	Number of table or array entries found in each table or array input record.

Indicate in columns 33-35 the exact number of table entries in each table or array input record. Every table or array input record except the last must contain the same number of entries as indicated in columns 33-35. The last record may contain fewer entries than indicated, but never more.

When two related tables are described, each table input record must contain the corresponding items from each table written in alternating form. These table items are considered as one entry (see *Example*). The number entered must end in column 35. Corresponding items from related tables must be on the same record. If there is room, comments may be entered on table input records in columns following table entries.

When loading an array the following must be considered:

1. To load a pre-execution time array, the array filename must be entered in columns 11-18 and an entry must be made in Number of Entries per Record (columns 33-35).

2. To load an array at compile time, the filename entry (columns 11-18) must be blank, but an entry must be made in Number of Entries per Record (columns 33-35).

3. To load an execution time array (via the input and/or calculations specifications), the From Filename (columns 11-18) and the To Filename (columns 19-26) entries must be blank and the Number of Entries per Record (columns 33-35) must be blank.

Example

Figure 56, insert A, shows the table items for the two related tables, TABA and TABB. The corresponding items in TABA and TABB are considered one entry. Even though there are 14 table items on the card, there are only 7 table entries. Insert B shows the Extension specifications which describe TABA and TABB as related tables.

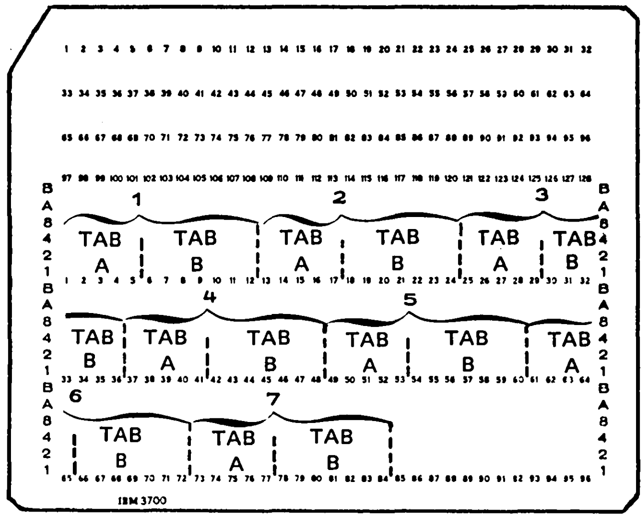
TABA (account number)	TABB* (amount due)
00126	00056.75
03240	00039.00
03648	00156.72
15632	00017.98
28887	00002.97
29821	00290.98
30001	00579.95

5 Positions — 7 Positions

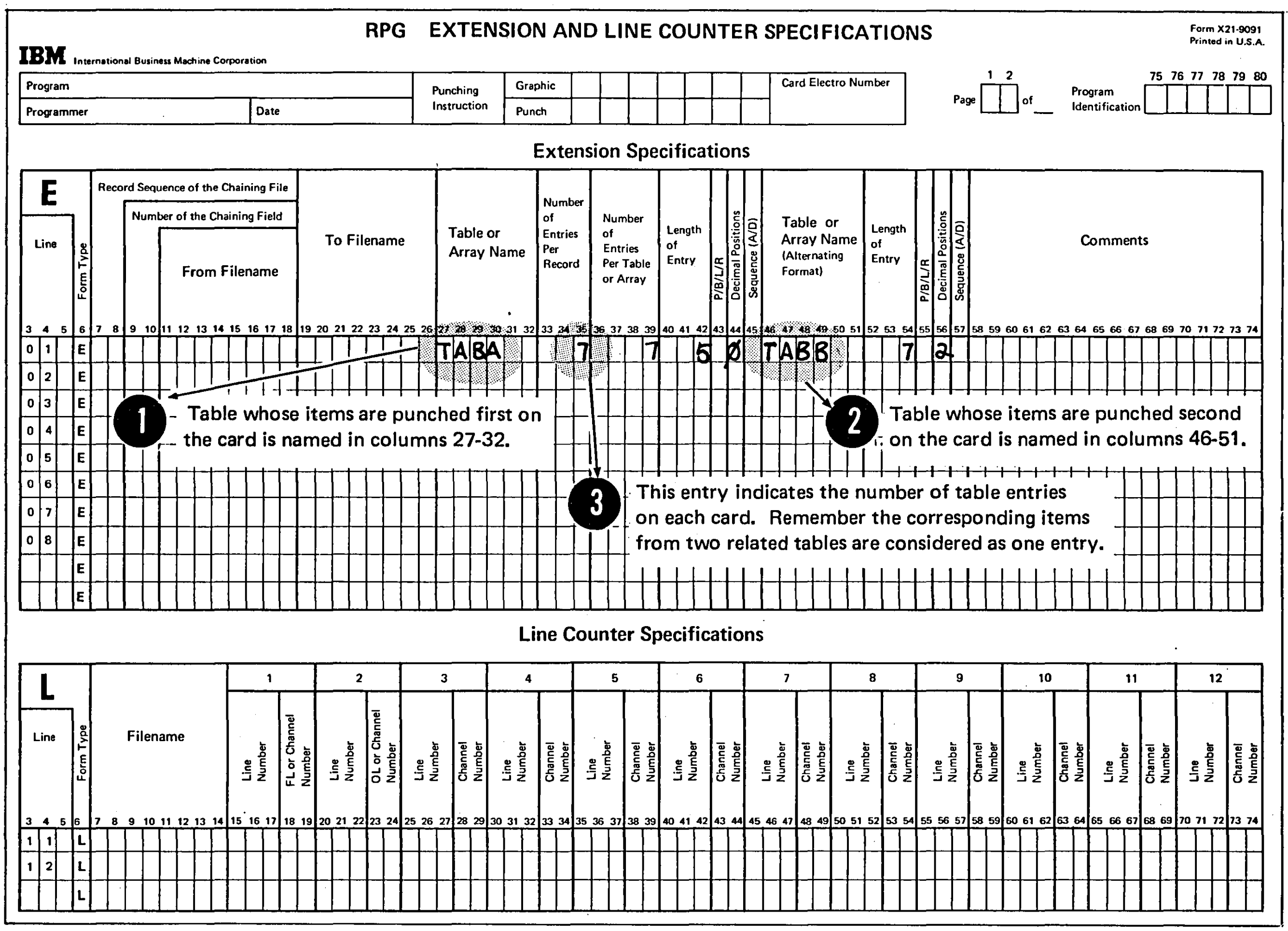
Corresponding Table Items

*Decimals in TABB are for illustration only. Decimal points are not a part of table or array input data.

Ⓐ



The corresponding items from the related tables are punched in alternating format on the table input card. The corresponding items from the two related tables are considered as one entry.



Ⓑ

Figure 56. Related Tables

COLUMNS 36-39 (NUMBER OF ENTRIES PER TABLE OR ARRAY)

Entry	*Explanation*
1-9999	Maximum number of table or array entries.

Use columns 36-39 to indicate the maximum number of table items which can be contained in the table named in columns 27-32, or the maximum number of array items which can be contained in the array named in columns 27-32. This number may apply to one table or to two alternating tables. If alternating tables are described, corresponding table items are considered one entry. Any number entered in these columns must end in column 39.

If your table or array is full, this entry gives the exact number of items in it. However, if the table or array is not full, the entry gives the number of items that can be put into it (Figure 57). A table or array that is not full is known as a short table or array.

Since the number of items for two related tables or arrays must be the same, the entry in these columns also gives the number of items in a second table or array (columns 46-51).

TABPRT (Part Number)	TABAMT* (Price)
001	127.62
002	198.32
003	000.27
004	000.01
005	001.98
009	003.79
010	005.67
014	002.33
026	014.67
045	029.33
096	029.34
097	000.05
098	000.09
099	001.19
100	002.22
101	126.73
110	596.74
115	393.75
126	697.75
137	001.92

If this data is loaded, TABPRT and TABAMT will be full (20 entries fill the table).

TABPRT (Part Number)	TABAMT* (Price)
001	127.62
002	198.32
003	000.27
004	000.01
005	001.98

If this data is loaded, TABPRT and TABAMT will not be full.

*Decimals are for illustration only.

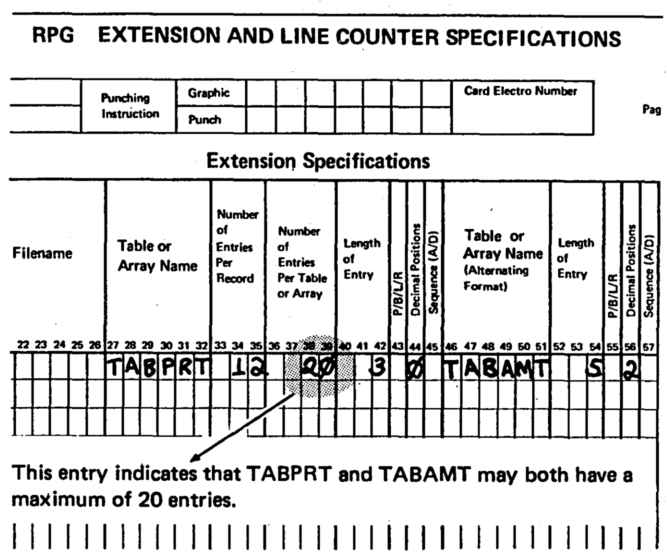


Figure 57. Table Entries (Number per Table)

COLUMNS 40-42 (LENGTH OF ENTRY)

Entry	*Explanation*
1-256	Length of a table or array entry.

Use columns 40-42 to give the length of each entry in the table or array named in columns 27-32. The number entered must end in column 42. For numeric tables or arrays in packed decimal format (see *Column 43, Packed or Binary Field*), enter the unpacked decimal length in columns 40-42. For numeric tables or arrays in binary format, enter the number of bytes required in storage for the binary field. For a 2 character binary field, the entry in columns 40-42 is 4; for a 4 character binary field the entry is 9.

All table items must have the same number of characters. It is almost impossible, however, for every item to be the same length. Therefore, add zeros or blanks to the front of numeric items to make them the same length and add blanks to alphameric items. For alphameric items, blanks may be added either before or after the item (see *Examples, Example 1*).

If two related tables or arrays are described on one Extension Sheet, the entry in columns 40-42 applies to the table whose item appears first on the record (see *Examples, Example 2*).

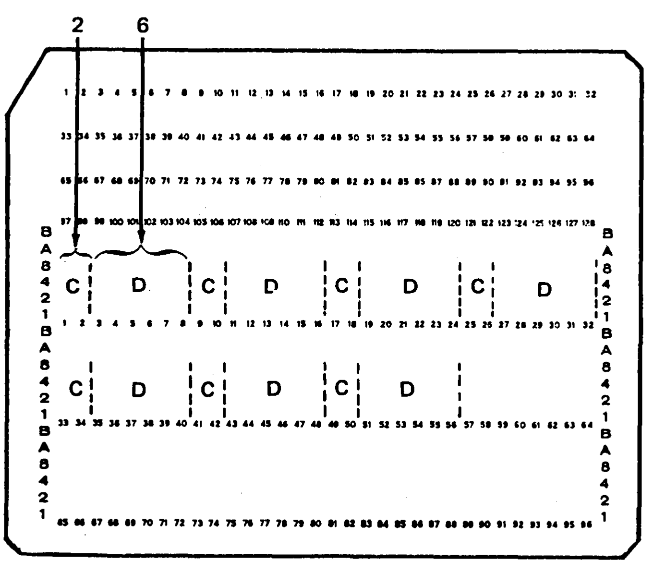
The maximum length of a numeric item is 15 characters. The maximum length of an alphameric item is 256 characters. See *Tables and Arrays* in this chapter for more information.

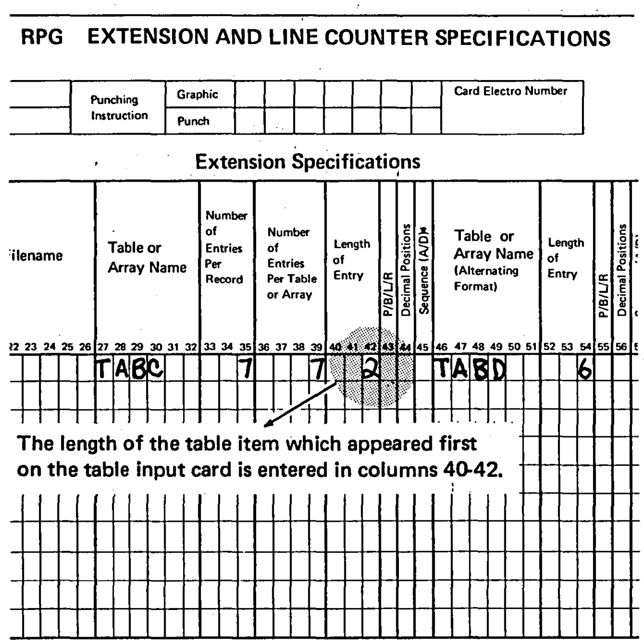
Examples

Example 1: The following table, called TABMO, lists the months of the year. The name SEPTEMBER, having nine characters, is the longest entry. Because the lengths of the entries must be the same, blanks are added to the remaining names to make each of them nine characters long.

List of Months	TABMO	
JANUARY	JANUARYbb	All entries must have the same length. Those items that are not as long as the longest item must be padded with blanks (b).
FEBRUARY	FEBRUARYb	
MARCH	MARCHbbbb	
APRIL	APRILbbbb	
MAY	MAYbbbbbb	
JUNE	JUNEbbbbb	
JULY	JULYbbbbb	
AUGUST	AUGUSTbbb	
SEPTEMBER	SEPTEMBER	
OCTOBER	OCTOBERbb	
NOVEMBER	NOVEMBERb	
DECEMBER	DECEMBERb	

Example 2: The following shows entries in a table input card for related tables, TABC and TABD. Each item in TABC is two characters long; each item in TABD is six characters long. Since TABC appears first on the card, its length (2) is specified in columns 40-42. The length of items in TABD is indicated in columns 52-54.





COLUMN 43 (PACKED OR BINARY FIELD)

Entry	*Explanation*
Blank	Data for table or array is in unpacked decimal format or is alphameric. This is used for execution time arrays (must be blank for compile-time tables or arrays).
P	Data for table or array is in packed decimal format (pre-execution time tables or arrays only).
B	Data for table or array is in binary format (pre-execution time tables or arrays only).

For a complete discussion of unpacked decimal, packed decimal, and binary data representation, see *Column 43, Packed or Binary Field* in Chapter 7, *Input Specifications.*

COLUMN 44 (DECIMAL POSITIONS)

Entry	*Explanation*
Blank	Alphameric table or array.
0-9	Number of positions to the right of the decimal in numeric table or array items.

Column 44 must always have an entry for a numeric table or array. If the items in a numeric table or array have no decimal positions, enter a 0.

If two alternating tables or arrays are described in one file, the specification in this column applies to the table containing the item which appears first on the record.

COLUMN 45 (SEQUENCE)

Entry	*Explanation*
Blank	No particular order.
A	Ascending order.
D	Descending order.

Use column 45 to describe the sequence (ascending or descending) of the data in a table or array. Execution time arrays are not checked for sequence, but column 45 must contain an entry if high or low LOKUP is to be used.

When an entry is made in column 45, the table or array is checked for the specified sequence. If a pre-execution time table or array is out of sequence, an error occurs and the program halts immediately. The program can be restarted from the point where it halted if you do not want to correct the out-of-sequence condition; however, if you do correct the out-of-sequence condition, program execution must be restarted from the beginning.

Ascending order means that the table or array items are entered starting with the lowest data item (according to the collating sequence) and proceeding to the highest. Descending order means that the table or array items are entered starting with the highest data item and proceeding to the lowest.

If alternating tables or arrays are described in one file, the entry in column 45 applies to the table or array containing the item which appears first on the record.

When you are searching a table or array for an item (LOKUP) and wish to know if the item is high or low compared with the search word, your table or array must be in either ascending or descending order. See *Operation Codes, Lookup* in Chapter 8 for more information. When a specific sequence has been specified, RPG II checks the data in the table or array to see if it really is in that sequence. In checking for sequence, an equal condition is considered valid. This allows you to pad the beginning of the table with zeros or blanks, or to pad the end of the table with 9's (assuming ascending sequence).

COLUMNS 46-57

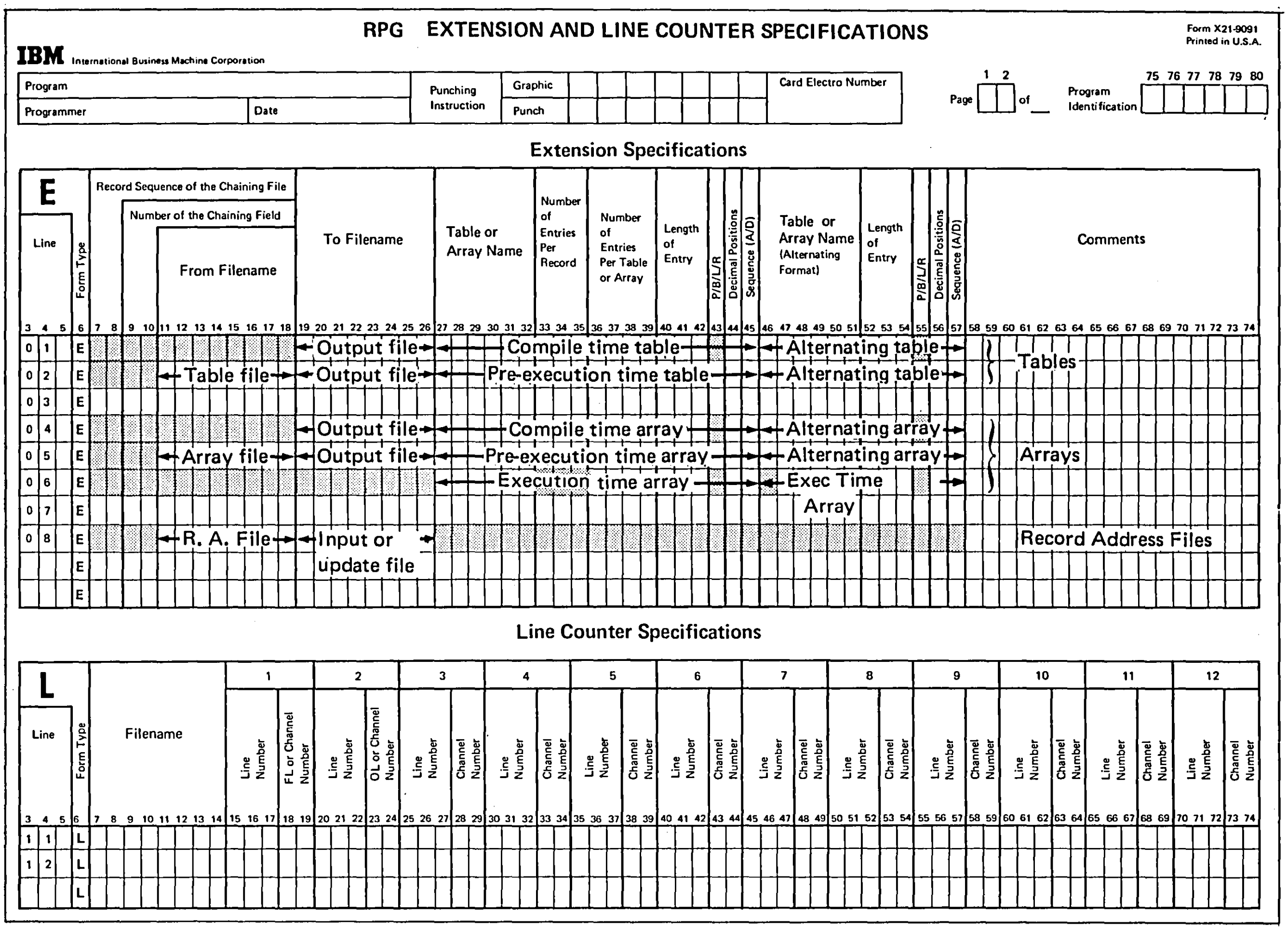
Use columns 46-57 when describing a second table or array. For compile time and pre-execution time tables and arrays, these columns are used to describe a table or array that is entered in alternating format with the table or array described in columns 27-32. For execution time arrays, these columns may be used to describe another table or array which is loaded independently of the array described in columns 27-32. All fields in this section have the same significance and require the same entries as the fields with corresponding titles in columns 27-45. See the previous discussion on those columns for information about correct specifications.

COLUMNS 58-74 (COMMENTS)

Enter any information you wish in columns 58-74. The comments you use should help you understand or remember what you are doing in each specification line. Comments are not instructions to the RPG II program; they serve only as a means of documenting your program.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

See Chapter 2.



- The shaded columns must be blank for the file named.
- For tables and arrays, columns 19-26 and columns 46-57 are optional.
- Execution arrays are loaded via input and/or calculation specifications.
- For record address files, columns 11-26 must have entries.

Figure 58. Possible File Entries for Extension Specifications

TABLES AND ARRAYS

Tables and arrays are systematic arrangements of data items having like characteristics; that is, the same field length, data type (alphameric or numeric), and number of decimal positions. Both tables and arrays are described on the Extension Specifications Sheet. See Figure 58 for possible entries. Important differences exist, however, in defining and processing tables and arrays.

Tables are used during the execution of a program much like a shipping clerk would use a rate table for obtaining freight rates. The clerk might scan the table for the desired city, then select the corresponding rate. Tables are referenced by searching the table one item at a time for a specific item of data with a unique identifier. Table names must begin with the letters TAB.

Arrays can also be searched for a uniquely identified data item. Unlike tables, however, array items can also be referenced by their relative position to other items. This is done by indexing to a specific item in the array. Also, an entire array can be processed sequentially by using the array name only once in certain calculation operations. Array names must not begin with the letters TAB.

Several terms are used to describe tables and arrays:

- *Compile time* tables and arrays are compiled with the source program and become a permanent part of the object program. A compile time table or array can be permanently changed only by recompiling the source program with the revised table or array.

- *Pre-execution time* tables and arrays are loaded with the object program before actual execution of the RPG II program begins; that is, before any input files are read, calculations performed, or output functions performed.

- *Execution time* arrays are loaded or created by input or calculation specifications. They are loaded after actual execution of your RPG II program has begun (read in as input data or created during calculations in your program). An execution time array is also described on the Extension Specifications Sheet.

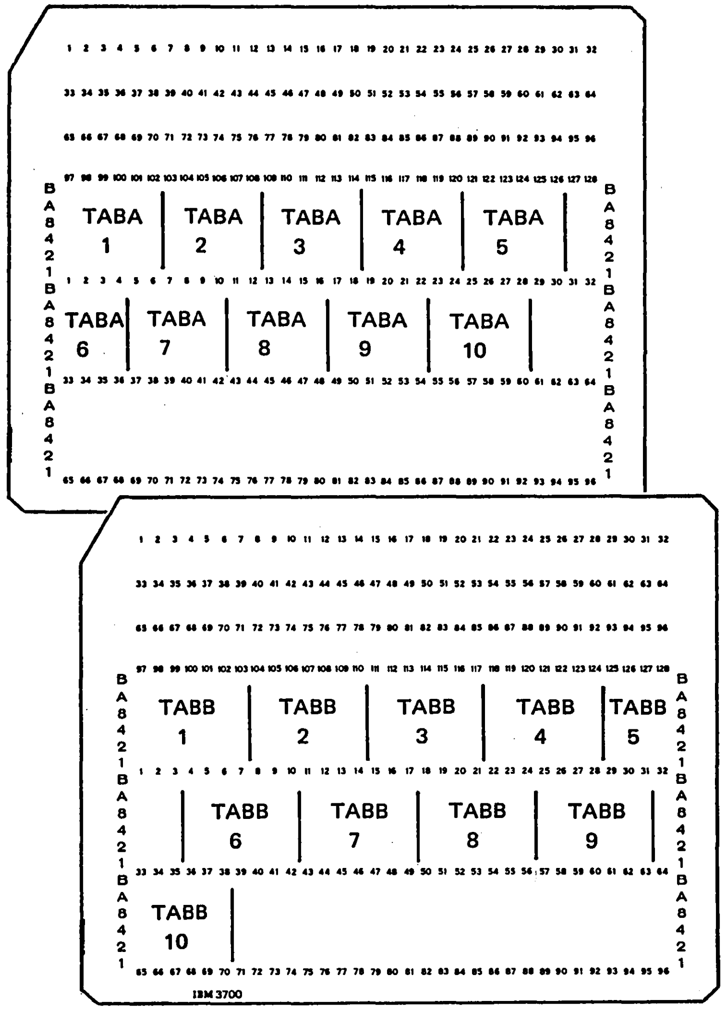
- *Related* tables and arrays are tables and arrays that are used together. The items in each table or array are called corresponding items; each item in the second gives additional information about its corresponding item in the first. In Figure 59, TABA and TABB are related. An item in TABA gives a part number, the corresponding item in TABB gives the part cost. Although all items within one table or array must have the same characteristics, corresponding items of related tables or arrays may have different characteristics. Related tables and arrays do not have to have the same number of entries unless they are described in the same extension specification.

- *Short* tables and arrays are those in which not all of the entries contain data. The unused parts of numeric tables and arrays are filled with zeros; the unused parts of alphameric tables and arrays are filled with blanks. You usually create short tables or arrays when you have only a few table or array items available when building the table, but know that more items will soon be included. Short tables and arrays must have at least one entry.

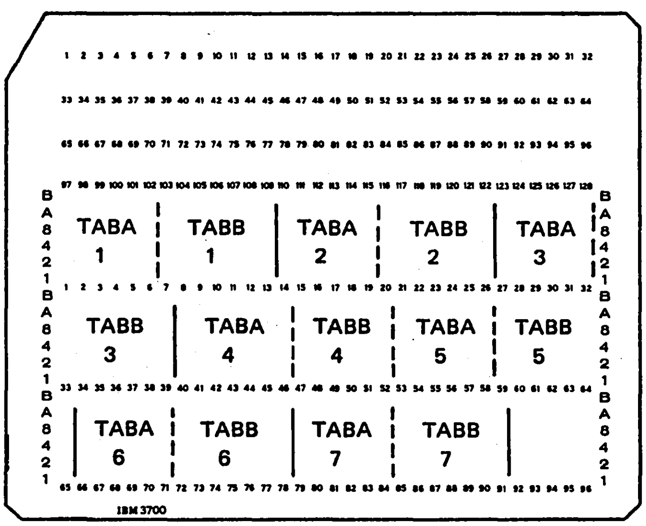
- *Full* tables and arrays are those in which all possible entries contain data.

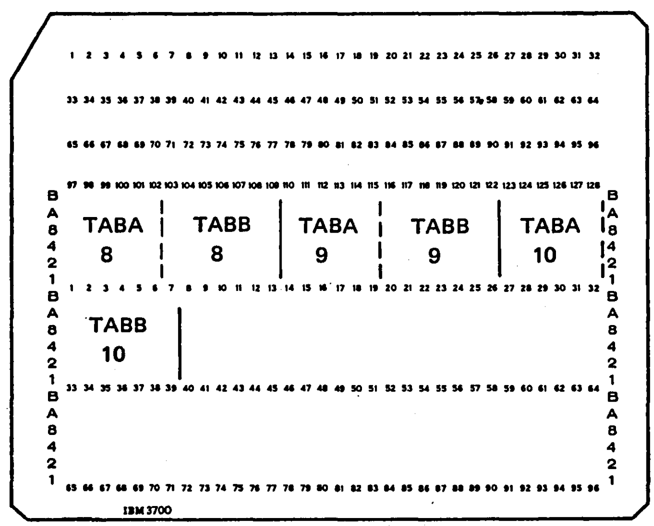
TABA	TABB
345126	000373
38A473	000498
39K146	001297
40B125	000093
41C043	041998
42D893	000087
43K532	000349
44M111	0C0679
45P673	000898
46C732	147587

(A) Related tables



(B) TABA and TABB described as separate tables.





(C) TABA and TABB described in alternating format.

Figure 59 Related Tables (TABA and TABB) Described Separately and Alternately

Creating Table or Array Input Records

When creating compile-time or pre-execution time tables or arrays, the table and array data must be recorded according to certain rules. In the following list of rules, the term *entry* refers to one element in a single table or array, or to corresponding items of related tables or arrays.

Rules

1. The first table or array entry for each record must begin in position 1.

2. An entire record need not be filled with entries. In this case, blanks or comments can be included after the entries. (Figures 60 and 61 show a table input record and extension specifications for alternating tables. Note that three blanks appear between the last table entry and the comment.)

3. Each record, except the last, must have the same number of entries. You may want to place just one entry on each record or as many entries as the record can hold.

4. An entire entry must be on one record. It cannot be split. Thus the length of a single entry is limited to the maximum record length for the device. If related tables or arrays are used, corresponding items must be on the same record and, together, cannot exceed maximum record length for the device.

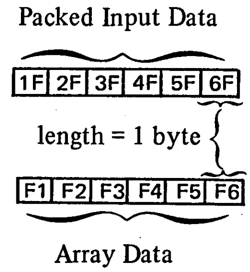
5. Related tables or arrays can be described separately or in alternating format. Alternating format means that the corresponding items are considered one table or array entry. Figure 59 shows ways in which related tables or arrays can be described.

6. The number of table and/or array names used in a program must be no more than 60.

Packed Data To/From Table/Array

The following examples show results of loading arrays from packed data fields. This applies to pre-execution time tables and arrays only.

With one-byte array elements:



With three-byte array elements:

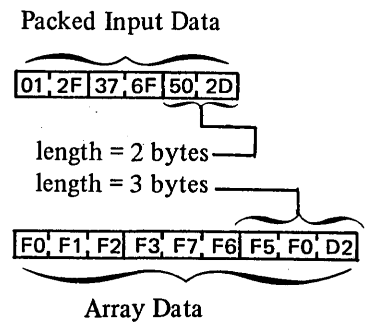


Figure 61 shows an example of packed data in a pre-execution time array. The from file must be a device that supports packed data. The packed fields must be four bytes long.

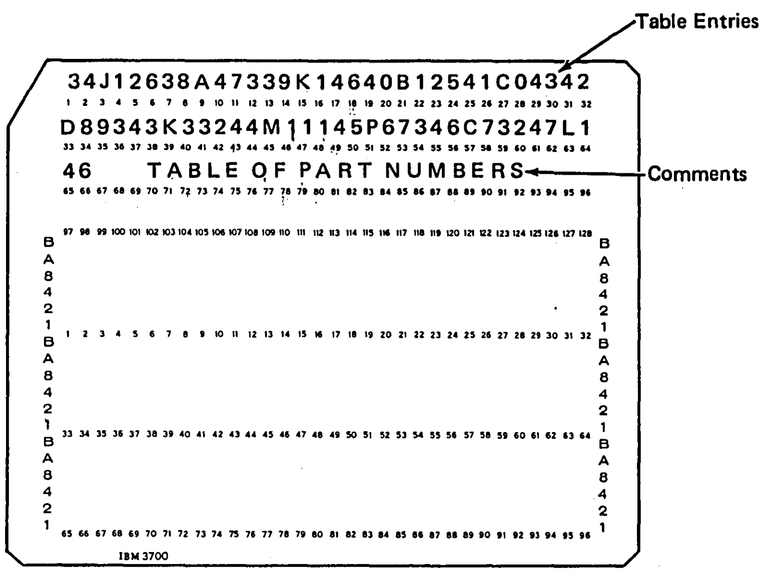


Figure 60. Input Record for Alternating Tables, TABPAR and TABID

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

IBM International Business Machine Corporation

Form X21-9091
Printed in U.S.A.

Program		Punching Instruction	Graphic	Card Electro Number	Page 1 2 of	Program Identification 75 76 77 78 79 80
Programmer	Date		Punch			

Extension Specifications

Line (3-5)	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 or Array Name (27-32)	Number of Entries Per Record (33-35)	Number of Entries Per Table or Array (36-39)	Length of Entry (40-42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46-51)	Length of Entry (52-54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58-74)
01	E			PARTNO		TABPAR	11	11	2			A	TABID	4				
02	E			PRICES		PRI	10	10	7	P	2							
03	E																	

Figure 61. Extension Specifications for Alternating Tables, TABPAR and TABID

Defining Tables and Arrays

All tables and arrays are described on the Extension Sheet. One line is used to describe one set of table or array input records. If only one table or array is described, columns 11-45 are used. If alternating tables or arrays are described on one set of input records, columns 46-57 are used to describe the second table or array. If pre-execution time tables and array are being described, entries in columns 11-18 and 27-45 are required, as described in the first part of this chapter. Columns 19-26 are used if the table or array is to be written or punched at the end of the job.

Tables and arrays can be specified in any sequence. Compile time and pre-execution time tables and arrays can be mixed. Remember the sequence in which tables and arrays are specified on the Extension Sheet determines the order in which they must be loaded at the start of the job (see *Loading Tables and Arrays*).

Figure 62 shows the necessary extension specifications for each type of array. Line 1 specifies a compile time array, ARRAYC. This array has a total of eight elements (three elements per record). Each element has a length of 12 positions, including four decimal places. Line 2 specifies pre-execution time array, ARRAYE, to be read from file CARDINP. ARRAYE has 250 alphameric elements (12 elements per record); each element is 5 positions long and is equal to or higher than the previous element in collating sequence. Line 3 specifies an execution time array, ARRAYI, to be read from input records. ARRAYI has ten numeric elements each ten positions long.

Compile time and pre-execution time arrays (lines 1 and 2) can include entries in columns 19-26 (To Filename) and in columns 46-57 (to describe an alternating array). Execution time arrays cannot have To Filename and alternating array specifications.

Loading Tables and Arrays

Tables and arrays can be loaded at compilation time or pre-execution time. When loaded at compilation or pre-execution time, the entire table or array is loaded. Arrays can also be loaded at execution time.

IBM International Business Machine Corporation

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

Form X21-9091
Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Extension Specifications

Line (3-5)	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 or Array Name (27-32)	Number of Entries Per Record (33-35)	Number of Entries Per Table or Array (36-39)	Length of Entry (40-42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46-51)	Length of Entry (52-54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58-74)
01	E					ARRAYC	3	8	12		4							COMPILE
02	E			CARDINP		ARRAYE	12	250	5			A						PRE-EXECUTION
03	E					ARRAYI		10	10		0							EXECUTION
04	E																	

Figure 62. Specifications for Three Types of Arrays

Compilation Time

Tables and arrays loaded at compilation time are compiled along with the RPG II source program. They become a part of that program. Rules for loading tables and arrays at compile time are as follows:

1. The table or array records must follow the RPG II source program.

2. A record with **ƀ (blank) in positions 1-3 must appear before each table or array entered. (Any record with these characters in positions 1-3 will be treated as a delimiter, so do not use these characters as the first three characters on a data record.)

3. /* record must appear at the end of the last compile-time table or array (followed by a // CEND statement if compiling from the source library).

4. The tables and arrays must be loaded in the same order as described on the Extension Sheet.

5. A compilation time array must have entries in columns 33-35 of the Extension Sheet and must not have entries in columns 11-18 of the Extension Sheet.

6. The tables and arrays must not be packed or binary.

Figure 63 shows the placement of compile time tables and arrays in relation to RPG II source specifications.

Pre-execution Time

Pre-execution time tables and arrays are not part of your source program. They are used by the object program like any other data file.

Rules for loading tables and arrays at pre-execution time are as follows:

1. The table or array must be loaded before any other processing is done.

2. A /* record must follow every pre-execution time table or array.

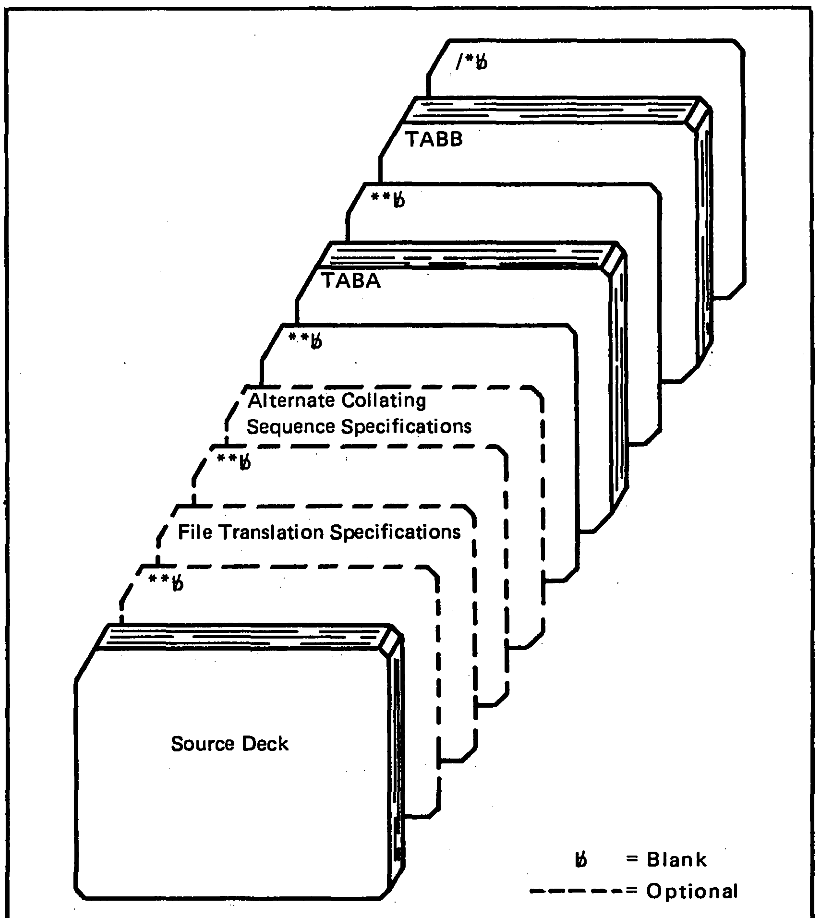


Figure 63. Placement of Compile-Time Tables in Relation to RPG II Source Specifications

3. If two or more tables or arrays are loaded, they must be loaded in the same order as described on the Extension Sheet.

4. If errors are encountered during loading, additional information about the error will be displayed on the printer-keyboard if it has been defined as the log device.

5. A pre-execution time array must have entries in columns 11-18 and 33-35, and may have entries in 43 and 55 if appropriate.

Execution Time

If you are loading an array from information in input records (execution time array), you must describe that information in your input specifications. How the entries are made depends on whether the array information is contained in one or more than one record. Any type of array (compile time, pre-execution time, execution time) can be described on the input specifications.

Execution time arrays are not checked for sequence, but column 45 (sequence) must contain an entry if high or low LOKUP is used.

If an execution time array is to be read in packed or binary format, an entry should be given in column 43 of the Input Sheet. In this case, the From and To columns on the Input Sheet should define the positions the array occupies in the record in the packed or binary format. The unpacked decimal length of each array element is defined on the Extension Sheet. An execution time array must not have an entry in columns 11-26, 33-35, 43, and 46-57 on the Extension Sheet.

When reading packed information into an array from input records, each element of information is read as if it were an individual field.

Array Information in One Record

If all of the array information is in one record, it can occupy consecutive positions in the record or be scattered throughout the record.

If the array elements are consecutive on the input record, they may be loaded with a single input specification. Figure 64 shows an array, INPARR, of six elements (twelve positions each) being loaded from a single record from the file ARRFILE.

If the array elements are scattered throughout the record, they may be defined and loaded one at a time, one to a

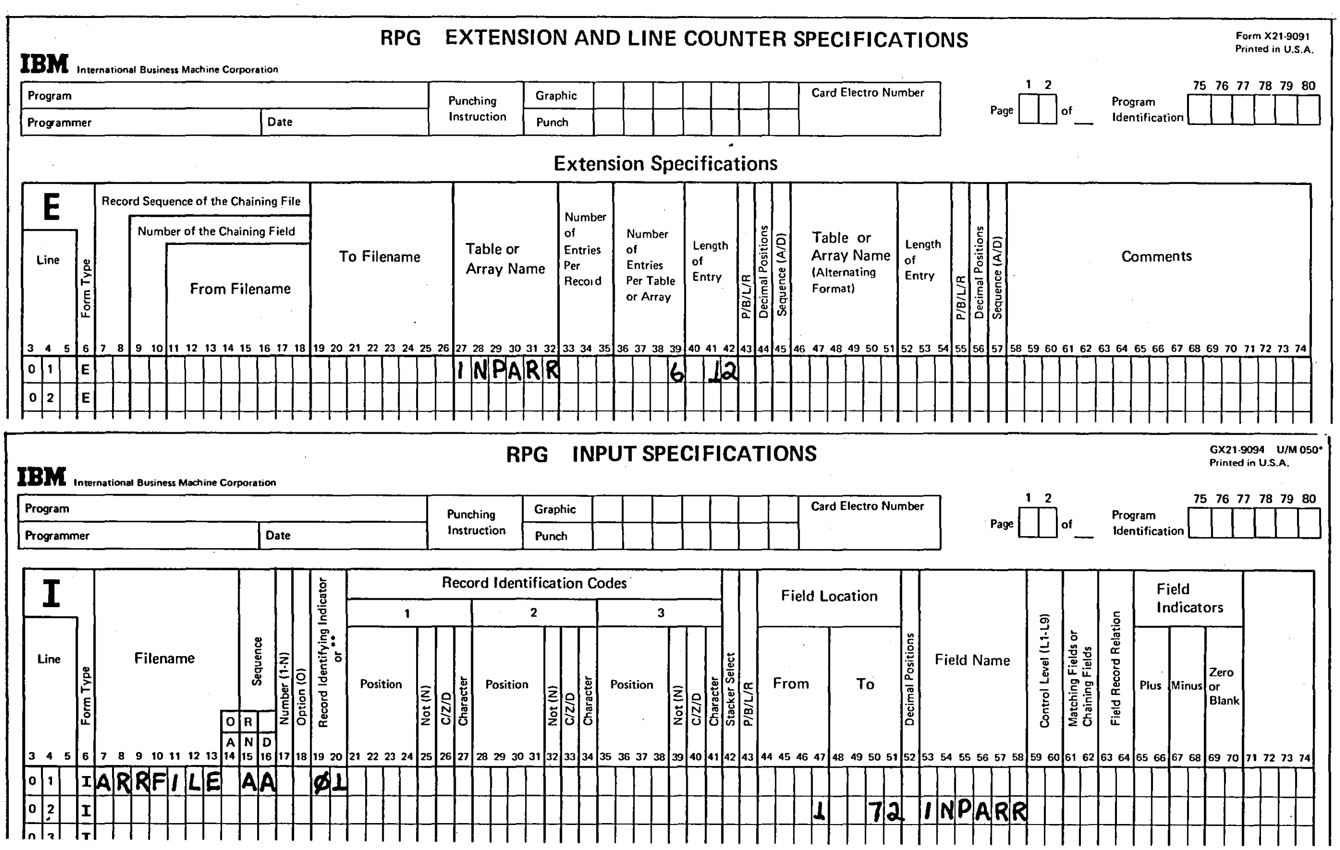
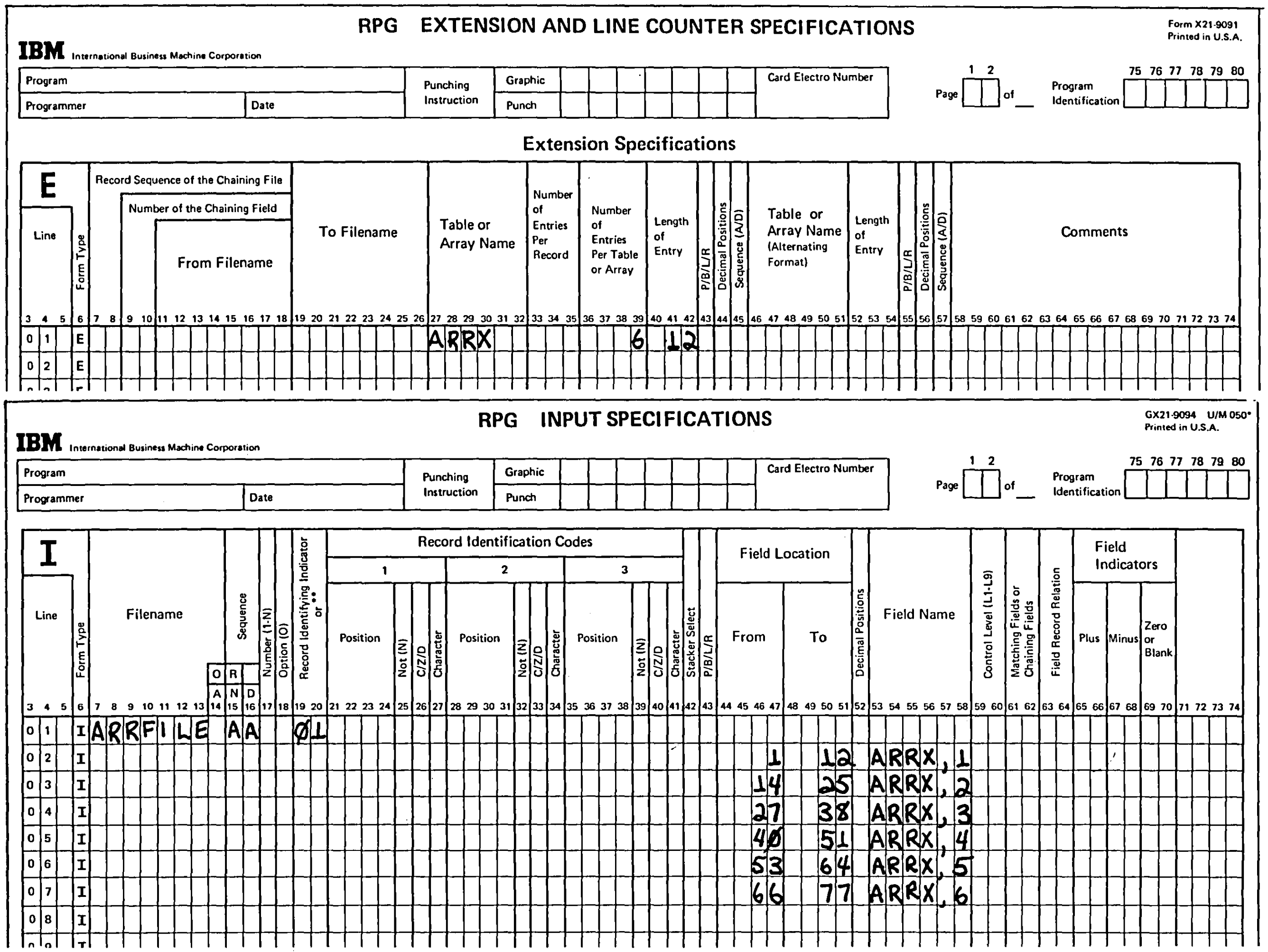


Figure 64. Defining an Execution Time Array with Consecutive Elements



RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

IBM International Business Machine Corporation

Form X21-9091 Printed in U.S.A.

Extension Specifications

Line	Form Type	Record Sequence of the Chaining File / Number of the Chaining Field / From Filename (11-18)	To Filename (19-26)	Table or Array Name (27-32)	Number of Entries Per Record (33-35)	Number of Entries Per Table or Array (36-39)	Length of Entry (40-42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46-51)	Length of Entry (52-54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58-74)
01	E			ARRX		6	12									
02	E															

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9094 U/M 050* Printed in U.S.A.

Line	Form Type	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes (21-41)	Stacker Select (42)	P/B/L/R (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 / Minus / Zero or Blank (65-70)
01	I	ARRFILE	AA			Ø1											
02	I									1	12		ARRX,1				
03	I									14	25		ARRX,2				
04	I									27	38		ARRX,3				
05	I									4Ø	51		ARRX,4				
06	I									53	64		ARRX,5				
07	I									66	77		ARRX,6				
08	I																

Figure 65. Defining an Execution Time Array with Scattered Elements

specification line. In Figure 65, an array, ARRX, of six elements with 12 positions each, is loaded from a single record from file ARRFILE; a blank column appears between each two elements.

Following are the input specifications required for loading an array from a single input record:

Column	*Entry*
6	I
7-42	Blank
43	P (packed), B (binary), or blank.
44-47 and 48-51	Field location of either an entire array (consecutive elements) or individual field locations of single elements of the array.
52	This column must be left blank.
53-58	The name of the array or the name of a single element (array name with index). This array name must be the same name as that used on the Extension Sheet.
59-62	Blank
63-64	Field record relation indicator. See *Columns 63-64* in Chapter 7 for information on this entry.
65-74	Blank

Array Information in More Than One Record

If the array information is in two or more records, there are many methods that may be used to introduce the array to the system. The method you use is primarily based on the size of the array and whether the array information is all together in the input records. Figure 66 shows the array that could result by loading array information from certain input records. Each record identified by a 1 or 3 in column 1 contains twelve items of array information. Records identified by a 2 in column 1 do not contain array information, although they appear in the same input file. Examples of loading and storing array information are found in *Examples of Using Arrays* in this chapter.

Keep in mind that the RPG II program processes one record at a time. You cannot process the entire array until all of the records containing the array information have been read and the information moved into the array fields. It may, therefore, be necessary to suppress calculation and output operations until the entire array has been read into the system.

Searching Tables and Arrays

Tables and arrays can be searched using the LOKUP operation code. LOKUP is described under *Operation Codes* at the end of the column descriptions in Chapter 8.

Using Arrays

Arrays can be used in input, output, or calculation specifications (see *Examples*). The elements in an array can be referenced individually, or the array can be referenced as a whole. Individual elements are referenced by an array name plus an index. The array name alone references the entire array.

Array Name and Index

The array name must begin in column 27 or column 46 of the Extension Sheet and must be a valid RPG II name.

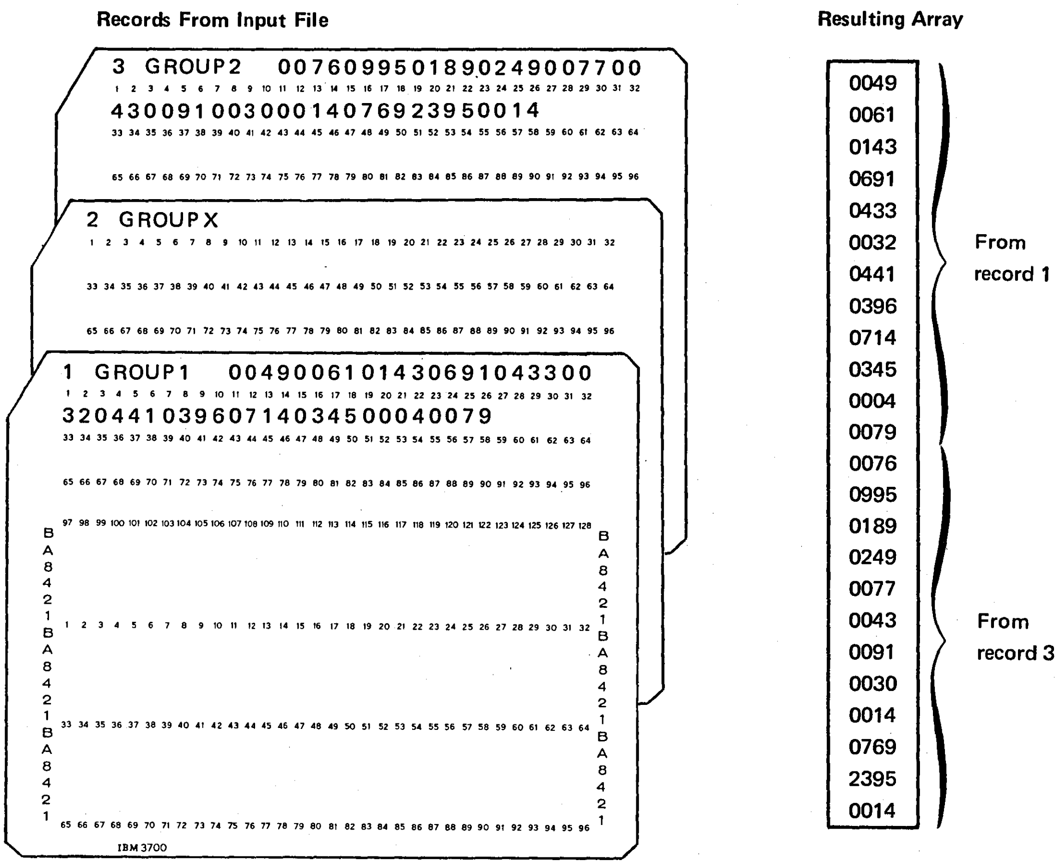


Figure 66. Loading an Array from Input Records

The length of the array name depends on how the array is being used. The array name can be from one to six characters long. The array name by itself is used only when referencing the entire array.

If individual elements of the array are to be referenced, the array name will require an index. An index may be a numeric field with zero decimal positions or a literal. The array name and index must be separated by a comma. The array name with comma and index entry is limited to six positions (input, output specifications, or Result Field of calculation specifications) or ten positions (Factor 1 or Factor 2 of calculation specifications). The index must not be zero, negative, or greater than the number of elements in the array.

Some examples of array names with and without indexes are as follows:

Valid	*Explanation*
ARAY01	
B	
AR,1	The first element of array AR.
X, YY2	Where YY2 is the name of a numeric field with zero decimal positions.
Invalid	
BALANCE	Array name has more than six characters.
6TOTAL	First character not alphabetic.
TOTAL-	Name contains special character.
CR TOT	Name contains blank.
A1, A1	Array is used as index.
BAL,XX1	Name including comma has more than six characters. This name is valid for Factor 1 and Factor 2 of the calculation specifications only.

Referencing an Array in Calculations

You can reference an entire array or individual elements in an array using calculation specifications. Process individual elements like normal fields. Remember, if an array field is to be used as a result field, the array name with comma and index cannot exceed six characters.

To reference an entire array use the array name without an index. The following operations may be used with an array name: ADD, Z-ADD, SUB, Z-SUB, MULT, DIV, SQRT, MOVE, MOVEL, MLLZO, MLHZO, MHLZO, MHHZO, MOVEA, DEBUG, XFOOT, and LOKUP. Factor 1 and Factor 2 cannot be an array name unless the Result Field is also an array name. For XFOOT and LOKUP operations, Factor 2 can be an array name without an index.

There are also several operations that can be used with an array element only (not the array name alone). These operations are: COMP, DSPLY, TESTZ, TESTB, BITON, BITOF, and MVR.

The following rules apply when using array names without an index in calculations:

1. When the factors and the Result Field all are arrays with the same number of elements, the operation is performed using the first element from every array, then the second element from every array, etc., until all elements in the arrays are processed. If the arrays do not have the same number of the entires, the operation ends when the last element of the array with the fewest elements has been processed.

2. When one of the factors is a field or constant and the other is an array, and the result field is an array, the operation is performed once for every element in the shorter array. The same field or constant is used in all of the operations.

3. If an operation code uses Factor 2 only (such as Z-ADD, A-SUB, or SQRT) and the Result Field is an array, the operation is performed once for every element in the array. The same field or constant is used in all of the operations. An exception is the MOVEA operation which moves the field into the array without regard to elements.

4. Resulting indicators (columns 54-59) cannot be used due to multiple operations being performed. Exceptions are XFOOT and LOKUP which allow resulting indicators.

Modifying the Contents of Tables and Arrays

Tables and arrays can be temporarily changed during execution of a job. This is done when the table or array name is used as a result field in an arithmetic or move operation. This causes the appropriate entry in the table or array to be modified for the duration of the job. The next time the job is executed, however, the table or array will have the original entries. Temporary changes can be permanent if the modified table or array entries are written or punched out and the new records, instead of the original ones, are used in the table or array input file or the original data is modified.

Figure 67 shows specifications for modifying the contents of corresponding tables TABFIL and TABLIT.

Adding Entries to a Short Table or Array

Entries can be added to short tables and arrays before or during execution of the job. The simplest way to add entries to a table or array is to write additional entries on the input records before program execution. However, entries can also be added during execution of a program. The entries added can be created by calculation operations or read from an input record.

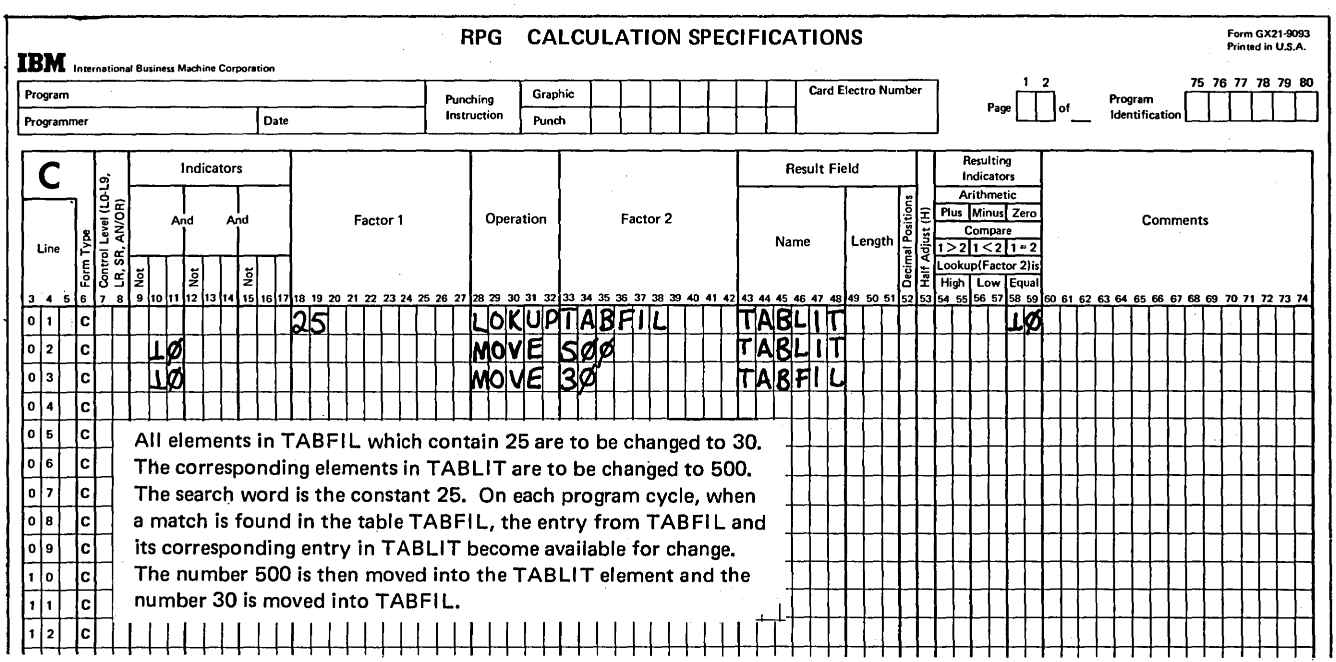
Figure 68 shows how entries are added to two related, numeric tables.

Table and Array Output

Tables and arrays can be written out one of two ways depending on whether or not you want to modify the table or array output. If you specify the name of the output file to be used in columns 19-26 of the Extension Sheet, the RPG II program will write out the entire table or array with all of its modifications. Using this method the RPG II program will write out all types of tables and arrays except execution time arrays.

If you wish to modify the output of a table or array, you must describe the table or array on the Output Sheet along with any normal fields for the output record. You must also specify the name of the table or array in columns 32-37 of the Output Sheet. Columns 40-43 must contain the record position where the last field of the table or array is to end.

If an output record is to contain only certain fields from a table or array, describe the fields in the same way as you do normal fields, using either a table name or an array name with an index.



RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation

Form GX21-9093
Printed in U.S.A.

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Resulting Indicators (Equal)
01	C		25	LOKUP	TABFIL	TABLIT	10
02	C	10		MOVE	500	TABLIT	
03	C	10		MOVE	30	TABFIL	

All elements in TABFIL which contain 25 are to be changed to 30. The corresponding elements in TABLIT are to be changed to 500. The search word is the constant 25. On each program cycle, when a match is found in the table TABFIL, the entry from TABFIL and its corresponding entry in TABLIT become available for change. The number 500 is then moved into the TABLIT element and the number 30 is moved into TABFIL.

Figure 67. Changing Table Data During Calculations

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust	Resulting Indicators High	Low	Equal	Comments
01	C		01	000	LOKUP	TABA	TABB						35	
02	C		35 01		MOVE	NEWA	TABA							
03	C		35 01		MOVE	NEWB	TABB							

The LOKUP operation is conditioned by indicator 01. Indicator 01 is on when a record is read containing information in the fields NEWA and NEWB. These fields are to be added to the short tables TABA and TABB respectively. To get the entry in the correct place in the table, a search is made to find the first empty entry. Unfilled entries are filled with zeros. Thus the search word used is 000. When the first 000 entry is found (indicator 35 turns on), NEWA and NEWB become part of the related tables TABA and TABB. These entries are temporary unless they are written (or punched) on table records.

Figure 68. Adding Table Entries to a Short Table

Editing Entire Arrays

When editing an entire array, any editing you specify applies equally to all fields in the array. If you require different editing for various elements, reference them individually.

When you specify an edit code for an entire array (column 38), note that two blanks are automatically inserted to the left of every field in the array. When you specify an edit word instead, the blanks are not inserted. The edit word must specify all the blanks you want inserted.

Example of Using Tables

A payroll job requires two related tables (Figure 69). TABNUM is the search table containing employee numbers. TABRAT is the related table containing employee salary rates. After an employee's rate has been found, the rate is multiplied by the number of hours worked. The result is the amount earned.

TABNUM	TABRAT
12345	407
12346	593
12347	369
12348	1379

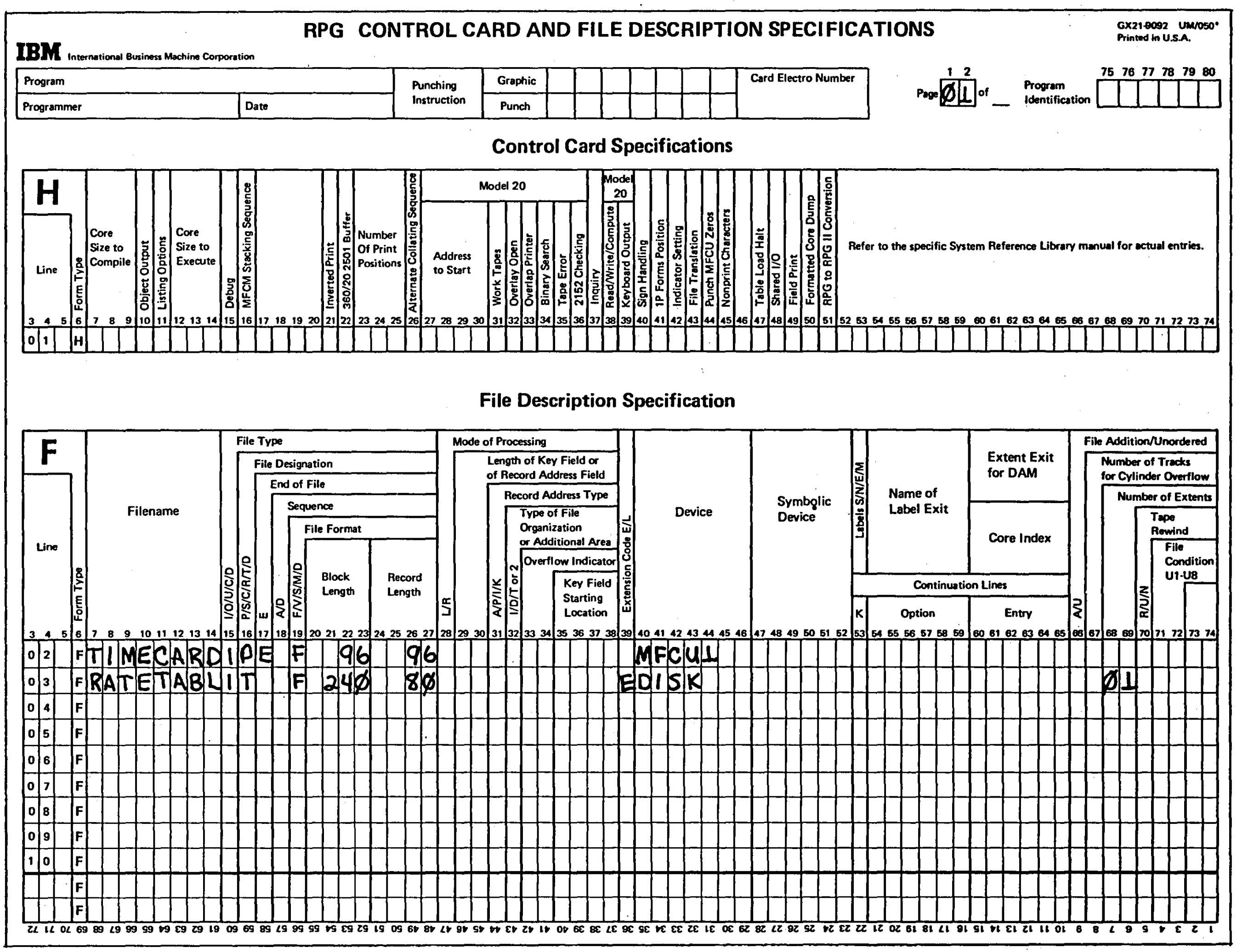
Figure 69. Tables Used in Payroll Job

The table entries are organized in alternating format on the input records. On line 01 of the Extension Sheet (Figure 70), the table searched is called TABNUM. There are eight entries in each input record and 500 entries in the table. Each table entry is five positions long and contains no decimal positions. The table is in ascending sequence. The related table is called TABRAT. Each entry is four positions long and contains two decimal positions.

Line 01 of the Calculation Sheet causes the employee number (EMPLNO) to be used as the search word for the data contained in TABNUM (the search table). Indicator 03 is turned on when the program finds an entry in TABNUM that is equal to the search word.

Line 02 of the Calculation Sheet is performed when indicator 03 is on. The rate for the employee, taken from the related table TABRAT, is multiplied by the number of hours worked (HRSWKD). The result is stored in the field EARNS, which is five positions long with two decimal positions. The result is half-adjusted.

When the search word does not find an equal entry in TABNUM (indicator 03 is not on), line 03 is performed. The literal 000.00 is then moved to the field EARNS, indicating that the employee does not have an entry in the table.



RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 UM/050* Printed in U.S.A.

IBM International Business Machine Corporation

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page 01 of __ | Program Identification

Control Card Specifications

Line	Form Type
01	H

File Description Specification

Line	Form Type	Filename (7–14)	File Type (15)	File Designation (16)	End of File (17)	File Format (19)	Block Length (20–23)	Record Length (24–27)	Extension Code (39)	Device (40–46)	Number of Extents (68–69)
02	F	TIMECARD	I	P	E	F	96	96		MFCU1	
03	F	RATETABL	I	T		F	240	80	E	DISK	01
04	F										
05	F										
06	F										
07	F										
08	F										
09	F										
10	F										

Figure 70 (Part 1 of 2). Specifications for Payroll Job

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

IBM International Business Machine Corporation

Form X21-9091 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 02 of __ | Program Identification (75 76 77 78 79 80)

Extension Specifications

Line (3-5)	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 or Array Name (27-32)	Number of Entries Per Record (33-35)	Number of Entries Per Table or Array (36-39)	Length of Entry (40-42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46-51)	Length of Entry (52-54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58-74)
01	E			RATETABL		TABNUM	8	500	5		0	A	TABRAT	4		2		
02	E																	
03	E																	

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 03 of __ | Program Identification (75 76 77 78 79 80)

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2: Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	3: Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	P/B/L/R (43)	Field Location From (44-47)	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)	(71-74)
01	I	TIMECARD	AA			01																									
02	I																				1	5	0	EMPLNO							
03	I																				42	44	1	HRSWKD							
04	I																														

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation

Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 04 of __ | Program Identification (75 76 77 78 79 80)

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR, AN/OR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Plus / 1>2 / High (54-55)	Minus / 1<2 / Low (56-57)	Zero / 1=2 / Equal (58-59)	Comments (60-74)
01	C			EMPLNO	LOKUP	TABNUM	TABRAT						03	
02	C		03	TABRAT	MULT	HRSWKD	EARNS	5	2	H				
03	C		N03		MOVE	000.00	EARNS							
04	C													
05	C													

Figure 70 (Part 2 of 2). Specifications for Payroll Job

Examples of Using Arrays

Example 1: Figure 71 illustrates a method of loading an array using fields in input records as indexes. The example shows a 12-element array with element length five. The array can be made larger without additional input specifications by assigning different values to the I1-I10 fields on each input record type 03 and to the I1 and I2 fields on each 04 record type. Succeeding type-03 records then load ten additional elements into array AR; each type-04 record loads two additional elements.

Blanks and other fields can appear on the input records since the array elements and their index are identified by From and To entries.

This method requires a minimum of coding and no calculations to set up the array. Extra work, however, is required to set up the indexing scheme for the input records.

Example 2: In Figure 72 we see a method whereby eighteen 5-position elements of array AR1 are loaded with only two specification lines. On succeeding lines of the Input Sheet other elements of AR1 are loaded one after another until the array is full. Each additional element is coded on a separate line. Each new record requires a separate means of identification. For example, if another 03 record followed the first, the fields on the second record would overlay the fields read in from the first record.

The method illustrated in Example 2 works well for small arrays.

IBM International Business Machine Corporation — **RPG INPUT SPECIFICATIONS** — GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page of | Program Identification

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Position (1)	C/Z/D	Character	Field Location From	Field Location To	Decimal Positions	Field Name
01	I	FILE	AA	Ø3	8Ø	C	1				
02	I							2	3	Ø	I1
03	I							4	8	Ø	AR,I1
04	I							9	1Ø	Ø	I2
05	I							11	15	Ø	AR,I2
06	I							16	17	Ø	I3
07	I							18	22	Ø	AR,I3
08	I										
09	I										
10	I										
11	I							65	66	Ø	I1Ø
12	I							67	71	Ø	AR,I1Ø
13	I		BB	Ø4	8Ø	C	2				
14	I							2	3	Ø	I1
15	I							4	8	Ø	AR,I1
16	I							9	1Ø	Ø	I2
17	I							11	15	Ø	AR,I2
18	I										
19	I										
20	I										
	I										
	I										
	I										
	I										
	I										

Figure 71. Building an Array Using Input Fields as Indexes

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050*
Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	From (44-47)	To (48-51)	Decimal Positions (52)	Field Name (53-58)
01	I	FILE1	AA			03	80		C	1				
02	I										1	90	0	AR1
03	I		BB			04	80		C	2				
04	I										1	50	0	AR1,19
05	I										6	100	0	AR1,20
06	I													
07	I													
08	I													
09	I													
10	I													
11	I													
12	I													
13	I													
14	I													
15	I													
16	I													
17	I													
18	I													
19	I													
20	I													

Figure 72. Building an Array Using Fixed Indexes

Example 3: The specifications in Figure 73 perform the function of tabulating three levels of totals. The fields FIELDA, FIELDB, FIELDC, and FIELDD are added, as they are read from input records, to the first level totals L1A, L1B, L1C, and L1D. These first level totals are added at the time of an L1 control break to totals L2A, L2B, L2C, and L2D. Similarly, at an L2 control break the second level totals are added to third level totals L3A, L3B, L3C, and L3D. In addition, as control breaks occur, L1, L2, and L3 total output is performed; total fields are zeros after they are written on the output device.

IBM International Business Machine Corporation — RPG CALCULATION SPECIFICATIONS — Form GX21-9093, Printed in U.S.A.

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions
01	C		FIELDA	ADD	L1A	L1A	6	2
02	C		FIELDB	ADD	L1B	L1B	6	2
03	C		FIELDC	ADD	L1C	L1C	6	2
04	C		FIELDD	ADD	L1D	L1D	6	2
05	C	L1	L1A	ADD	L2A	L2A	6	2
06	C	L1	L1B	ADD	L2B	L2B	6	2
07	C	L1	L1C	ADD	L2C	L2C	6	2
08	C	L1	L1D	ADD	L2D	L2D	6	2
09	C	L2	L2A	ADD	L3A	L3A	6	2
10	C	L2	L2B	ADD	L3B	L3B	6	2
11	C	L2	L2C	ADD	L3C	L3C	6	2
12	C	L2	L2D	ADD	L3D	L3D	6	2

IBM International Business Machine Corporation — RPG OUTPUT SPECIFICATIONS — GX21-9090 U/M 050, Printed in U.S.A.

Line	Form Type	Type (H/D/T/E)	Space Before	Space After	Output Indicators	Field Name	Edit Codes	Blank After (B)	End Position in Output Record
01	O	T	2	Ø	L1				
02	O					L1A	K	B	15
03	O					L1B	K	B	3Ø
04	O					L1C	K	B	45
05	O					L1D	K	B	6Ø
06	O	T	2	Ø	L2				
07	O					L2A	K	B	15
08	O					L2B	K	B	3Ø
09	O					L2C	K	B	45
10	O					L2D	K	B	6Ø
11	O	T	2	Ø	L3				
12	O					L3A	K	B	15
13	O					L3B	K	B	3Ø
14	O					L3C	K	B	45
15	O					L3D	K	B	6Ø
16	O								
17	O								
18	O								
19	O								
20	O								

Figure 73. Calculating Totals Without Arrays

Figure 74 shows the same functions being performed along arrays. Note the reduction in coding required to specify the functions. For example, line 5 of the Calculation Sheet performs the same function as lines 5 through 8 of the Calculation Sheet of Figure 73. Similarly, the output specifications are reduced from 15 lines to 6. (Notice, however, that the method using array results in only two positions between array elements.)

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS (IBM, Form X21-9091)

Extension Specifications

Line	Form Type	From Filename	To Filename	Table or Array Name	Number of Entries Per Record	Number of Entries Per Table or Array	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)	Comments
01	E			SL1		4	6		2		
02	E			SL2		4	6		2		
03	E			SL3		4	6		2		

RPG CALCULATION SPECIFICATIONS (IBM, Form GX21-9093)

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Resulting Indicators	Comments
01	C			FIELDA	ADD	SL1,1	SL1,1				
02	C			FIELDB	ADD	SL1,2	SL1,2				
03	C			FIELDC	ADD	SL1,3	SL1,3				
04	C			FIELDD	ADD	SL1,4	SL1,4				
05	C	L1		SL1	ADD	SL2	SL2				
06	C	L2		SL2	ADD	SL3	SL3				

RPG OUTPUT SPECIFICATIONS (IBM, GX21-9090)

Line	Form Type	Filename	Type (H/D/T/E)	Space Before	Space After	Output Indicators	Field Name	Edit Codes	B/A/C/1-9/R	End Position in Output Record	Constant or Edit Word
01	O		T	2	0	L1					
02	O						SL1	K	B	60	
03	O		T	2	0	L2					
04	O						SL2	K	B	60	
05	O		T	2	0	L3					
06	O						SL3	K	B	60	

Figure 74. Calculating Totals With Arrays

Example 4: This example illustrates the use of three arrays defined as follows. Refer to Figure 75.

Array Name	*Number of Fields*	*Field Length*
ARA	4	5
ARB	5	10
ARC	6	4

Array ARA is contained in the input records corresponding to indicator 01, ARB in the records corresponding to 02, and ARC in both types of records. Array ARC and the first field of array ARA are to be included together in an output record as are arrays ARC and a field (identified by field X1) of array ARB. Every field in array ARC is edited according to the edit word 0ƀ.ƀƀ&CR. (where b represents a blank).

Assume that the contents of the arrays in the first two input records are:

Record	*Array*	*Array Contents*
1	ARA	12345678901234567890
	ARC	0123456789012345678987 6N (note that N equals minus 5)
2	ARB	JOHNƀDOEƀƀJOEƀSMITHƀLEEƀ MARXƀƀJIMƀKNOTSƀTIMƀTYLERƀ
	ARC	(The same as in record 1)

In the first output record, the location and contents of the arrays are (ƀ represents a blank):

Array	*Location*	*Contents*
ARA (first field)	85-89	12345
ARC	37-84	ƀ1.23ƀƀƀ45.67ƀƀƀ 89.01ƀƀƀ23.45ƀƀƀ 67.89ƀƀb87.65ƀCR

For the second output record assume that the contents of field X1 is 4. The locations and contents of the arrays are:

Array	*Location*	*Contents*
ARB (fourth field)	91-100	JIMƀKNOTSƀ
ARC	37-84	The same as in the first record.

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

IBM International Business Machine Corporation — Form X21-9091, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Extension Specifications

Line (3-5)	Form Type (6)	Record Sequence of the Chaining File / Number of the Chaining Field / From Filename (7-18)	To Filename (19-26)	Table or Array Name (27-32)	Number of Entries Per Record (33-35)	Number of Entries Per Table or Array (36-39)	Length of Entry (40-42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46-51)	Length of Entry (52-54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58-74)
01	E			ARA		4	5		Ø							
02	E			ARB		5	1Ø									
03	E			ARC		6	4		2							

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 U/M 050*, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2: Position / Not / C/Z/D / Character (28-34)	3: Position / Not / C/Z/D / Character (35-41)	Stacker Select (42)	P/B/L/R (43)	Field Location From (44-47)	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 / Minus / Zero or Blank (65-70)
01	I	IN	AA			Ø1	8Ø		C													
02	I	OR				Ø2	8Ø		C	1												
03	I														51	74		ARC				
04	I														1	2Ø		ARA			Ø1	
05	I														1	5Ø		ARB			Ø2	
06	I																					

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050*, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Commas	Zero Balances to Print	No Sign	CR	-	
Yes	Yes	1	A	J	X = Remove Plus Sign
Yes	No	2	B	K	Y = Date Field Edit
No	Yes	3	C	L	Z = Zero Suppress
No	No	4	D	M	

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T/E) (15)	Stacker #/Fetch(F) (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators (23-31)	Field Name / *AUTO (32-37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40-43)	P/B/L/R (44)	Constant or Edit Word (45-70)
01	O	OUT	D			1									
02	O														
03	O									ARC			84		'Ø &CR'
04	O								Ø1	ARA,1	Z		89		
05	O								Ø2	ARB,X1			1ØØ		
06	O														

Figure 75. Using Arrays to Format Field Output

Example 5: Figure 76 shows a method of writing short arrays on the output device. The contents of one element of a 22-element array, AR2, is written to the output file ARFILE each time the specification in line 3 of the Calculation Sheet is performed.

Example 6: Figure 77 shows a method of writing a large array on the output device. The number of fields printed on a line depends on the value assigned to the compare on line 10 of the Calculation Sheet. If an edit code is used, each array field will be separated by two spaces. These spaces must be considered when computing the end position in the output specifications.

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093 Printed in U.S.A.

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators (Plus / Minus / Zero; High / Low / Equal)	Comments
01	C	LR			Z-ADD	1	IN	2	Ø			
02	C	LR		DUMP	TAG							
03	C	LR			EXCPT							
04	C	LR		IN	ADD	1	IN					
05	C	LR		IN	COMP	22					5Ø	
06	C	LR	N5Ø		GOTO	DUMP						
07	C											
08	C											
09	C											
10	C											
11	C											

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050* Printed in U.S.A.

Line	Form Type	Filename	Type (H/D/T/E)	Stacker #/Fetch (F)	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name	Edit Codes	B/A/C/1-9/R	End Position in Output Record	P/B/L/R	Constant or Edit Word
01	O	ARFILE	E		1				LR						
02	O									AR2,IN			2Ø		
03	O														
04	O														
05	O														
06	O														
07	O														
08	O														
09	O														
10	O														
11	O														

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Figure 76. Printing One Array Element Per Line

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page of | Program Identification

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators High (Plus, 1>2)	Low (Minus, 1<2)	Equal (Zero, 1=2)	Comments
01	C	LR			Z-ADD	1	IN	2	0					
02	C	LR		DUMP	TAG									
03	C	LR			Z-ADD	1	I1	2	0					
04	C	LR		UP	TAG									
05	C	LR			MOVE	AR2,IN	AR1,I1							
06	C	LR		I1	ADD	1	I1							
07	C	LR		IN	ADD	1	IN							
08	C	LR		IN	COMP	50					12			
09	C	LR	12		GOTO	OUT								
10	C	LR		I1	COMP	10					14			
11	C	LR	N14		GOTO	UP								
12	C	LR		OUT	TAG									
13	C	LR			EXCPT									
14	C	LR	14N12		GOTO	DUMP								
15	C													
16	C													
17	C													
18	C													
19	C													
20	C													

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050*, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page of | Program Identification

Line	Form Type	Filename	Type (H/D/T/E)	Stacker #/Fetch(F)	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name	Edit Codes	B/A/C/1-9/R	End Position in Output Record	P/B/L/R	Constant or Edit Word
01	O	ARFILE	E			1			LR						
02	O									AR1		B	100		
03	O														
04	O														
05	O														
06	O														
07	O														
08	O														
09	O														
10	O														
11	O														

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign; Y = Date Field Edit; * = Field Edit; Z = Zero Suppress

Figure 77. Printing More Than One Array Element Per Line

Line counter specifications should be used for each printer file (except the console printer) in your program. If the dual carriage feature is used, two specification lines should be completed. Line counter specifications indicate at what line overflow occurs and the length of the form used in a printer. Both of these entries must be specified on the Line Counter Sheet (Figure 78). If no line counter specifications exist, the forms length used will be either:

1. The forms length specified on the // FORMS card, or

2. The forms length specified at system generation time (if no // FORMS card was specified).

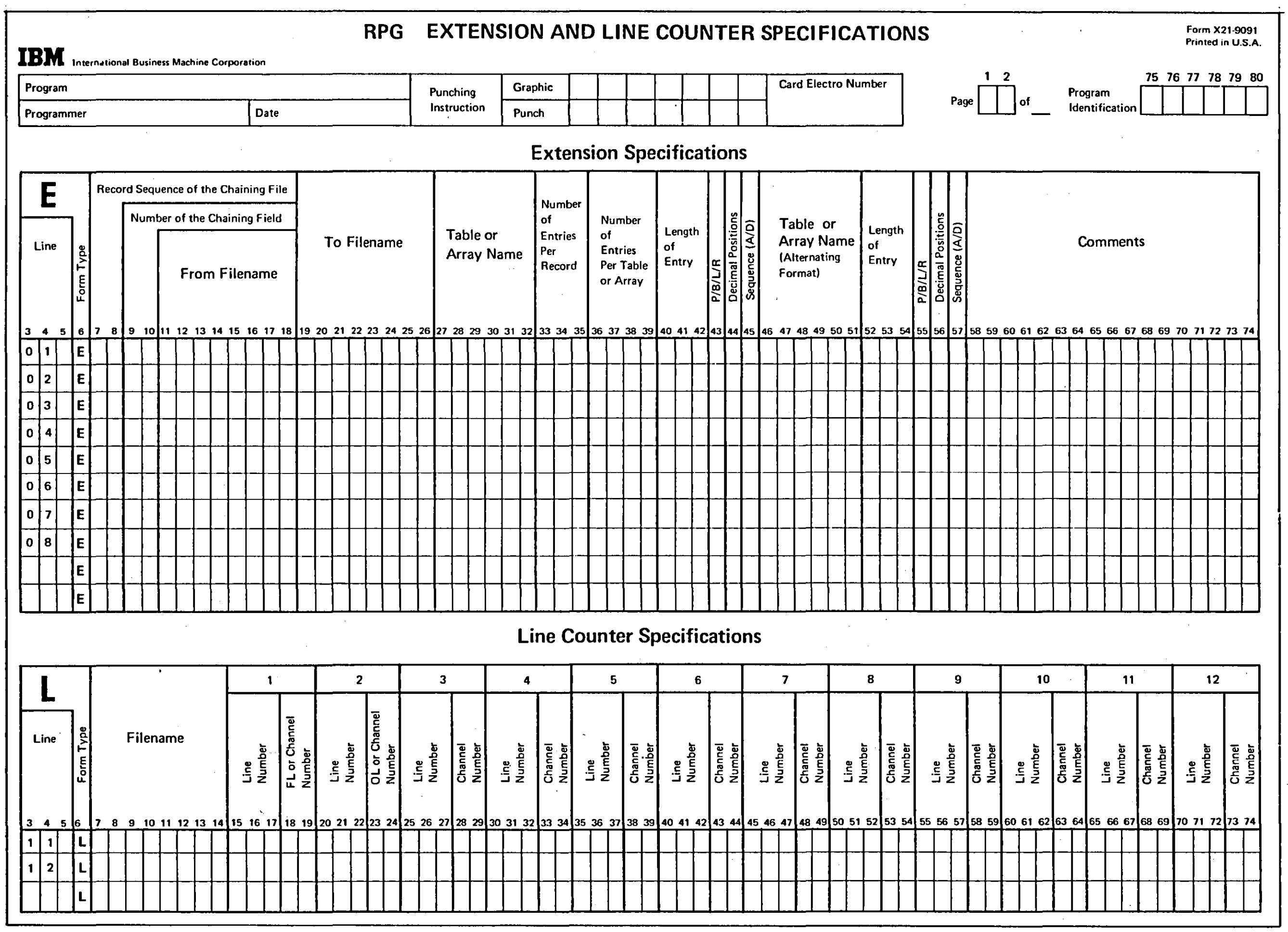
In either case, the overflow line is assumed to be six lines less than the specified forms length.

COLUMNS 1-2 (PAGE)

See Chapter 2.

COLUMNS 3-5 (LINE)

See Chapter 2.


RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

Form X21-9091
Printed in U.S.A.

IBM International Business Machine Corporation

Program
Programmer
Date
Punching Instruction
Graphic
Punch
Card Electro Number
Page 1 2 of
Program Identification 75 76 77 78 79 80

Extension Specifications

E
Line
Form Type
Record Sequence of the Chaining File
Number of the Chaining Field
From Filename
To Filename
Table or Array Name
Number of Entries Per Record
Number of Entries Per Table or Array
Length of Entry
P/B/L/R
Decimal Positions
Sequence (A/D)
Table or Array Name (Alternating Format)
Length of Entry
P/B/L/R
Decimal Positions
Sequence (A/D)
Comments

Line	Form Type
01	E
02	E
03	E
04	E
05	E
06	E
07	E
08	E
	E
	E

Line Counter Specifications

L
Line
Form Type
Filename
1: Line Number, FL or Channel Number
2: Line Number, OL or Channel Number
3–12: Line Number, Channel Number

Line	Form Type
11	L
12	L
	L

Figure 78. Extension and Line Counter Sheet

COLUMN 6 (FORM TYPE)

An *L* must appear in column 6.

COLUMNS 7-14 (FILENAME)

Use columns 7-14 to identify the output file to be written on the printer. Filename must begin in column 7.

Any filename entered in these columns must be previously defined on the File Description Sheet. The output device assigned to the file on the File Description Sheet must be a printer.

COLUMNS 15-17 (LINE NUMBER–NUMBER OF LINES PER PAGE)

Entry	*Explanation*
12-112	Number of printing lines available.

Columns 15-17 specify the exact number of lines available on the form or page to be used. The entry must end in column 17. Leading zeros may be omitted. If a number less than 12 is specified, RPG II does not indicate an error. However, unpredictable skipping errors may occur.

COLUMNS 18-19 (FORM LENGTH)

Entry	*Explanation*
FL	Form length

Columns 18-19 must contain the entry *FL*. This entry indicates that the preceding entry (columns 15-17) is the form length.

COLUMNS 20-22 (LINE NUMBER–OVERFLOW LINE)

Entry	*Explanation*
1-112	A line number from 1-112 is the overflow line.

Columns 20-22 specify the line number that is the overflow line. The entry must end in column 22. Leading zeros may be omitted.

When the destination line of a space, skip, or print operation is a line beyond the overflow line you have specified (but not beyond the form length), the overflow indicator turns on to indicate that the end of the page is near. When the overflow indicator is on, the following occur before forms advance to the next page:

1. Detail lines are printed (if this part of the program cycle has not already been completed).
2. Total lines are printed.
3. Total lines conditioned by the overflow indicator are printed.

Because all these lines are printed on the page after the overflow line, you have to specify the overflow line high enough on the page to allow all these lines to print. You know the data you will be printing out after the overflow line is reached. Thus, you can judge what line should be the overflow line on this basis. See *Columns 33-34*, Chapter 4 for a discussion of overflow.

COLUMNS 23-24 (OVERFLOW LINE)

Entry	*Explanation*
OL	Overflow line

Columns 23-24 must contain the entry *OL*. This entry indicates that the preceding entry (columns 20-22) is the overflow line.

COLUMNS 25-74

Columns 25-74 are not used.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

See Chapter 2.

Input specifications describe the data files, records, and fields of the records to be used by your program. These specifications may be divided into two categories:

1. File and record type identification (columns 7-42). These specifications describe the input record and its relationship to other records in the file.

2. Field description entries (columns 43-74). These specifications describe the fields in the records.

The specifications are written on the Input Sheet (Figure 79). The field description entries must start at least one line lower than file and record type identification entries.

COLUMNS 1-2 (PAGE)

See Chapter 2.

COLUMNS 3-5 (LINE)

See Chapter 2.

COLUMN 6 (FORM TYPE)

An *I* must appear in column 6.

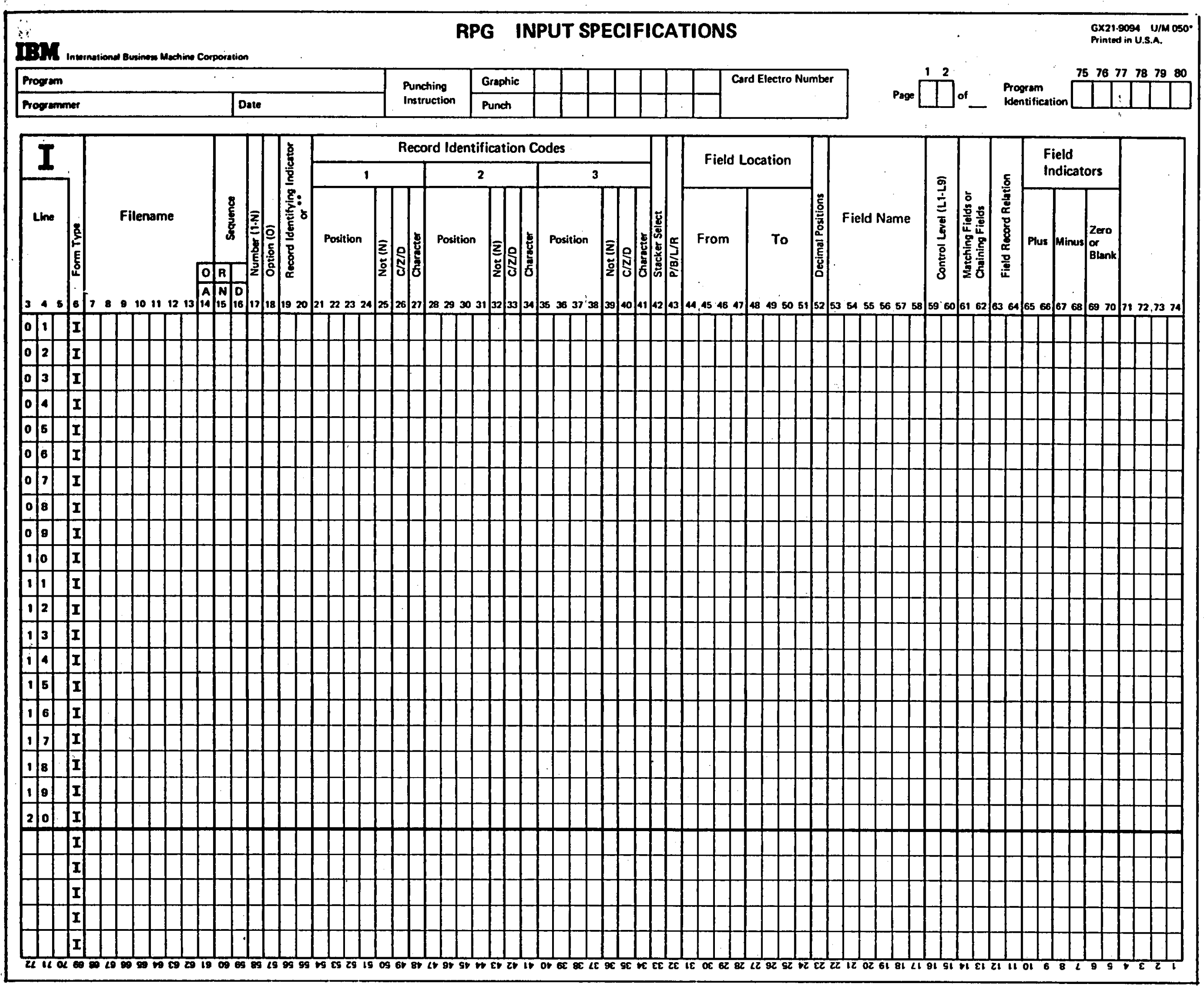


Figure 79. Input Sheet

COLUMNS 7-14 (FILENAME)

Columns 7-14 identify the input, update, or combined file you are describing. The filename must begin in column 7 and conform to RPG II naming specifications. Use the same filename given in the file description specifications. The name of every input, update, or combined file (except table input files and record address files) described in the file description specifications must be entered at least once on this sheet. The filename must appear on the first line that contains information concerning the records in that file. If the filename is omitted, the last filename entered is assumed to be the file being described. All records and fields for one file must be completely described before another file can be described.

COLUMNS 15-16 (SEQUENCE)

Entry	*Explanation*
Any two alphabetic characters	No check for special sequence.
Any two-digit number	Check for special sequence.

Columns 15-16 may contain a numeric entry which assigns a special sequence to different record types in a file.

If different types of records do not need to be in any special order, use two alphabetic characters (see *Examples, Example 1*). Alphabetic characters must be used for chained files and look ahead records. Within one file record types having alphabetic and numeric sequence entries can be specified for the same file, but all alphabetic entries must be before the numeric entries.

Use columns 15-16 to assign sequence numbers to different types of records within a file. Your job may require that one record type (identified by a record identification code) must appear before another record type within a sequenced group. For instance, you may want a name record before an address record. You must provide a record identification code for each type of record and then number the record types in the order that they should appear. The program will check this order as the records are read. The first record type must have the lowest sequence number (01), the next record type should be given a higher number, etc. (See *Examples, Example 2*.)

Numeric sequence numbers only ensure that all records of record type 01 precede all records of record type 02, etc., in any sequenced group. The sequence numbers do not ensure that records within a record type are in any certain order. Numeric sequence numbers have no relationship with control levels, nor do they provide for sequence checking of data in fields of a record (see *Examples, Example 3*).

Gaps in sequence numbers are allowed, but the numbers used must be kept in ascending order. The first sequence number *must* be 01.

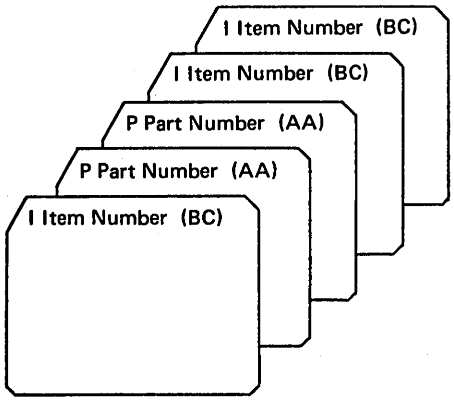
A record type out of sequence causes the program to stop. The program may be restarted by pressing the START key on the processing unit. The record that causes the halt is bypassed and the next record is read from the same file.

Records in an AND or OR line cannot have a sequence entry in these columns. The entry in these columns from the previous line also applies to the card in the OR line. See *Columns 53-58* in this chapter for information on OR relationships.

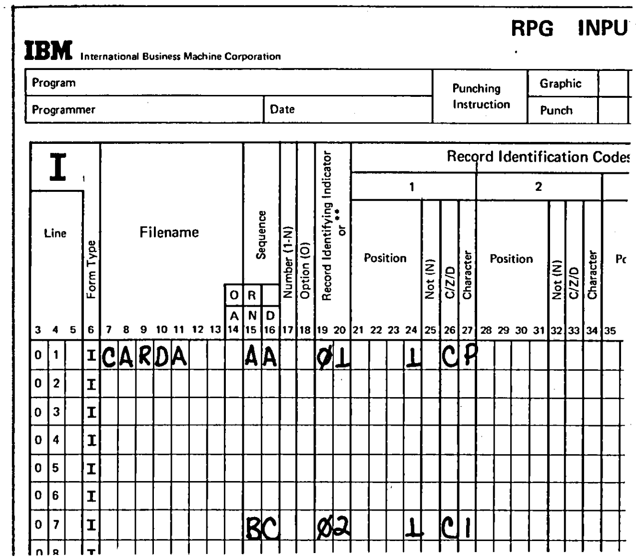
Examples

Example 1: Figure 80, insert A, shows a file having two types of records (part number and item number) which may appear in any order. Since they are not to be checked for sequencing, they are assigned two alphabetic characters (AA and BC, respectively) instead of numbers. See Figure 80, insert B for the coding of this example.

Example 2: Figure 81, insert A shows the order of four different types of records within a file. The records are arranged in groups according to some control field. The name record is first in each group and is assigned sequence number 01. Street record is next and is assigned 02. City/state record is 03. Item number is last and is assigned 07. See Figure 81, insert B for the coding of this example.

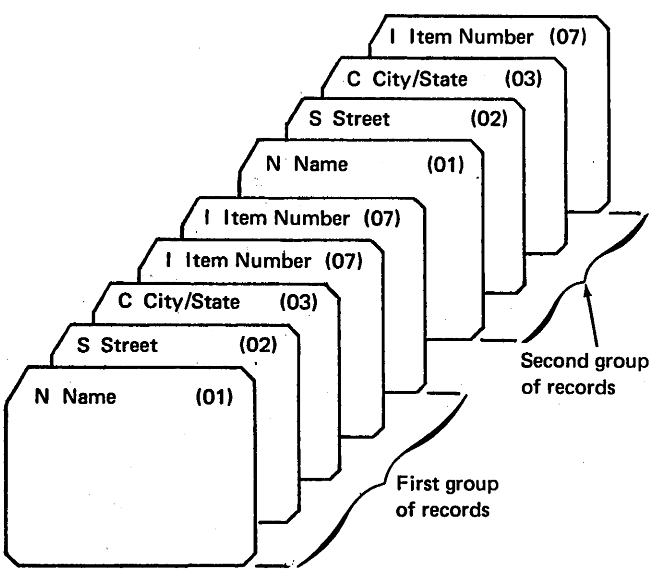


Ⓐ

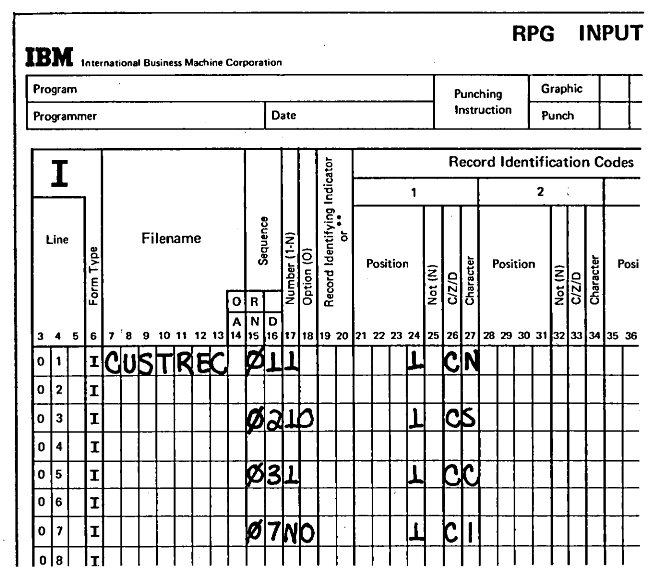


Ⓑ

Figure 80. Unsequenced Card Types in a File



Ⓐ



Ⓑ

Figure 81. Sequence Checking of Record Types

Example 3: Figure 82 shows three groups of four different record types. Each group is in proper sequence according to the assigned sequence numbers (01, 02, 03, and 07). Notice, however, that the city/state record for group B is in group C and vice versa. The sequence entry which you specify in columns 15-16 will not catch this mistake since the sequence entry does not cause the data on the record to be checked.

COLUMN 17 (NUMBER)

Entry	*Explanation*
Blank	Record types are not being sequence checked (columns 15-16 have alphabetic entries).
1	Only one record of this type is present in the sequenced group.
N	One or more records of this type may be present in the sequenced group.

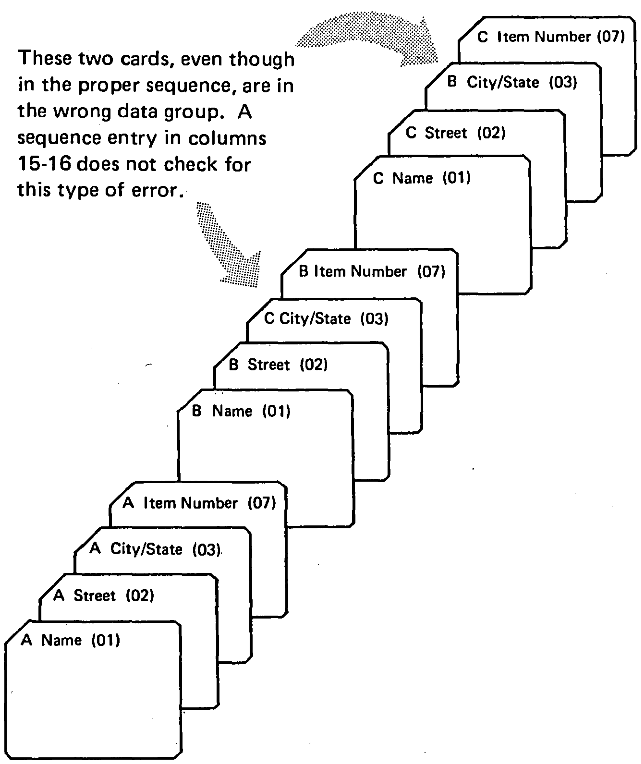


Figure 82. Correct Card Sequence (Incorrect Data in Each Group)

Use column 17 only if sequence checking is to be done (columns 15-16 contain numbers). Often, when sequence checking, you may have more than one record of a particular type within the sequenced group (see *Example*). If this occurs, you must indicate by an 'N' in column 17 that more than one record of one type may be found in the sequence group.

AND or OR lines (columns 14-16 have the letters AND or OR) should not have an entry in this column. It is assumed that the number of records of this type to be found in the sequenced group is the same as the entry in column 17 of the previous line. (See *Columns 21-41* in this chapter for more information on AND lines; see *Columns 53-58* for more information on OR lines.)

Example

Figure 83 shows a sequenced record file in which there is more than one record per type in a group. The record type called item number appears three times.

There is probably no reason for a name, street, or city/state record to appear more than once in one group. A *1* is entered in column 17 to indicate that these record types appear only once in each group. However, since one person may have purchased more than one item, there may be two or more item number records per group; an *N* is entered in column 17 for this field. See Figure 81, insert B for the coding of this example.

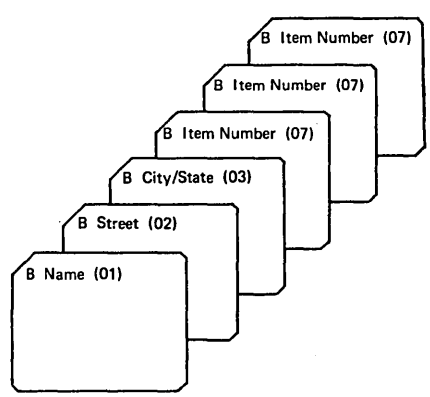


Figure 83. Sequenced Card File (More Than One Record Per Type in a Group)

COLUMN 18 (OPTION)

Entry	*Explanation*
Blank	Record type must be present (if sequence checking is specified).
O	Option. Record type may or may not be present.

Column 18 is used when record types are being sequence checked. A blank entry specifies that a record of this record type must be present in each sequenced group.

The O entry specifies that a record of this record type may or may not be present in each sequenced group (see *Example*). If all record types are optional, no sequence errors will be found.

AND or OR lines should not have an entry in this column. The entry in this column on the previous line also applies to this line. (See *Columns 21-41* in this chapter for more information on AND lines; see *Columns 53-58* for more information on OR lines.)

Example

Figure 84 shows a sequenced card file in which a card type may be optional. For instance, the street or item number records may not be included. Since it is not always necessary to have a street address, this record is optional. Suppose this job required a list of all items purchased during one month by the individual named in the name record. It is possible that a person might not buy anything during the month. In this case, there would be no item record; therefore, the item record would also be optional. (See Figure 81, insert B for a coding example.)

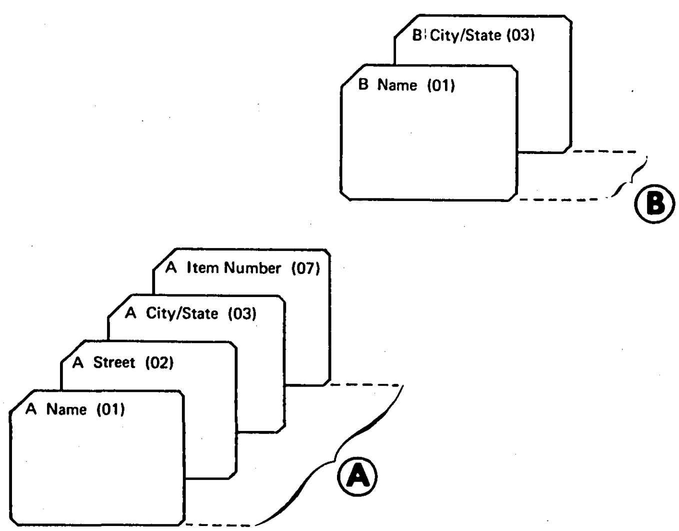


Figure 84. Sequenced Card File (Optional Record Types)

COLUMNS 19-20 (RECORD IDENTIFYING INDICATOR, **)

Entry	*Explanation*
01-99	Record identifying indicator (see general discussion under *Columns 54-59*, Chapter 8).
L1-L9	Control level indicator, used for a record identifying indicator when a record type rather than a control field signals the start of a new control group (see general discussion under *Columns 59-60*, Chapter 7).
LR	Last record indicator (see *Columns 7-8*, Chapter 8).
H1-H9	Halt indicator, used for a record identifying indicator when checking for a record type that causes an error condition (see general discussion under *Columns 54-59*, Chapter 8).
**	Look-ahead fields.
TR	Spread cards.

Columns 19-20 may be used for three purposes:

1. Specifying record identifying indicators.
2. Indicating look-ahead fields.
3. To specify the trailer portion of spread cards.

RECORD IDENTIFYING INDICATORS

Use columns 19-20 to assign an indicator to each record type. When you have different types of records within a file, you often want to do different operations for each record type. Therefore, you must have some way of knowing which type of record has just been read. To do this, you assign different record identifying indicators to each record type. Whenever a record type is selected to be processed next, its corresponding identifying indicator is turned on. (All other record identifying indicators are off at this time, unless chained files or demand files are being processed, when several may be on at the same time.) This indicator signals throughout the rest of the program cycle which record type has just been selected. A record identifying indicator need not be assigned if you are not concerned about different record types.

Because the record identifying indicator is on for the rest of the program cycle, you may use it to condition calculation operations (see *Columns 9-17* in Chapter 8) and output operations (see *Columns 23-31* in Chapter 9).

Note: Record identifying indicators are not on during Last Record time. See *Detailed RPG II Object Program Cycle* in Appendix C.

Record identifying indicators do not have to be assigned in any order.

When a control level indicator used as a record identifying indicator turns on to reflect the type of record read, only that one control level indicator turns on. All lower level indicators remain off.

You may assign the same indicator to two or more different record types provided you want the same operations performed on these types. This can be done by using the OR relationship (see *Columns 21-41* in this chapter).

No record identifying indicator may be specified in the AND line of an AND relationship. Record identifying indicators for OR lines may be specified for every record type in the OR relationship that requires special processing. An OR line with any record identifying indicator not used elsewhere in the program allows unwanted records (such as blank records) to be bypassed. (See *Columns 21-41* in this chapter for information on AND lines. See *Columns 53-58* in this chapter for information on OR lines.)

LOOK AHEAD FIELDS

Use asterisks in columns 19-20 to indicate that fields named in columns 53-58 on the following specifications lines are look-ahead fields. A look-ahead field allows you to look at information in a field on the next record that is available for processing in any input file. In update and combined files, the look-ahead field is for the record currently in process.

Two of the uses for look-ahead fields are:

1. Determining when the last card of a control group is being processed.

2. Extending the RPG II matching record capability.

Look-ahead fields can be used with input, update, and combined files whether or not they are processed by a record address file. They cannot be specified for chained or demand files or files that contain header/trailer records. You can describe one set of look-ahead fields per file; the description applies to all records in the file, regardless of their type. (The specifications for describing the fields are given later.) Look-ahead fields cannot be altered in the program (cannot be used as a result field or blanked after).

Note: An extra buffer is provided by the RPG program for the look-ahead fields.

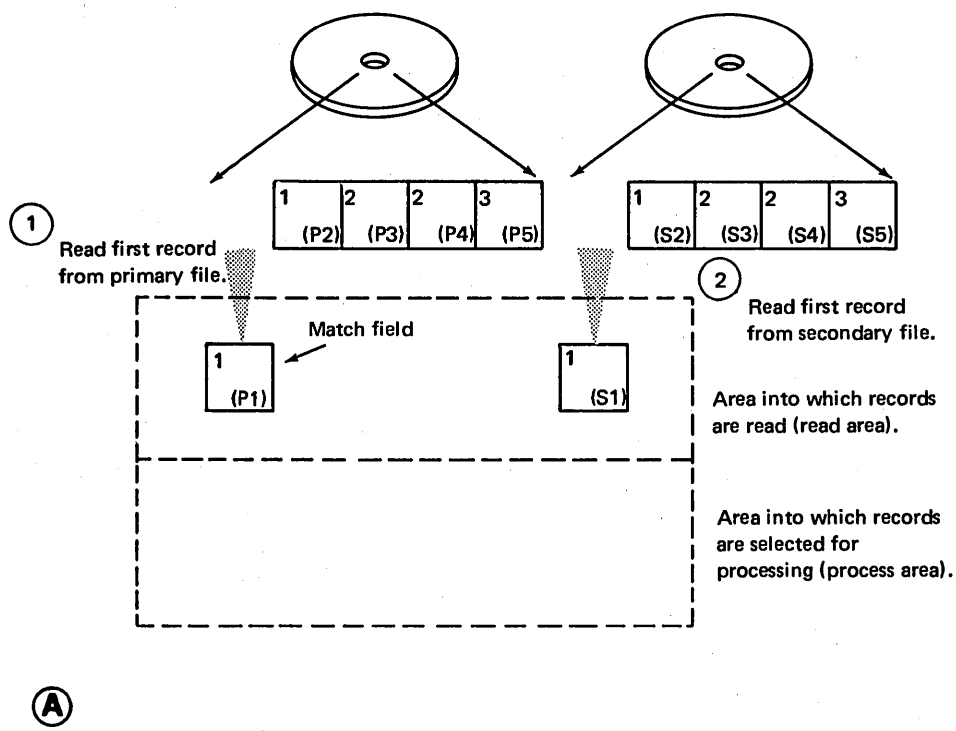
If you wish to use information both before and after the record is selected for processing, you must describe the field twice; once as a look-ahead field and once as a normal field.

For combined and update files, the look-ahead fields apply to the next record in the file only if the current record was not read from that file. Therefore, when you are reading from only one file and the file is a combined or update file, look-ahead fields always apply to the current record.

Figure 85 shows processing three records from two input files, one primary and one secondary. The first record from each file is read (see Figure 85, insert A). In Figure 85, insert B, record P1 is selected for processing; in Figure 85, insert C, record P2; and in Figure 85, insert D, record S1. The records available for look-ahead during the processing of these records are:

Record Processed	*Records Available*
P1	P2 and S1
P2	P3 and S1
S1	P3 and S2

In general, when the record being processed is from an input file, the next record in the input file is available as are the records which were read but not selected from the other files.



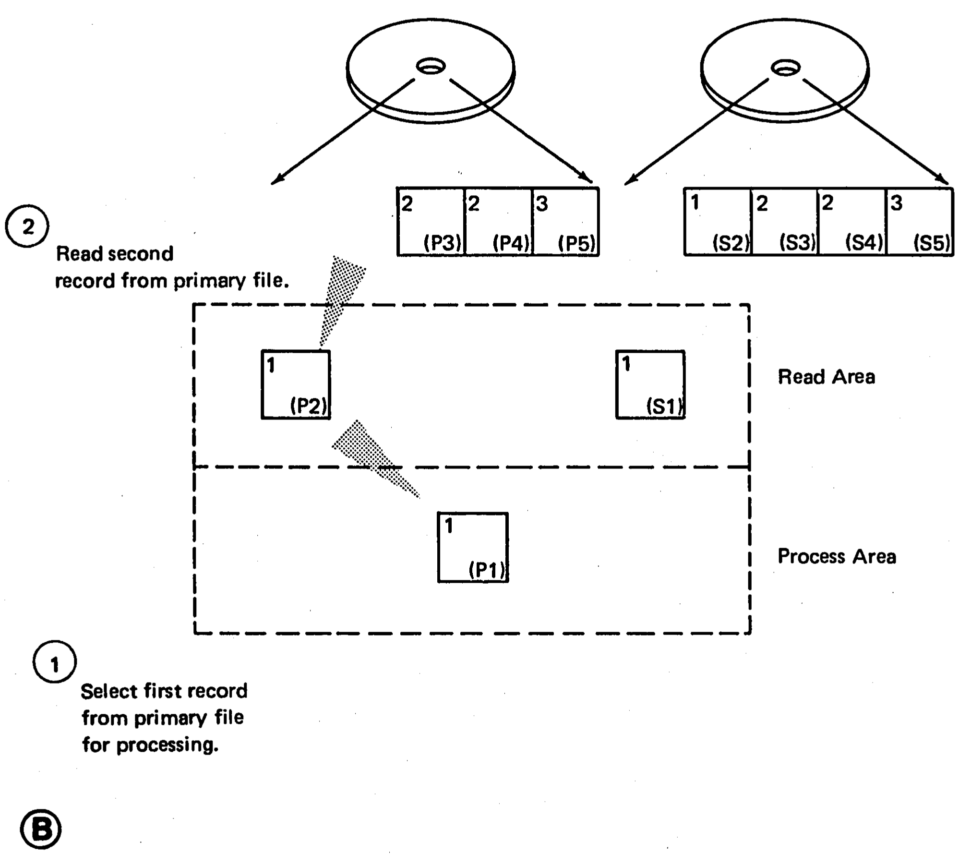


Figure 85 (Part 1 of 2). Available Records: Two Input Files

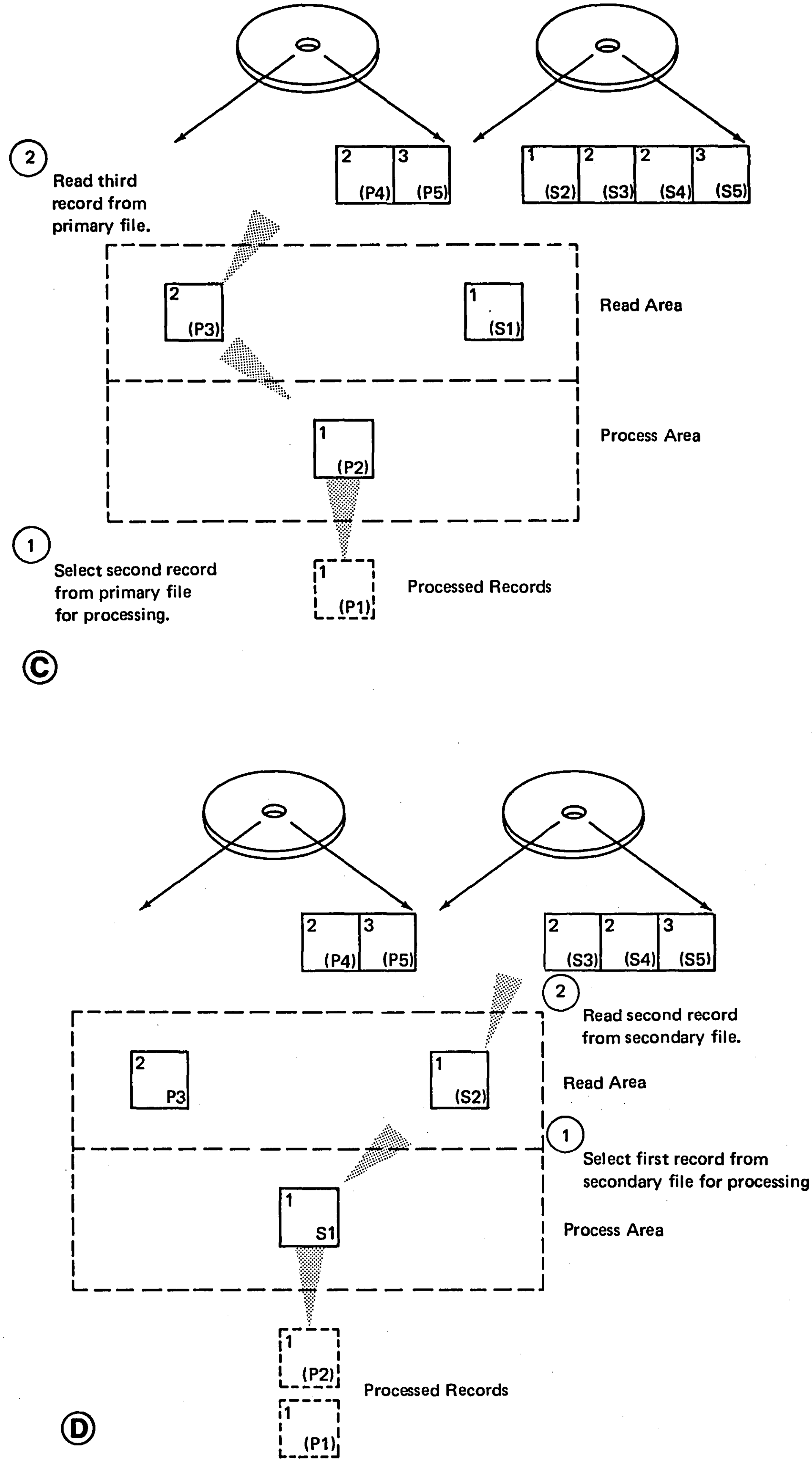


Figure 85 (Part 2 of 2). Available Records: Two Input Files

Figure 86 shows the same files as Figure 85 with one exception: file A is an update file. The records available for look-ahead during the processing of the three records are:

Records Processed	*Records Available*
U1	U1 and S1
U2	U2 and S1
S1	U3 and S2

In general, when the record being processed is from a combined or update file, only the records which were read, but not selected, from the other files are available for look-ahead. The next record from the combined or update file is not read until after the current record has been processed. Therefore, the next record from the combined or update file is not available for look-ahead.

After the last record from a file has been processed, every look-ahead field for the file is automatically filled with 9's. For example, a field three record-positions long contains 999. The 9's remain in the fields until the job ends. Note also that blank after (B in column 39 of the Output Sheet) cannot be used with look-ahead fields.

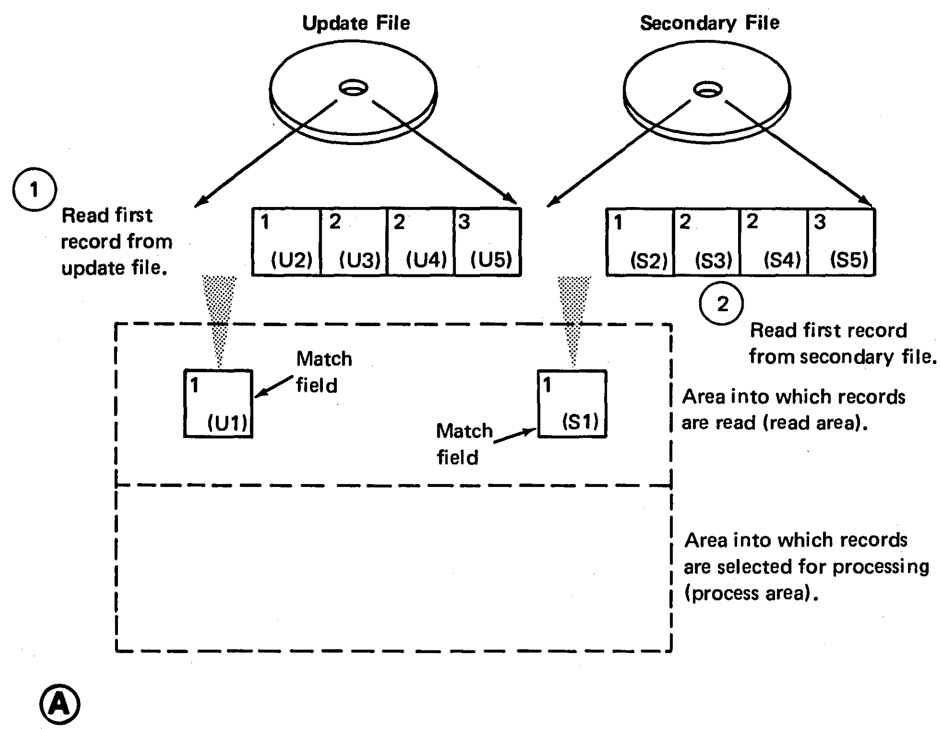
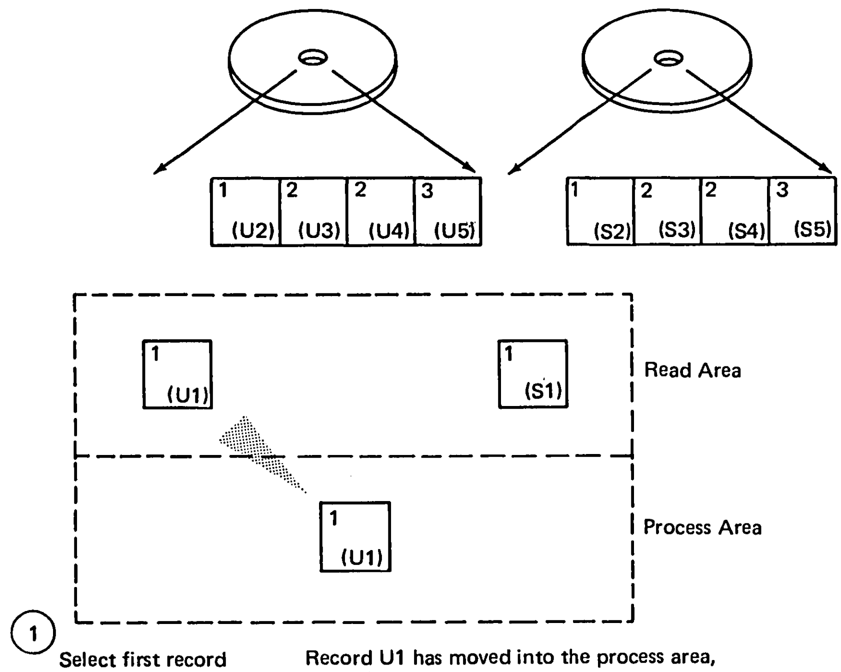


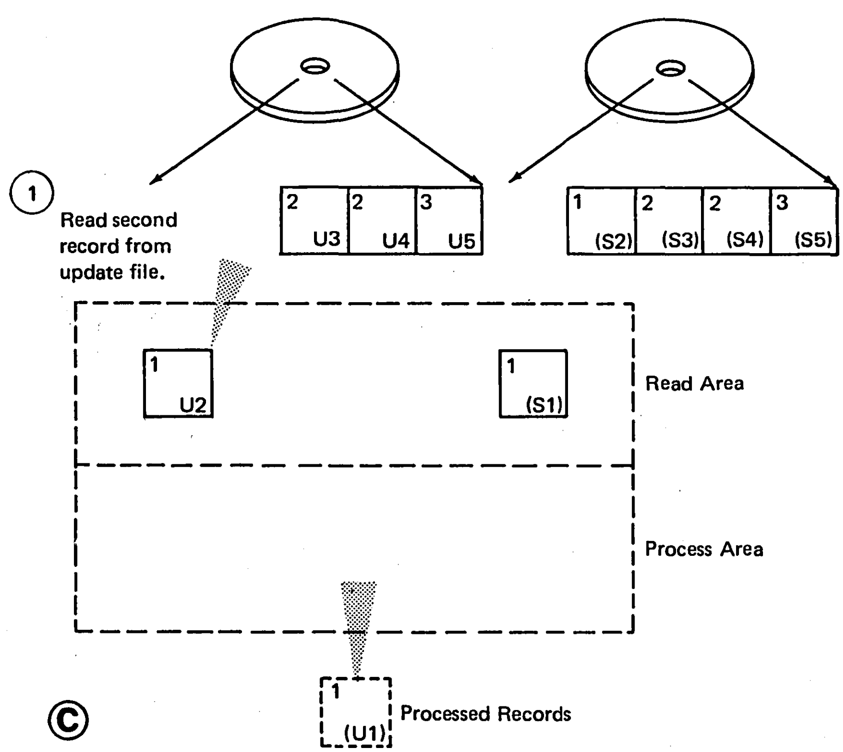
Figure 86 (Part 1 of 3). Available Records: One Input File, One Update File



Select first record from update file for processing.

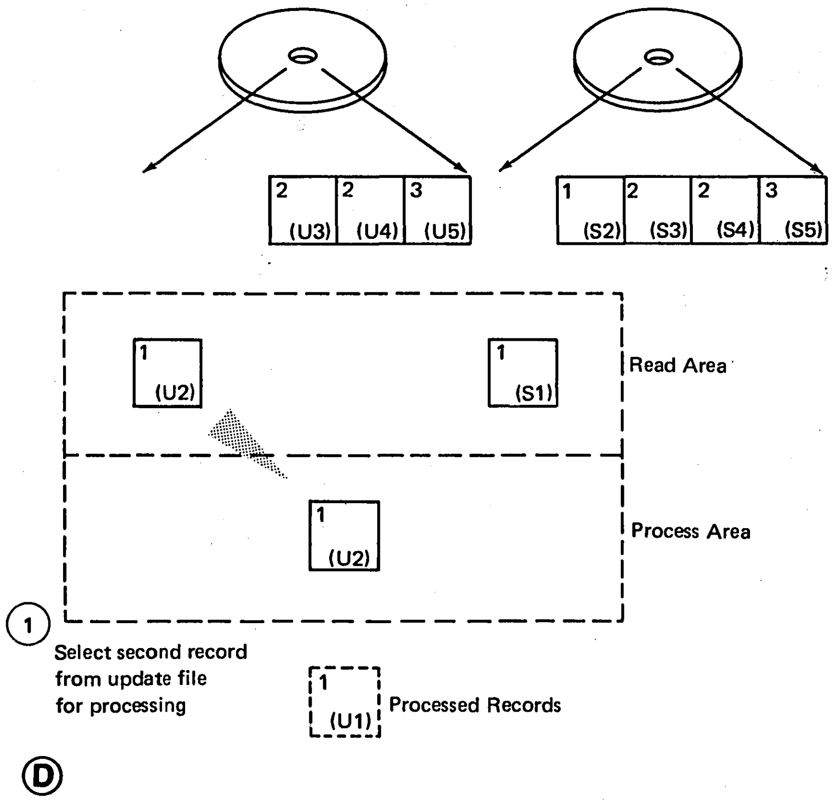
Record U1 has moved into the process area, but a data image of U1 remains in the read area until U2 is read in. U2 will not be read in until U1 is completely processed. Therefore, while U1 is in the process area, records available for look-ahead are S1 and U1 (the data image).

Ⓑ



Ⓒ

Figure 86 (Part 2 of 3). Available Records: One Input File, One Update File



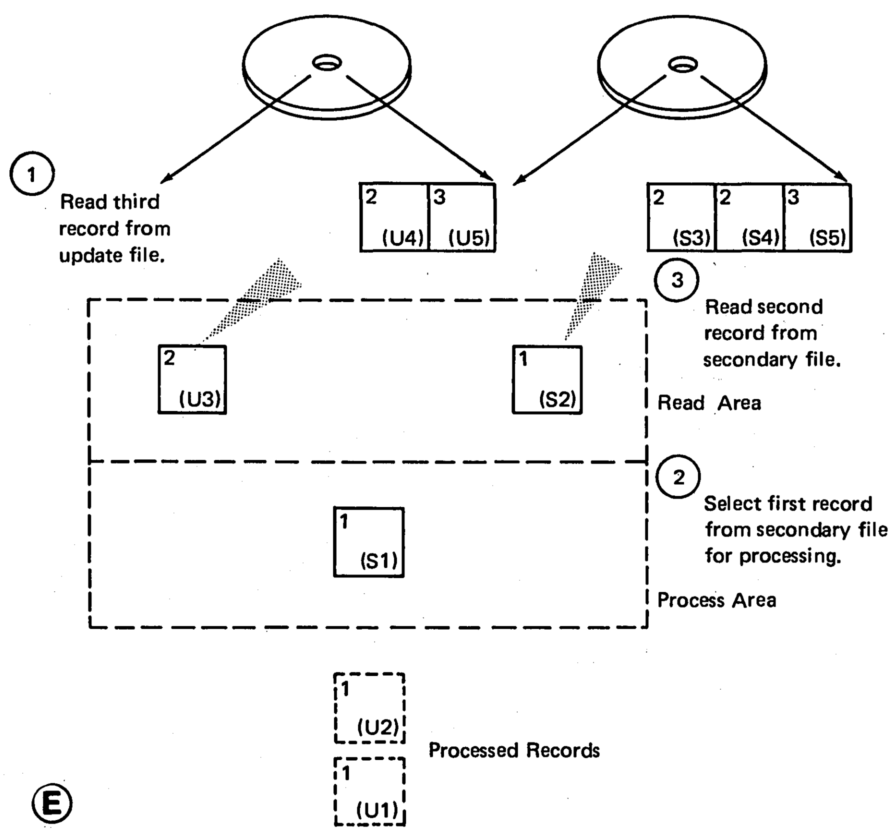


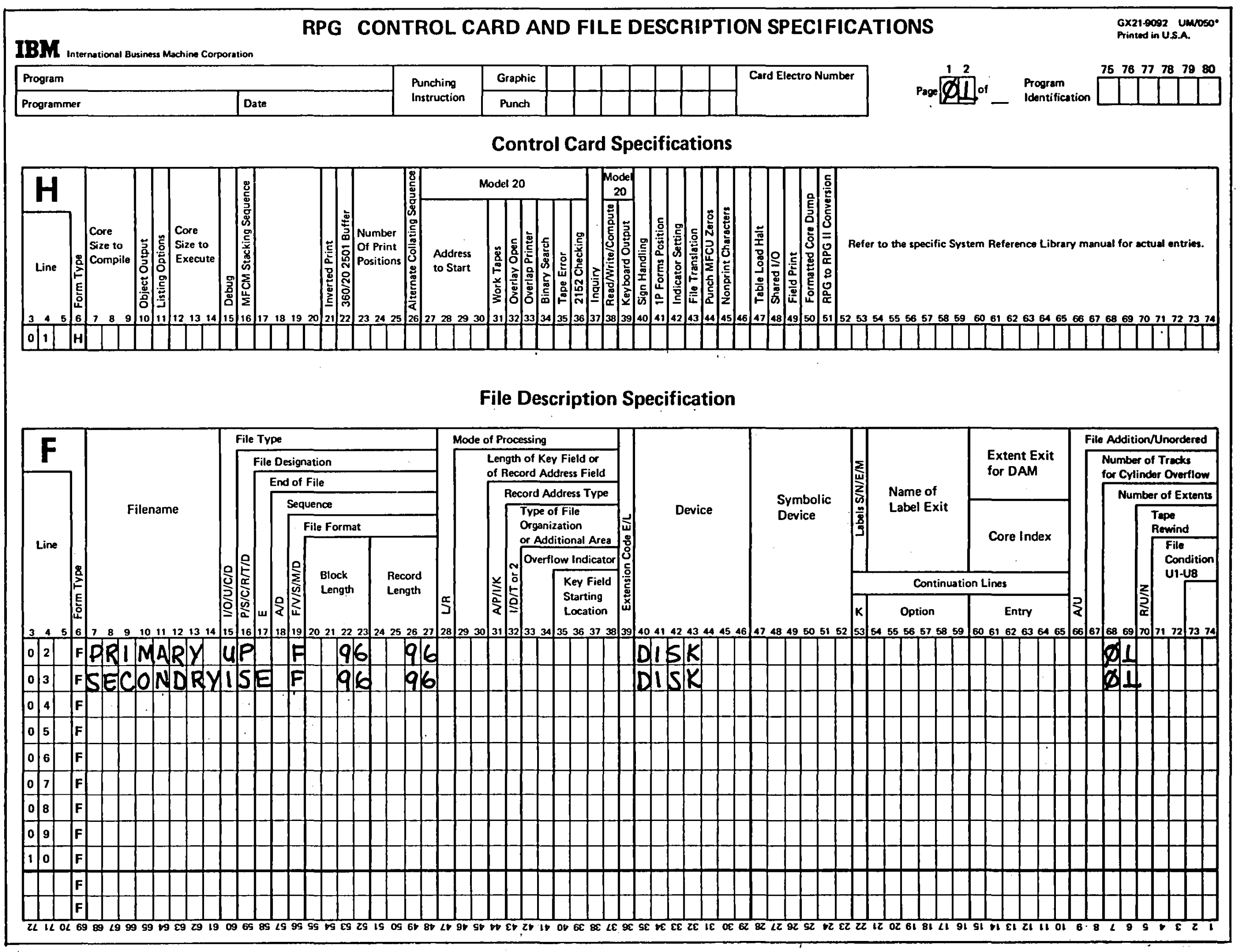
Figure 86 (Part 3 of 3). Available Records: One Input File, One Update File

Specifications

You can describe one set of look-ahead fields per file. The description applies to all records in the file, regardless of their type. Look-ahead fields must not be described for demand or chained files, and they must not be used as array fields. To describe a set of look-ahead fields, place ** in columns 19-20 of a line following the normal field descriptions for the file. The ** line can follow only a file or record type which has an alphabetic sequence entry. Leave columns 17-18 and 21-74 blank. Place any alphabetic characters under Sequence in columns 15-16. Describe the look-ahead fields on separate lines following the ** line (as in Figure 87, Part 2, insert B).

Figure 87 shows a job which reads records from two files. The primary file is named PRIMARY; the secondary file, SECONDRY. If a record from the primary file matches one from the secondary file, the information in positions one through ten of the secondary file record is placed in positions 31-40 of the primary file record. When there is no match, a *6* is placed in position 1 of the primary file record. The *6* will indicate an unmatched record in the primary file.

Because the primary file record is processed first when it matches a secondary file record, the information from the secondary file record has to be described as a look-ahead field.


RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9092 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Punching Instruction
Graphic
Punch
Card Electro Number
Page Ø1 of
Program Identification

Control Card Specifications

Line	Form Type
01	H

File Description Specification

Line	Form Type	Filename	File Type	File Designation	Sequence	File Format	Block Length	Record Length	Device	Number of Extents
02	F	PRIMARY	U	P		F	96	96	DISK	Ø1
03	F	SECONDRY	I	S	E	F	96	96	DISK	Ø1
04	F									
05	F									
06	F									
07	F									
08	F									
09	F									
10	F									
	F									
	F									

(A)

Figure 87 (Part 1 of 2). Look-Ahead Fields

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page 02 of | Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes	Field Location From	Field Location To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators
01	I	PRIMARY	AA			01									
02	I							1	10		NAME1				
03	I							11	14		MATCH		M1		
04	I	SECONDARY	AB			02									
05	I							1	10		NAME2				
06	I							11	14		MATCH		M1		
07	I		AC			**									
08	I							1	10		NXTNAM				
09	I														
10	I														
11	I														

Look-ahead field-field from secondary file records needed in primary file records.

Ⓑ

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page 03 of | Program Identification

Line	Form Type	Filename	Type (H/D/T/E)	Stacker #/Fetch (F)	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name *AUTO	Edit Codes	B/A/C/1-9/R	End Position in Output Record	P/B/L/R	Constant or Edit Word
01	O	PRIMARY	D						01 MR						
02	O									NXTNAM			40		
03	O		D						01NMR						
04	O												1		'6'
05	O														
06	O														
07	O														
08	O														

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Place the look-ahead field from secondary records into positions 31-40 of the primary record if the two records match.

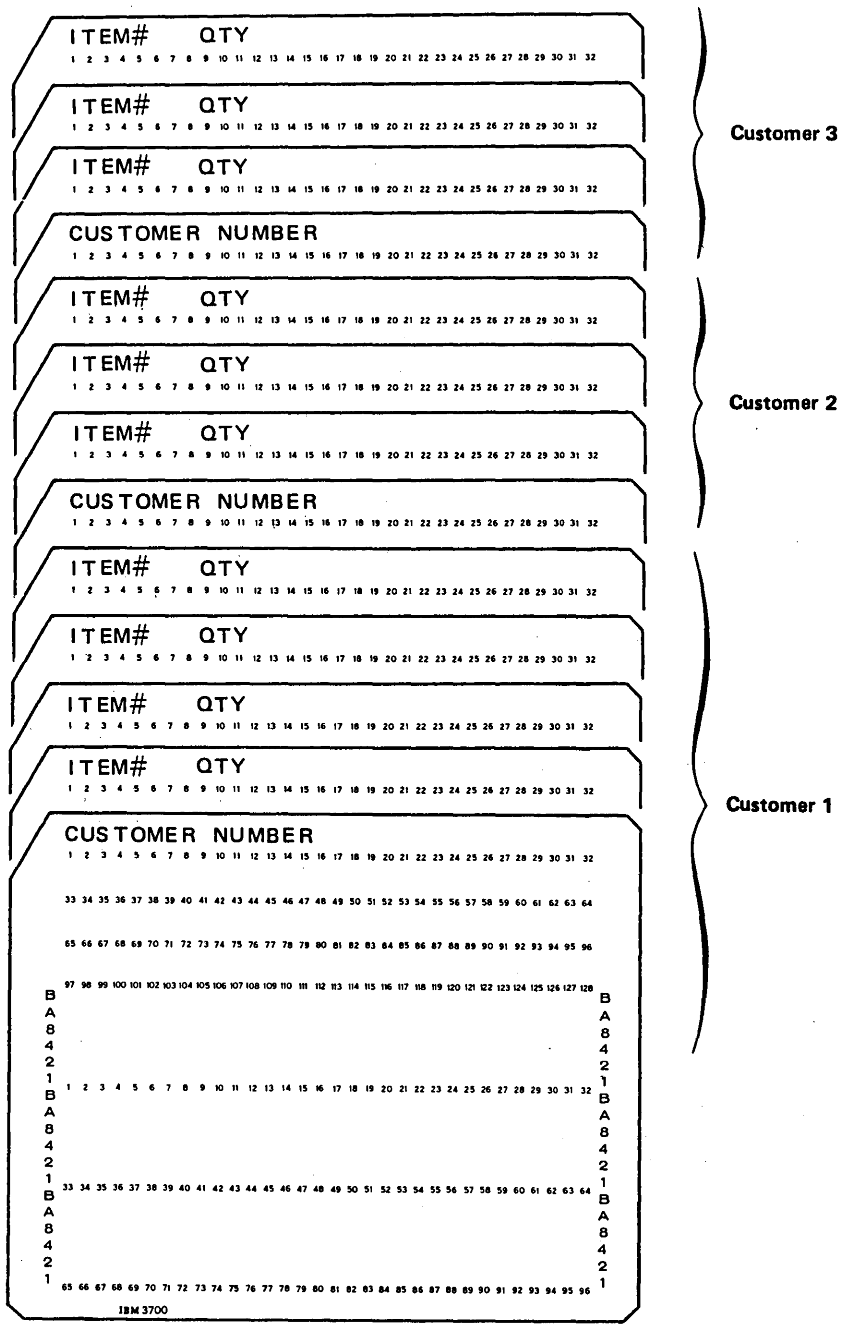
Place a 6 in position 1 of the primary record if the record matches no secondary record.

Ⓒ

Figure 87 (Part 2 of 2). Look-Ahead Fields

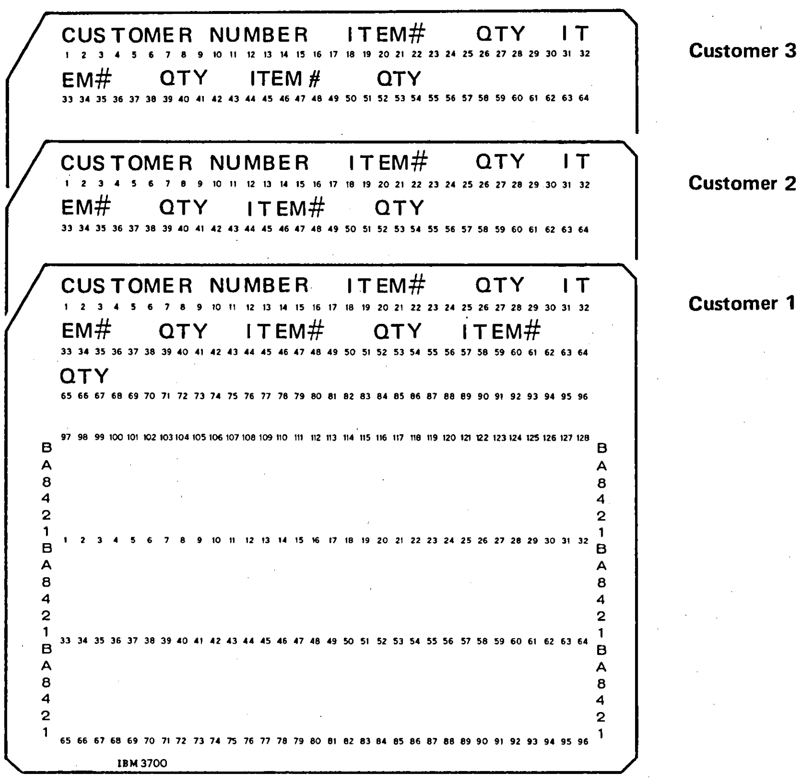
SPREAD CARDS

Certain jobs require that you keep data files containing a header card and a separate card for each item or transaction being recorded. Thus, for a billing job you may have a data file, for each customer, with the following cards.



With the spread card capability of RPG II, you can store more data on each card or diskette. You do not need to use a header card and a separate card for each item or transaction. You can specify a spread card with a header portion followed by trailer portions which contain the item or transaction data. A trailer portion can consist of as many fields as are necessary; however, the same fields must appear in each trailer portion. A trailer portion must not be split between two records.

Thus, a data file for a billing job such as the one shown previously may have the following spread cards.



Spread cards containing header and trailer portions. CUSTOMER NUMBER is the header portion; each set of ITEM# and QTY fields is a trailer portion.

Specifications

The only time you can specify spread cards is when the input card files are designated as primary or secondary. No look ahead fields can be described for spread cards. You can describe a maximum of 128 valid TR lines (TR in columns 19-20) in a program.

Specify spread cards as follows:

1. Describe the fields in the header portion of the spread card on separate specification lines immediately following the proper file and record type entries. The header is considered to be all positions up to the first trailer in the record. Any record identification codes specified for the header/trailer record must be contained within the header portion of the record. If a numeric entry is made in columns 15-16 of the specifications line containing the file and record type entries, an N must be entered in column 17 of the same line.

   Describe each field in the header portion as you would any normal RPG II field. You are required to describe only those fields in the header portion that are used later in the program. If no field in the header portion is used, you can omit the header field specification and specify the TR line immediately following the file and record type entries.

2. Enter TR in columns 19-20 of a specification line to indicate that the fields in the first trailer portion are described in the specification lines that follow. Leave columns 17-18 and 21-74 of the TR line blank.

3. Describe the fields in the first trailer portion on separate lines immediately following the TR line. Leave columns 7-43, 59-62, and 71-74 of the trailer specifications blank. Describe the fields in the first trailer portion as you would any normal RPG II field.

   You are required to describe only those fields in the first trailer portion that are used later in the program. Be sure, however, that you describe the fields that indicate the start and end position of the first trailer portion.

   Since all trailer portions must be the same length and must include the same fields, you need only describe the first one. The compiler uses this trailer specification to calculate how many trailer portions the record contains and to determine the start and end position of each.

Processing Spread Cards

Note: Spread cards cannot be specified for device-independent input files.

The following considerations apply when processing spread cards:

1. One trailer portion from a spread card is processed per program cycle. The system treats that trailer portion, along with its associated header portion, as one logical record.

2. The next spread card is read when:

   a. the system has processed all trailer portions in the current record.

   b. the system encounters a trailer portion in the card being processed which is entirely blank.

Example: The following input specifications are needed to describe a file (CARDIN) containing spread cards. Each card in the file contains a header portion in positions 3-8 which is made up of the customer number field (CUSTNO). The header is followed by a number of trailer portions. Each trailer is made up of an item number field (ITMNO), a field indicating the size of the item (SIZE), and a quantity field (QTY).

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	Record Identification Codes 2: Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	Record Identification Codes 3: Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	P/B/L/R (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)	(71-74)
01	I	CARDIN	Ø1	N		Ø1	1		C	A	2		C	B																	
02	I																				3	8		CUSTNO							
03	I					TR																									
04	I																				13	18		ITMNO							
05	I																				19	2Ø		SIZE							
06	I																				21	26	Ø	QTY							
07	I																														
08	I																														
09	I																														
10	I																														
11	I																														
12	I																														
13	I																														
14	I																														
15	I																														
16	I																														
17	I																														
18	I																														
19	I																														
20	I																														
	I																														
	I																														
	I																														
	I																														
	I																														

COLUMNS 21-41 (RECORD IDENTIFICATION CODES)

Use columns 21-41 to describe the information that identifies a record type.

When you have many record types in one file, you often want to perform different operations for each type. Therefore, you must identify each type by giving each a special code consisting of a combination of characters in certain positions in the record. This code must be described in columns 21-41 so that when a record is read the record type can be determined by these specifications. The first record identifying character should be identified in columns 21-27, the second in columns 28-34, and so forth.

When more than one record type is used in a file, only one record type will be selected for processing in each cycle. The record identifying indicator for that record type will be turned on at the time of selection. If a data record meets the requirements of more than one of the record types, it will belong to the first record type for which it qualifies. When all records are to be processed alike regardless of their type, or if there is only one type, leave columns 21-41 blank.

Position

Entry	*Explanation*
Blank	No record identification code is needed.
1-9999	Record position of the record identification code.

Use columns 21-24, 28-31, and 35-38 to give the location in the record of every character in the identification code. Entries in these columns must end in columns 24, 31, and 38 respectively. Leading zeros can be omitted.

Not (N)

Entry	*Explanation*
Blank	Record ID code is present in the specified column.
N	Record ID code is not present in the specified column.

Use columns 25, 32, and 39 to indicate that a certain character should not be present in the specified position.

C/Z/D

Entry	*Explanation*
C	Entire character.
Z	Zone portion of character.
D	Digit portion of character.

Use columns 26, 33, and 40 to indicate what portion of a character is used as part of the record identifying code (see *Character Structure* following *Examples*). Only the zone portion, only the digit portion, or both portions (the whole character) may be used (see *Examples, Example 3,* and *Example 4*). When establishing record identifying codes, remember that many characters have either the same zone or the same digit portion. For a list of characters that have identical zone or digit portions see *Appendix D,* Table D-4.

Character

Use any alphabetic character, special character, or digit in columns 27, 34, and 41 to identify the character that was used in the record to serve as the code or part of the code.

Note: If none of the identifying codes you have specified is found on a record, processing stops. You may continue, however, by pressing START on the processing unit. The record that caused the halt is not processed, and the next record in that file is read.

AND Relationship

A maximum of three identifying characters may be described in one specification line. Thus, if the identification code consists of more than three characters, an AND line must be used. This means that the first three identifying characters are described in the first line. The additional identifying characters are described in as many following lines as are needed. Write the word AND in columns 14-16 to indicate an AND line (see *Examples, Example 1*).

You may specify up to 20 AND or OR lines in any combination to describe the record identifying code. The record must contain all the characters indicated as its record identification code before the record identifying indicator will turn on. In an AND relationship, all conditions must be met to determine the record type.

OR Relationship

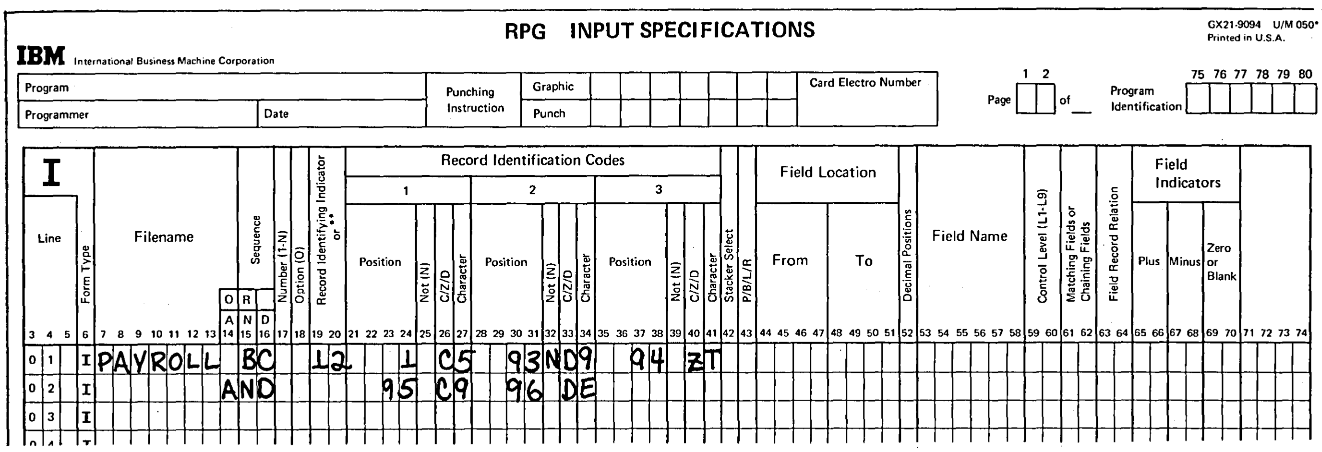
A particular record type may be identified by two different codes. If this is the case, OR lines must be used to indicate that either one of the codes may be present to identify the record. Write the word OR in columns 14-15 to indicate an OR line (see *Examples, Example 2*). A maximum of 20 AND or OR lines in any combination are allowed in any record identification line.

Seven columns are set aside for the description of one character in the record identification code. Each specification line contains three sets of seven columns: columns 21-27, 28-34, and 35-41. Each set consists of 4 fields: Position, Not, C/Z/D, and Character. Coding is the same for all three sets.

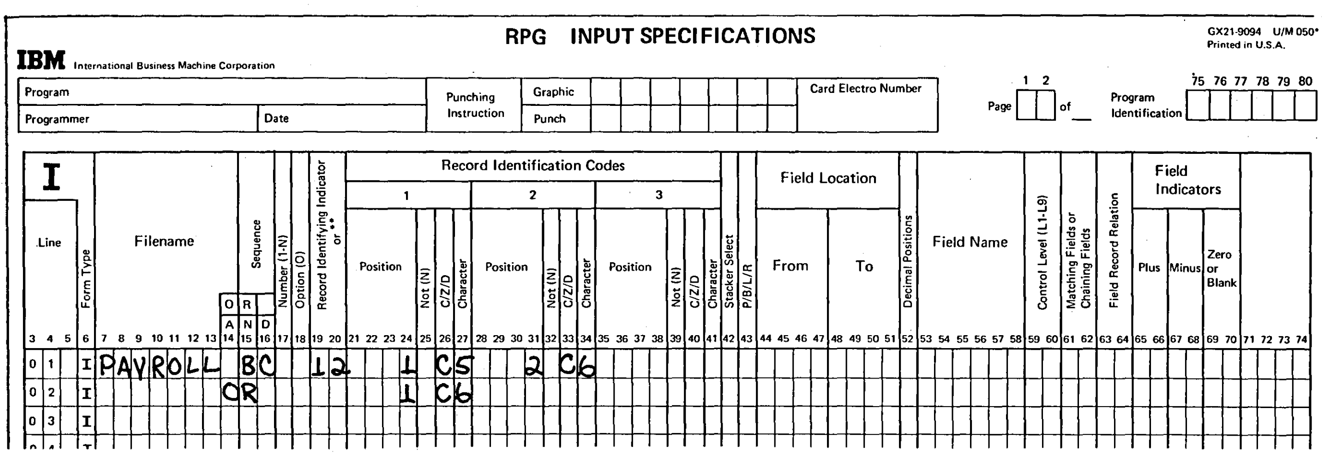
Examples

Example 1: Figure 88, insert A shows a record identification code consisting of five characters. The first character is located in position 1, the other four record ID tests are made in positions 93, 94, 95, and 96. Since only three identifying characters may be described on one line, the word AND must be used on the next line to indicate that the last two characters of the code are part of the preceding record identification entries.

Example 2: Figure 88, insert B shows the use of an OR line to describe record type identification codes. The record used with record identifying indicator 12 can be identified by two different codes. The record can be identified by a code consisting of a *5* in position 1 and a *6* in position 2 or a code consisting of a *6* in position 1.



A



B

Figure 88. Record Identification Codes

Example 3: In Figure 88, insert A, the entry in column 32 indicates that the digit 9 must not be present in position 93 for the records in the file.

Example 4: Figure 88, insert A shows that only the zone portion of the character *T* located in position 94 is part of the identifying code. In position 96 only the digit portion of the character *E* is part of the code.

CHARACTER STRUCTURE

Every alphabetic character, numeric character, or special character is represented by different combinations of punches in the 80- or 96-column card. Each character punched on the card is composed of two parts, a zone portion and a digit portion. Even after a character has been read into the machine, it is still composed of these two parts. *Appendix D*, Table D-4 shows grouping of characters by equal zones and equal digits for the 96-column card code. Refer to that table while you read the following paragraphs.

A character is represented in the computer by eight magnetic bits. Because the character is represented by six punch positions on a card, translation has to take place so that it can be represented by eight bits in storage. This is an automatic function. As a result of it, however, the way characters are represented in the machine and the way they appear on the punched card are not always identical. Not all characters having a *B* zone punched in the card have identical zone structures in the machine.

Whenever you use just the zone or just the digit portions of characters in specific functions such as sequencing, testing, or identifying records, you must keep in mind the exact structure of the characters when represented in the machine. For example, when you are identifying a record type on the basis of the zone portion of character *D*, you must remember that several characters have the same zone structure as the letter *D*. If a card with the record identifying code of *E* is read, it is still considered to be a *D* type record because the zone of character *E* is the same as the zone of character D.

Note: Characters with the same zone punch in the card do not necessarily have the same representation in the machine. For instance, character $ has the same zone punch in the card as character *K*. However, they do *not* have the same zone representation in the machine.

All characters can be arranged in a certain order according to the way their zone and digit portions are represented in the machine. This means that if you are to sequence the characters, each character has a special position in relation to all others on the basis of its representation in the machine. This special order or positioning is known as the collating sequence (see *Column 26, Alternate Collating Sequence* in Chapter 3). The characters can also be arranged in a special order on the basis of just the zone portion or just the digit portion. Each type of sequencing, whether according to zone, digit, or the entire character, results in a different arrangement of the characters. The standard sequence order of the characters, when both zone and digit portions are used to sequence, is shown in *Appendix D*, Table D-5. When using only the digit or only the zone portion of the character to sequence the characters, remember that often characters have the same zone or the same digit portion. Thus they each rightly belong in the same position. The only thing that then determines their position is the order in which they are read into the machine.

Use Table D-4 in *Appendix D* to determine which characters have identical zone and digit portions. All characters in each group have either the same zone or the same digit portions (depending on the figure). The groups are arranged from low to high according to the collating sequence supported by RPG II.

Structure of Negative Numbers

Negative numbers have a different character structure than positive numbers because negative numbers are formed by punching a minus sign over the number. Numbers 0-9 have only digit portions. However, a minus sign is a B zone punch. Thus when the zone punch (minus sign) and the digit punch (0-9) are put together, a different character is formed. Therefore, negative numbers are represented in the machine by the characters J-R. Negative zeros are converted to positive zeros in the system.

COLUMN 42 (STACKER SELECT)

Entry	*Explanation*
Blank	Cards automatically fall into a predetermined stacker.
1-2	Stacker into which the card type is stacked for 1442.
1-4	Stacker into which the card type is stacked for 5424 MFCU or 2560 MFCM Model A2.
1-5	Stacker into which card type is stacked for 2560 MFCM Model A1.

Column 42 is used to indicate that certain types of input cards must be stacked in a specific stacker. If you make no entry, all cards will go into a predetermined stacker as follows:

	Primary Hopper	*Secondary Hopper*
MFCU	1	4
1442	1	N/A
Model 15 only		
MFCM (Model A1)	1	5
MFCM (Model A2)	1	4

Only input file and combined file cards may be stacker selected in the input specifications.

You may stacker select cards from the input file in input specifications only. However, cards from a combined file may be stacker selected in either input specifications or output specifications (see *Column 16* in Chapter 9).

Any card type that is stacker selected on the input specifications should not have an output operation specified for it. If an output operation is specified, however, the input stacker selection specification is overridden (see *Column 16* in Chapter 9) if the output is performed.

When the same stacker is used for both input (or combined) and output files, a card from the output file is put in the stacker before a card from the input or combined file. This procedure is reversed (input or combined card before output card) if Look Ahead Fields or dual I/O areas are specified for the input file (a stacker select specification may not be made for input files with dual I/O areas).

The card type in an OR line may be selected for a special stacker by an entry in column 42. If the card type in an OR line has no entry in column 42, the card goes into the predetermined stacker. (See *Columns 53-58* in this chapter for more information on OR lines.) AND lines may not have an entry in stacker select.

COLUMN 43 (PACKED OR BINARY FIELD)

Entry	*Explanation*
Blank	Field is in unpacked decimal format, or is alphameric.
P	Field is in packed decimal format.
B	Field is in binary format.

Column 43 is used to indicate that a numeric field is in packed decimal or binary format. Numeric data fields in packed decimal or binary format must be converted to the unpacked decimal format before they can be processed. This conversion ignores decimal points.

Column 43 must contain a *P* if the input field named in columns 53-58 is in packed decimal format. Column 43 must contain a *B* if the input field named in columns 53-58 is in binary format.

Any array which was read in packed or binary format should have an entry in column 43 of the Input Sheet. In this case, the From and To columns in the Input Sheet should define the positions the array occupies in the record in the packed or binary format. The unpacked decimal length of each array element is defined on the Extension Sheet.

Note: When using whole arrays, treat each array element as a field.

Unpacked Decimal Format

Unpacked decimal format means that each byte of storage, can contain one character. (That character may be a decimal number or it may be an alphabetic or special character.) In the unpacked decimal format, each byte of storage is divided into a 4-bit zone portion and a 4-bit digit portion. Figure 89 shows the unpacked decimal format.

Note: RPG II does not perform data verification on numeric data. The value of the digit portion of a character is assumed to be the numeric value of that character.

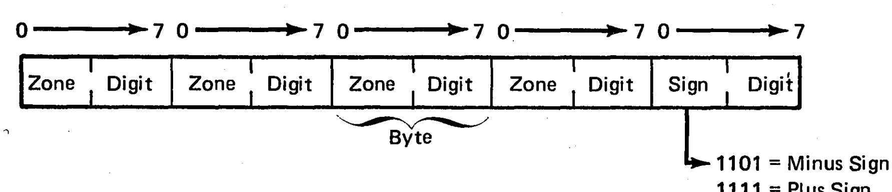


Figure 89. Unpacked Decimal Format

The zone portion of the rightmost byte indicates whether the decimal number is positive or negative. In unpacked decimal format, the zone portion is included for each digit in a decimal number; however, only the zone over the rightmost digit serves as the sign. Figure 90 shows the unpacked decimal format for decimal number 8, 191.

Once data has been read into the computer, it must be represented in unpacked decimal format before it can be processed. Thus, it is perfectly correct to read data into the computer in the unpacked decimal format. This eliminates converting the input data since it is already in the required format.

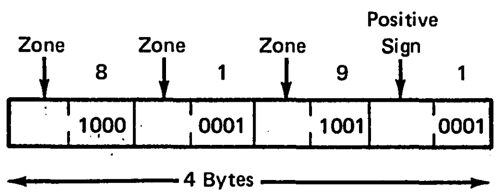


Figure 90. Unpacked Format of Decimal Number 8, 191

Packed Decimal Format (P)

Packed decimal format means that a byte of storage can contain two decimal numbers. This format allows you to get almost twice as much data into a byte as you can using the unpacked decimal format.

In the packed decimal format, each byte of storage, except the rightmost byte, is divided into two 4-bit digit portions. The rightmost portion of the rightmost byte contains the sign (plus or minus) for that field. Figure 91 shows packed decimal format.

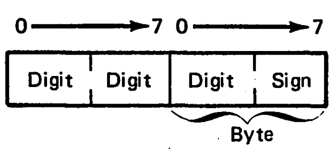


Figure 91. Packed Decimal Format

The sign portion of the rightmost byte is used to indicate whether the numeric value represented in the digit portions is positive or negative. In the packed decimal format, the sign is included for each decimal number; the zone portion is not given for each digit in the number. Compare how the decimal number 8,191 is represented in packed decimal format (Figure 92) with its unpacked representation (Figure 90).

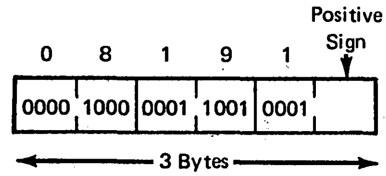


Figure 92. Packed Format of Decimal 8, 191

Since data must be represented in unpacked decimal format once it is inside the computer, you must give the RPG II program an indication when input fields are in a different format. A *P* in column 43 indicates that the input field is in the packed decimal format and that the system must convert this field to the required unpacked format.

Packed key fields can be up to 8 bytes long. The following chart shows the packed equivalents for unpacked fields up to 15 bytes long:

Unpacked Length in Bytes	Packed Length in Bytes
15 14	8
13 12	7
11 10	6
9 8	5
7 6	4
5 4	3
3 2	2
1	1

Binary Format

Binary format means that two bytes of storage can contain up to four decimal numbers, and that four bytes of storage can contain up to nine decimal numbers. In the binary format, each field must be either two or four bytes long.

Each 2-byte binary field consists of a 1-bit sign followed by a 15-bit numeric value. In binary format, a decimal number as high as 9,999 requires only two bytes of storage. For each 2-byte binary field stored, the system automatically sets aside four bytes of storage to accommodate the field when it is converted to unpacked format. Figure 93 shows a 2-byte field in binary format.

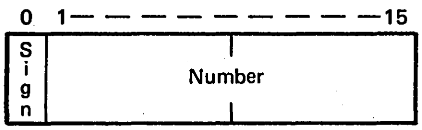


Figure 93. Two-Byte Field in Binary Format

Each 4-byte binary field consists of a 1-bit sign followed by a 31-bit numeric value. In binary format, a decimal number as high as 999,999,999 requires only four bytes of storage. For each 4-byte binary field the system automatically sets aside nine bytes of core storage to accommodate the field when it is unpacked. Figure 94 shows a 4-byte field in binary format.

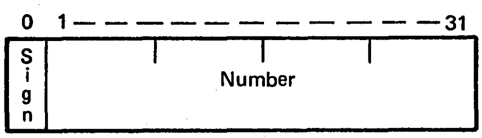


Figure 94. Four-Byte Field in Binary Format

Binary fields containing values greater than decimal 9,999 (4-byte decimal field) or 999,999,999 (9-byte decimal field) cannot be converted into 4-byte or 9-byte decimal fields without loss of data. High order (leftmost) digits of decimal numbers longer than four or nine digits are lost in such cases.

The following devices support packed and/or binary data:

- Disk (5444, 5445, or 3340)
- Tape
- MFCM
- 1442
- 2501
- 3741 (directly attached)

Note: For 80-column card devices, any of the 256 EBCDIC characters may be read or punched; column binary is not supported. For a directly attached 3741, any of the 256 EBCDIC characters may be read or written. The B or P entry in column 43 of the Input Specifications can apply to fields from only 80-column cards (not 96-column cards).

In both 2-byte and 4-byte binary fields, the sign bit indicates whether the numeric value is positive (sign bit is off) or negative (sign bit is on). Notice that in binary format the zone portion of the decimal number is not included. Compare the binary format of the number 8,191 (Figure 95) with its packed and unpacked representation (Figures 90 and 92). Figure 96 shows the binary format of -8,191. Note that the sign bit is *on* (negative number). The same procedure shown in Figure 96 can be used to convert any negative binary field to decimal.

Since data must be represented in unpacked decimal format once it is inside the computer, you must give the RPG II program an indication of when input fields are in another format. A *B* in column 43 indicates that the input field is in the binary format and that the system must convert this field to the required unpacked format.

COLUMNS 44-51 (FIELD LOCATION)

Entry	*Explanation*
Two 1-4 digit numbers	Beginning of a field (From) and end of a field (To).

Use columns 44-51 (From and To) to describe the location on the record of each field containing input data named in columns 53-58 (Field Name). Enter the number of the record position in which the field begins in columns 44-47. Enter the number of the record position in which the field ends in columns 48-51.

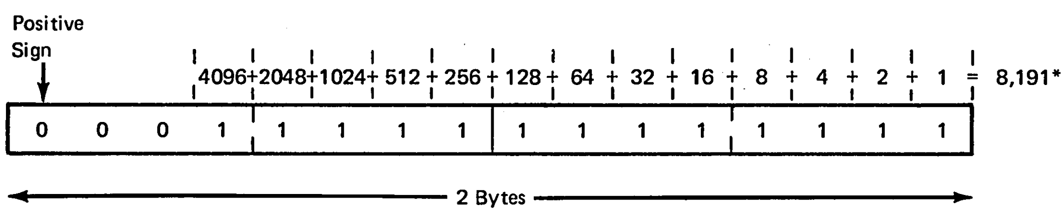
A single position field is defined by putting the same number in both From (columns 44-47) and To (columns 48-51). If a field of more than one position is defined, the number entered in From (columns 44-47) must be smaller than the number entered in To (columns 48-51).

It is not necessary that the From and To columns specify a whole array. A portion of an array may be read in; however, the array will be read in from element 1 up to as many elements as will fit in the numbers specified in the From and To columns.

The maximum field length for a numeric field is 15 positions (eight if packed, four if binary). The maximum field length for an alphameric field is 256 characters.

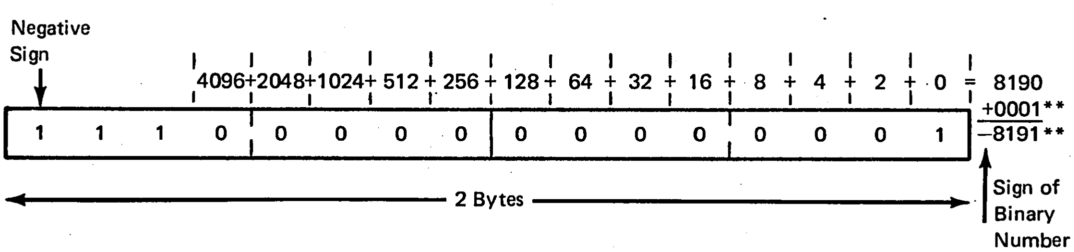
Entries in these columns must end in columns 47 and 51. Leading zeros may be omitted.

Input CRT files display data on the CRT in positions 121-240; however, when specifying the FROM and TO locations, positions 1-120 must be specified.



* The numeric value of a positive binary field is obtained by adding the values of the bits that are on (represented as 1's). The sign bit is not included in the addition.

Figure 95. Binary Format of Decimal Number 8, 191



** The numeric value of a negative binary field is obtained by adding the values of the bits that are off (represented as 0's), plus one. The sign bit is not included in the addition.

Figure 96. Binary Format of Decimal Number 8, 191

COLUMN 52 (DECIMAL POSITION)

Entry	*Explanation*
Blank	Alphameric field.
0-9	Number of decimal positions in numeric field.

Use column 52 to indicate the number of positions to the right of the decimal in any numeric field named in columns 53-58. Column 52 must always have an entry when the field named in columns 53-58 is numeric. If you wish to define a field as numeric with no decimal position, enter a *0*. If a field is to be used in arithmetic operations or is to be edited, it must be numeric. The number of decimal positions must be less than or equal to the field length.

COLUMNS 53-58 (FIELD NAME)

Entry	*Explanation*
Valid RPG II field name	Field name, array name, or array element
PAGE	
PAGE1	Special words
PAGE2	

Use columns 53-58 to name a field, array, or array element found on your input records. If you are referencing an array, additional entries may be needed in these columns (see *Tables and Arrays* in Chapter 5). Use this name throughout the program whenever you refer to this field. You must indicate the names of the fields for all types of records. However, you should name only the fields that you use.

Field Names

A field name can be from one to six characters long, must begin in column 53, and must be a valid RPG II name.

All fields in one type of record should have different names. If two or more fields on the same record type have the same name, only the field described last is used. However, fields from different record types may have the same name if the fields are the same length and contain the same type of data. This applies even if the fields are found in different locations in each record type.

Fields which are read in from a card are limited to the length of one punched card.

Fields that are used in arithmetic operations or fields that are edited or zero suppressed (see *Column 38* and *Columns 45-70* in Chapter 9) must be defined as numeric. This means that column 52 must have a decimal position entry.

A separate line is used for each field description.

Field Names in OR Relationship

Even though two or more record types contain identical fields, you must describe each field. This may require duplicate coding. To eliminate duplicate coding of identical fields from different record types, you may use the OR relationship. A maximum of twenty OR or mixed AND and OR lines can be used for each record sequence group.

An OR relationship means that the fields named may be found in either one of the record types. You may use OR lines when:

1. Two or more record types have the same fields in the same positions (see *Example*).

2. Two or more record types have some fields which are identical and some fields which differ in location, length, or type of data. See *Columns 63-64* in this chapter for sample coding of such record types.

Write the word OR in columns 14 and 15 to indicate an OR line (see *Example*). If there are several AND or OR lines, field description lines start after the last record identification line.

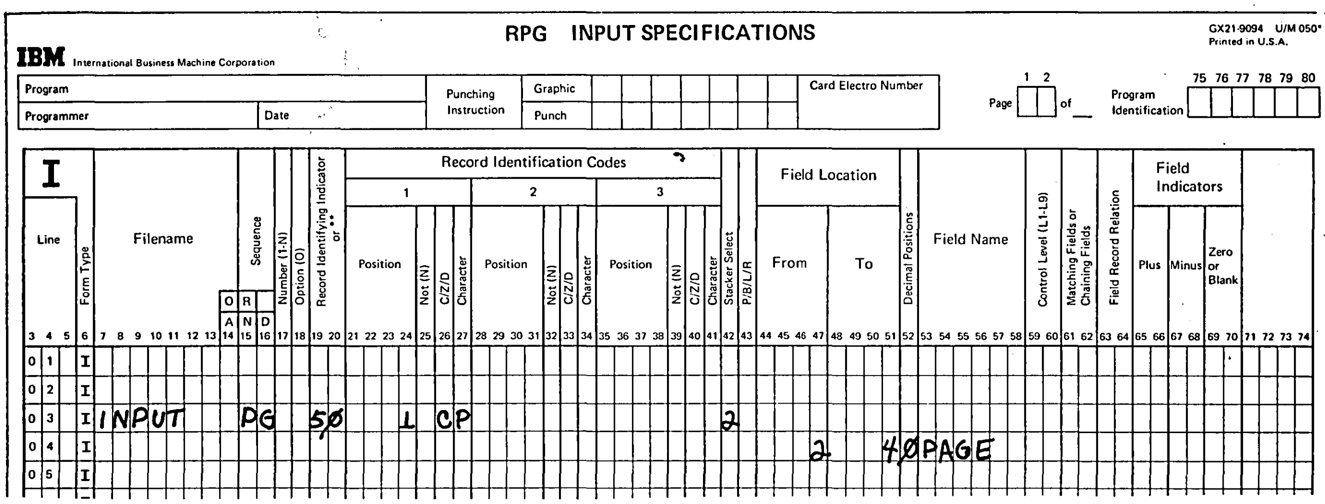
Special Words (PAGE, PAGE1, PAGE2)

If your printed report has several pages, you may want to number the pages. The special word PAGE allows you to indicate that page numbering is to be done. When you use a PAGE entry on the Output-Sheet, page numbering automatically starts with 1 (see *Columns 32-37* in Chapter 9).

If you want to start at a page number other than 1, you can enter that page number in a field of an input record and name that field PAGE in columns 53-58. The number you enter in the PAGE field of the input record should be one number less than the starting page number. If your numbering should start with 24, enter a 23 in the PAGE field. The PAGE field can be of any length (up to 15 positions), but must have zero decimal positions specified (Figure 97). Any entry you make in the PAGE field should be right justified, such as 0023.

Page numbering can be restarted during a program run by entering a number in a PAGE field of any input record. The PAGE field can be defined and used in calculations like any other field.

The three possible PAGE entries: PAGE, PAGE1, and PAGE2 are provided for naming different output files. Care must be taken when using the same entry for two different output files.



IBM International Business Machine Corporation — RPG INPUT SPECIFICATIONS — GX21-9094 U/M 050* Printed in U.S.A.

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator	Position	Not (N)	C/Z/D	Character	Stacker Select	From	To	Decimal Positions	Field Name
01	I														
02	I														
03	I	INPUT	PG			5Ø	1		C	P	2				
04	I											2	4	Ø	PAGE
05	I														

Figure 97. Page Record Description

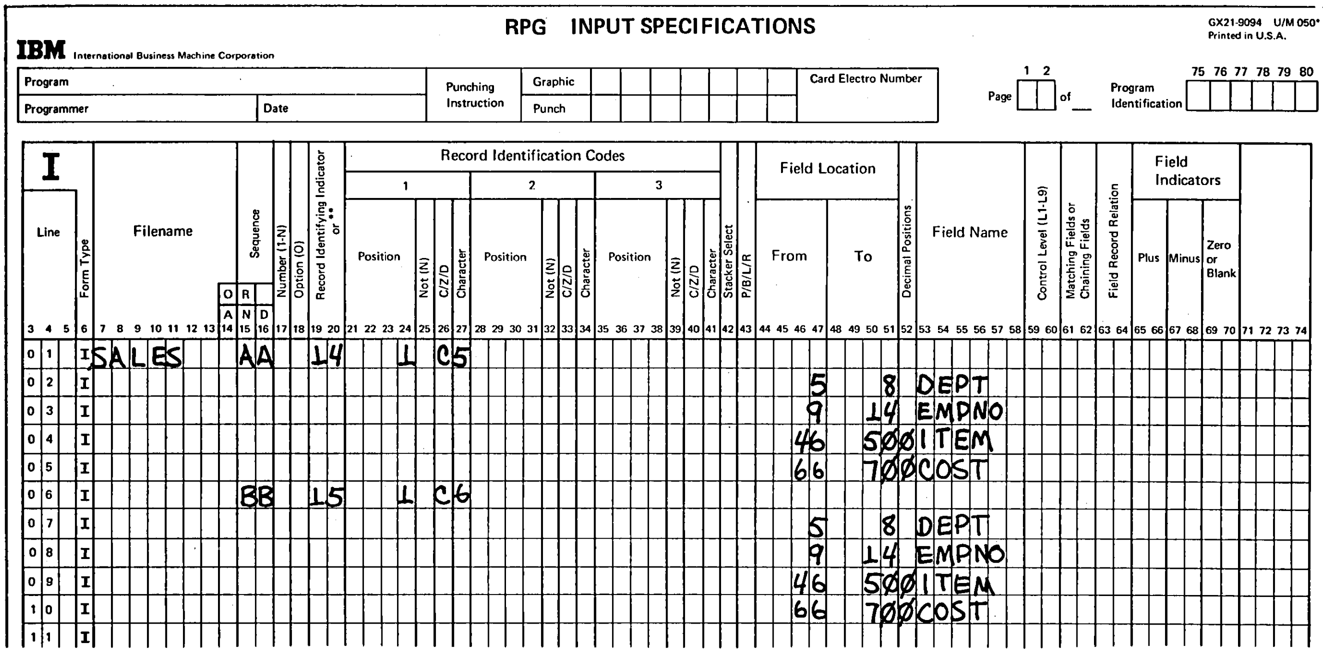
Example

Figure 98 shows how the use of OR lines can save duplicate coding. The two different record types (one identified by a *5* in column 1, the other by a *6* in column 1) both have identical fields which must be described. Figure 98, insert B shows the use of OR lines to do the same thing with less coding. The coding in Figure 98, insert B says that all four fields can be found on either the record type identified by the *5* in column 1 or the record type with a *6* in column 1.

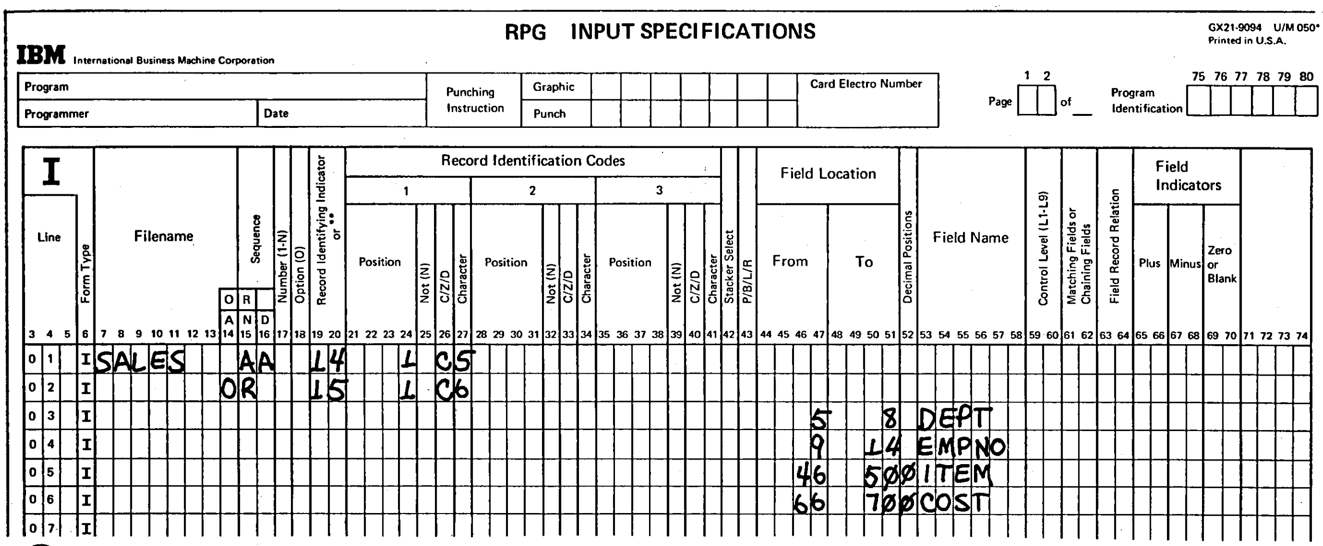
COLUMNS 59-60 (CONTROL LEVEL)

Entry	*Explanation*
L1-L9	Any control level indicator.

Use columns 59-60 to assign control level indicators to input fields. (Control level indicators may not be associated with a chained or demand file.) Control level indicators are used to specify the point at which specified operations are to be done. You may assign a control level



Ⓐ



Ⓑ

Figure 98. Record Types with Identical Fields

indicator to any field except a binary field. This field is then known as a *control field* and is checked for a change in information. When information in the control field changes, a *control break* occurs. All records having the same information in the control field are known as a *control group.*

Whenever a record containing a control field is selected, the data in the control field is compared with data in the same control field from the previously selected record. When a control break occurs, the control level indicator turns on. Operations conditioned by the control level indicators are then done (see *Columns 7-8* and *Columns 9-17* in Chapter 8 or *Columns 23-31* in Chapter 9).

L1-L9 (Control Level Indicators)

Control level indicators are used to signal when a change in a control field has occurred. Because they turn on when the information in a control field changes, they may be used to condition operations (such as finding totals) that are to be performed only when all records having the same information in the control field have been read. They may also be used to do total printing or to condition operations that are to be done on only the first record in a control group. Control level indicators always turn on after the first record of a control group is read.

The indicators are ranked in order of importance with larger numbers ranking higher than lower numbers. L4 has a higher rank than L1. All lower ranked indicators turn on when a higher level indicator turns on. For example, if an L8 control break occurs, L1-L7 also turn on. The importance of a control field in relation to others should determine how you assign indicators. For example, the type of data which demands a subtotal has a lower control level indicator than data which needs a grand total. A field containing department numbers is given a higher control level indicator than a field containing employee numbers (see *Examples, Example 1*).

Control level indicator L0, since it is always on, cannot be assigned to a control field. Nevertheless, you may use it to condition operations (see *Columns 7-8* in Chapter 8). Normally, control level indicators are used to:

1. Condition certain calculations to be performed when the information in the control field changes.

2. Condition certain punching (summary punching) or printing (total printing) to be done after totals have been accumulated for one control group.

3. Condition certain operations to be done on the record that causes a change in a control field (first record of a new control group).

Control level indicators may be used in input, calculation, and output specifications.

A control level indicator may be turned on or off by operation codes SETON and SETOF and may be used as record identifying indicators. However, not all control level indicators lower than the one specified are turned on or off in these cases. For example, when L2 is set on, L1 does not automatically turn on.

Using Control Fields

When using control fields, remember:

1. If the same control level indicator is used in different record types or in different files, the control fields associated with that control level indicator must be the same length and same type (alphabetic or numeric). See *Examples, Example 2.*

2. In the same record type, record columns in control fields assigned different control level indicators may overlap (Figure 99). However, the total number of columns assigned as control fields (counting each control level only once) must not be greater than 144. In Figure 99 for example, a total of 15 columns is assigned to control levels.

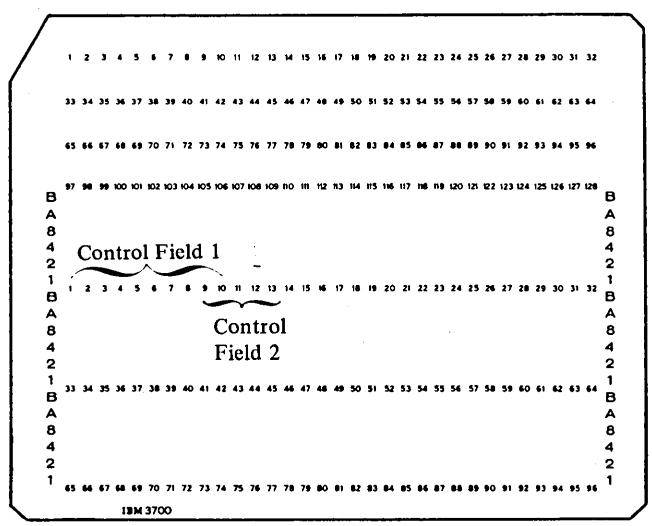


Figure 99. Overlapping Control Fields

3. Field names are ignored in control level operations. Therefore, fields from different record types which have been assigned the same control level indicator may have the same name.

4. Control levels need not be written in any sequence. L2 entry can appear before L1. Also, there may be gaps in the control levels assigned.

5. When numeric control fields with decimal positions are compared to see if a control break has occurred, they are always treated as if they have no decimal positions. For example, 3.46 is considered equal to 346.

6. If a field is specified as numeric, only the digit portion is used to determine if a control break has occurred. This means that a field is always considered to be positive. A minus five is considered equal to a plus five.

7. All control fields given the same control level indicator are considered numeric if any one of those control fields is described as numeric (column 52 has an entry). This means that when numeric control fields are compared to see if the information has changed, only the digit portion of each character is compared.

8. Control fields are initialized to hexadecimal (logical) zeros or to the lowest alternate collating sequence value given.

9. A control break is highly probable after the first record containing a control field is read. The control fields in this record are compared to an area in storage which is void of any type of data. Since fields from two different records are not being compared, total calculations and total output operations are bypassed for the first record containing a control field.

10. If different record types in a file do not have the same number of control fields, unwanted control breaks may occur. See *Examples, Example 3* for a method of avoiding unwanted control breaks.

Split Control Fields

If a control field is made up of more than one field of a record, it is then known as a split control field. A split control field is created when the same indicator is assigned to two or more connected or unconnected fields on the same record type.

All fields in one record that have the same control level indicators are combined by the program in the order specified in the input specifications and treated as one control field (see *Examples, Example 4*). Some special rules for split control fields are:

1. For one control level indicator, a field may be split in some record types and not in others if the field names are different. However, the length of the field, whether split or not, must be the same in all record types.

2. The length of the portions of a split control field may vary for different record types if the field names are different. However, the total length of the portions must always be the same.

3. No other specification lines may come between lines which describe split control fields.

4. If one section of a split control field is numeric, the whole field is considered numeric.

5. A numeric split control field may have more than 15 characters if any one portion of the split field does not exceed 15 characters and the sum of all control fields (counting each control level only once) is not greater than 144 characters.

6. A split control field cannot be made up of a packed decimal field and an unpacked decimal field. Both portions of the control field must be packed, or both unpacked.

Note: Additional rules applying to control level indicators when used with indicators in the Field Record Relation columns are discussed in *Columns 63-64* in this chapter.

Examples

Example 1: Figure 100 shows the assignment of three indicators. The names of the control fields (DIVSON, DEPT, EMPLNO) give an indication of their relative importance. The division (DIVSON) is the most important group. It is given the highest control level indicator used (L3). The department (DEPT) ranks below the corporation; L2 is assigned to it. The employee field has the lowest control level indicator (L1) assigned. Note the overlap of control fields on lines 02 and 06.

Example 2: Figure 100 shows that the same control level indicators may be used for different record types. Notice, however, that the control fields having the same indicators are the same length. EMPLNO, in both cases, is 6 columns in length, DEPT is 4, and DIVSON is one.

IBM International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050*
Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Position (1)	Not (N)	C/Z/D	Character	Position (2)	Not (N)	C/Z/D	Character	Position (3)	Not (N)	C/Z/D	Character	Stacker Select	P/B/L/R	From	To	Decimal Positions	Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Plus	Minus	Zero or Blank
01	I	EMPLREP	AB			1Ø	1		C	A									2											
02	I																				5	1Ø	Ø	EMPLNO	L1					
03	I																				11	3Ø		NAME						
04	I																				8Ø	8Ø		DIVSON	L3					
05	I																				33	33		SHIFT						
06	I																				2	5		DEPT	L2					
07	I		CD			2Ø	1	N	C	A									3											
08	I																				5	1Ø	Ø	EMPLNO	L1					
09	I																				11	14		DEPT	L2					
10	I																				6Ø	7Ø		HRSWKD						
11	I																				8Ø	8Ø		DIVSON	L3					
12	I																													
13	I																													
14	I																													
15	I																													
16	I																													
17	I																													
18	I																													
19	I																													
20	I																													
	I																													
	I																													
	I																													
	I																													
	I																													

Figure 100. Control Level Indicators (Two Record Types)

Example 3: Different record types normally contain the same number of control fields. However, some applications require a different number of control fields in some records. This is shown in Figure 101, insert A. The salesman records contain only the L2 control field. The item records contain both L2 and L1 control fields.

With normal RPG II coding, an unwanted control break is created by the first item record following the salesman record. This is recognized by an L1 control break immediately following the salesman record and results in an asterisk being printed on the line below the salesman record (see Figure 101, insert B).

Figure 101, inserts C and D, contain excerpts from a program that processes the input shown in Figure 101, insert A, and prevents the unwanted control break from occurring. The corrected output produced is shown in Figure 101, insert B.

Line 01 of the Calculation sheet sets on indicator 11 when the salesman record is read. When the next item record causes an L1 control break, no total output is printed because indicator 11 is on (line 07 of Output Sheet). Detail calculations are then processed for the item record and line 02 of the Calculation Sheet sets indicator 11 off. This allows the normal L1 control break to occur.

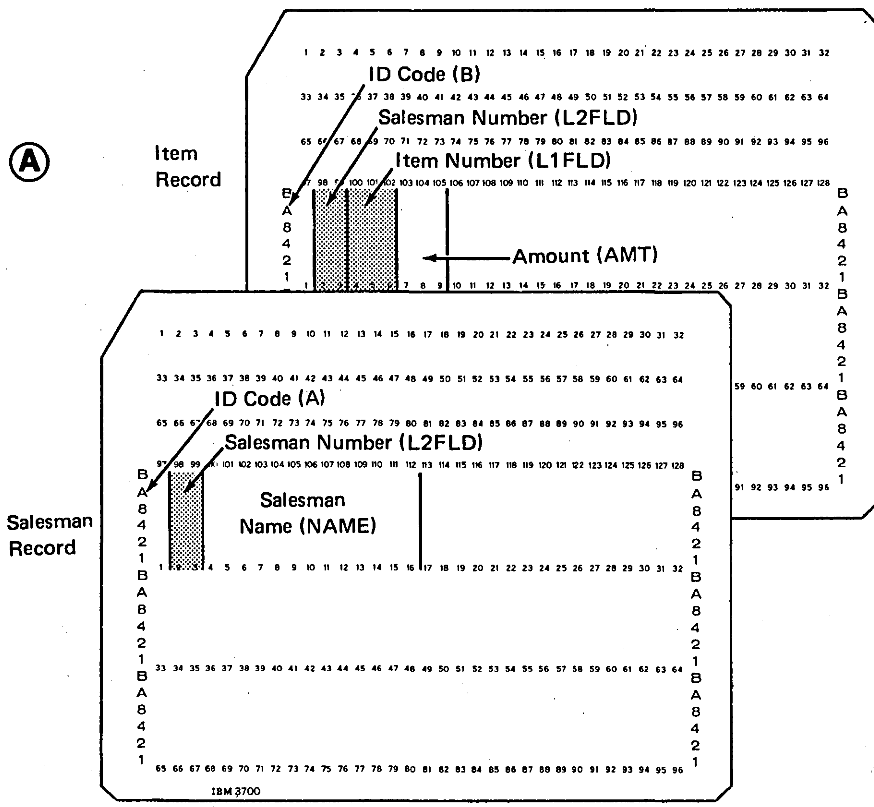


Figure 101 (Part 1 of 4). Unwanted Control Breaks

```
01     DICK LOVE
                    *  <---- Unwanted
                             Control
                             Break
            100   3
            100   2
                  5 *
            101   4
                  4 *
                  9 **

02     CAL WINBUSH
                    *  <---- Unwanted
                             Control
                             Break
            100   6
            100   2
                  8 *
            101   3
                  3 *
                 11 **

                 20
```

Output Showing Unwanted Control Level Break

```
01     DICK LOVE
            100   3
            100   2
                  5 *
            101   4
                  4 *
                  9 **

02     CAL WINBUSH
            100   6
            100   2
                  8 *
            101   3
                  3 *
                 11 **

                 20
```

Corrected Output

B

Figure 101 (Part 2 of 4). Unwanted Control Breaks

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Position (1)	Not (N)	C/Z/D	Character	Position (2)	Not (N)	C/Z/D	Character	Position (3)	Not (N)	C/Z/D	Character	Stacker Select	P/B/L/R	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
01	I	SALES	AA			Ø1	1		C	A																				
02	I																				2	3		L2FLD	L2					
03	I																				4	16		NAME						
04	I		BB			Ø2	1		C	B																				
05	I																				2	3		L2FLD	L2					
06	I																				4	6		L1FLD	L1					
07	I																				7	9	Ø	AMT						
08	I																													
09	I																													
10	I																													
11	I																													
12	I																													
13	I																													
14	I																													
15	I																													
16	I																													
17	I																													

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation

Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	And	And	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus / High (1>2)	Minus / Low (1<2)	Zero / Equal (1=2)	Comments
01	C		Ø1				SETON						11			
02	C		Ø2				SETOF						11			
03	C		Ø2			AMT	ADD	L1TOT	L1TOT	5	Ø					
04	C	L1				L1TOT	ADD	L2TOT	L2TOT	5	Ø					
05	C	L2				L2TOT	ADD	LRTOT	LRTOT	5	Ø					
06	C															
07	C															
08	C															
09	C															
10	C															
11	C															
12	C															
13	C															
14	C															
15	C															
16	C															
17	C															

©

Figure 101 (Part 3 of 4). Unwanted Control Breaks

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050*
Printed in U.S.A.

Program | Punching Instruction | Graphic | Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Programmer | Date

Line (3-5)	Form Type (6)	Filename (7-14)	Type (H/D/T/E) (15)	Stacker # / Fetch (F) (16)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators (23-31)	Field Name / *AUTO (32-37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40-43)	P/B/L/R (44)	Constant or Edit Word (45-70)	(71-74)
01	O	PRINTER	D		1	1			Ø1							
02	O									L2FLD			5			
03	O									NAME			25			
04	O		D			1			Ø2							
05	O									L1FLD			15			
06	O									AMT	Z		25			
07	O		T			1			L1N11							
08	O									L1TOT	Z	B	25			
09	O												27		'*'	
10	O		T			1			L2							
11	O									L2TOT	Z	B	25			
12	O												28		'**'	
13	O		T			1			LR							
14	O									LRTOT	Z	B	25			
15	O															
16	O															
17	O															
18	O															
19	O															
20	O															

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Ⓓ

Figure 101 (Part 4 of 4). Unwanted Control Breaks

Example 4: Figure 102 shows a split control field made up of three portions. The control level indicator (L4) which is used for all three portions indicates that they are all to be treated as one control field. The field can be pictured as follows:

CUSNO	ACCTNO	REGNO
1 6 7 9	8 6 5 3 9 7	1 1 1

The first field assigned the same control level indicator begins the control field; the last ends it.

COLUMNS 61-62 (MATCHING FIELDS)

Entry	*Explanation*
M1-M9	Any matching level

Use columns 61-62 to specify matching fields and sequence checking.

An entry in columns 61-62 indicates:

1. Matching fields and sequence checking when you have two or more input, update, or combined files with match fields (see general discussion that follows under *Multifile Processing*).

2. Only sequence checking when you have just one input, update, or combined file.

Matching Fields

Make an entry in columns 61-62 when you wish to compare records from two or more input, update, or combined files in order to determine when records match. Records can be matched by matching one field, many fields, or entire records. You can indicate as many as nine matching fields (M1-M9). Whenever the contents of the match fields from records of the primary file are the same as the contents of the match fields from a secondary file, the matching record (MR) indicator turns on. M1-M9 are used only to identify fields by which records are matched. The values M1-M9 are not indicators, but do cause MR to turn on when a match occurs. Matching is allowed with primary and secondary files only. Figures 103 and 104 show selection of records by matching fields from two or three input files.

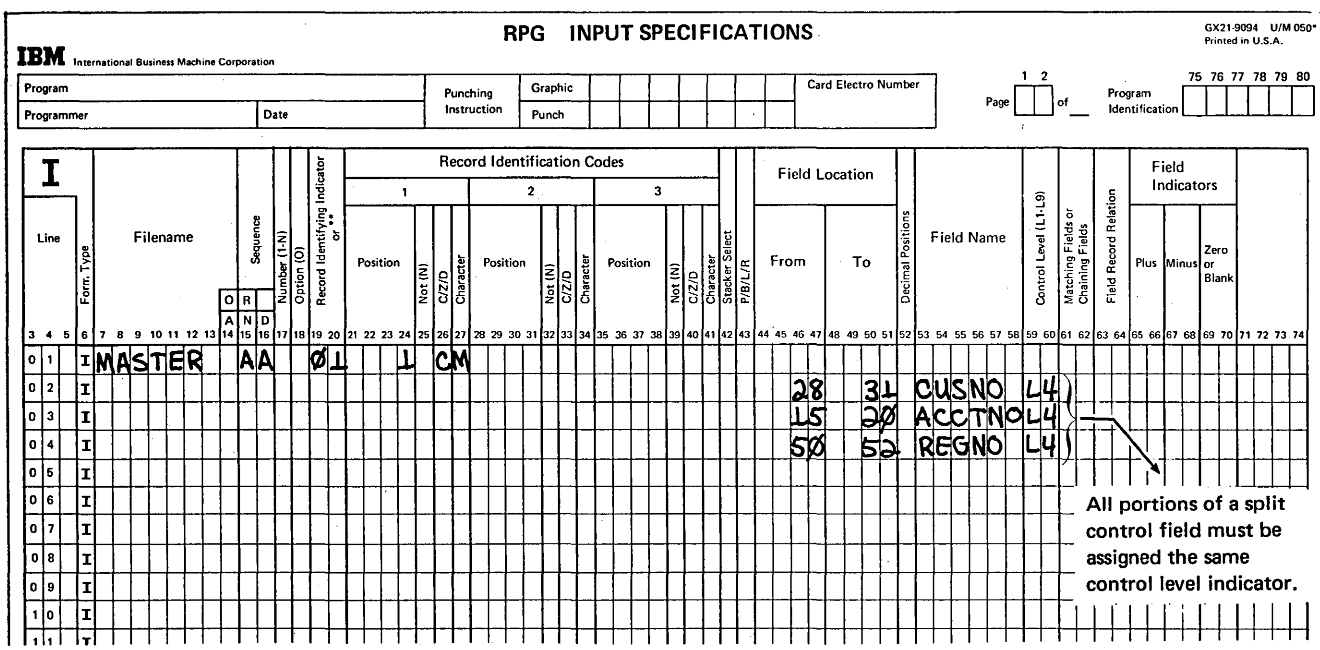


Figure 102. Split Control Fields

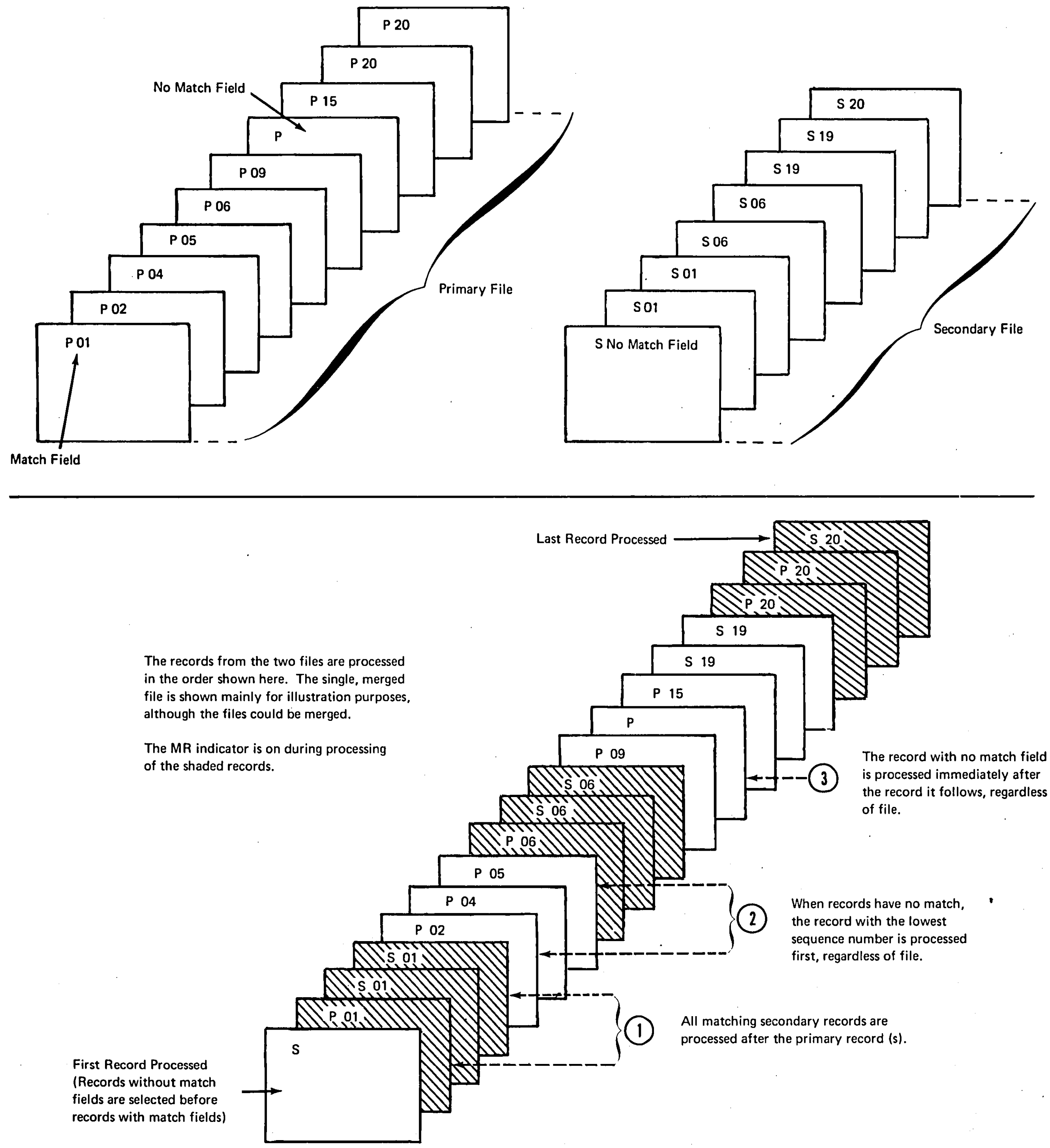


Figure 103. Processing Two Files by Matching Fields

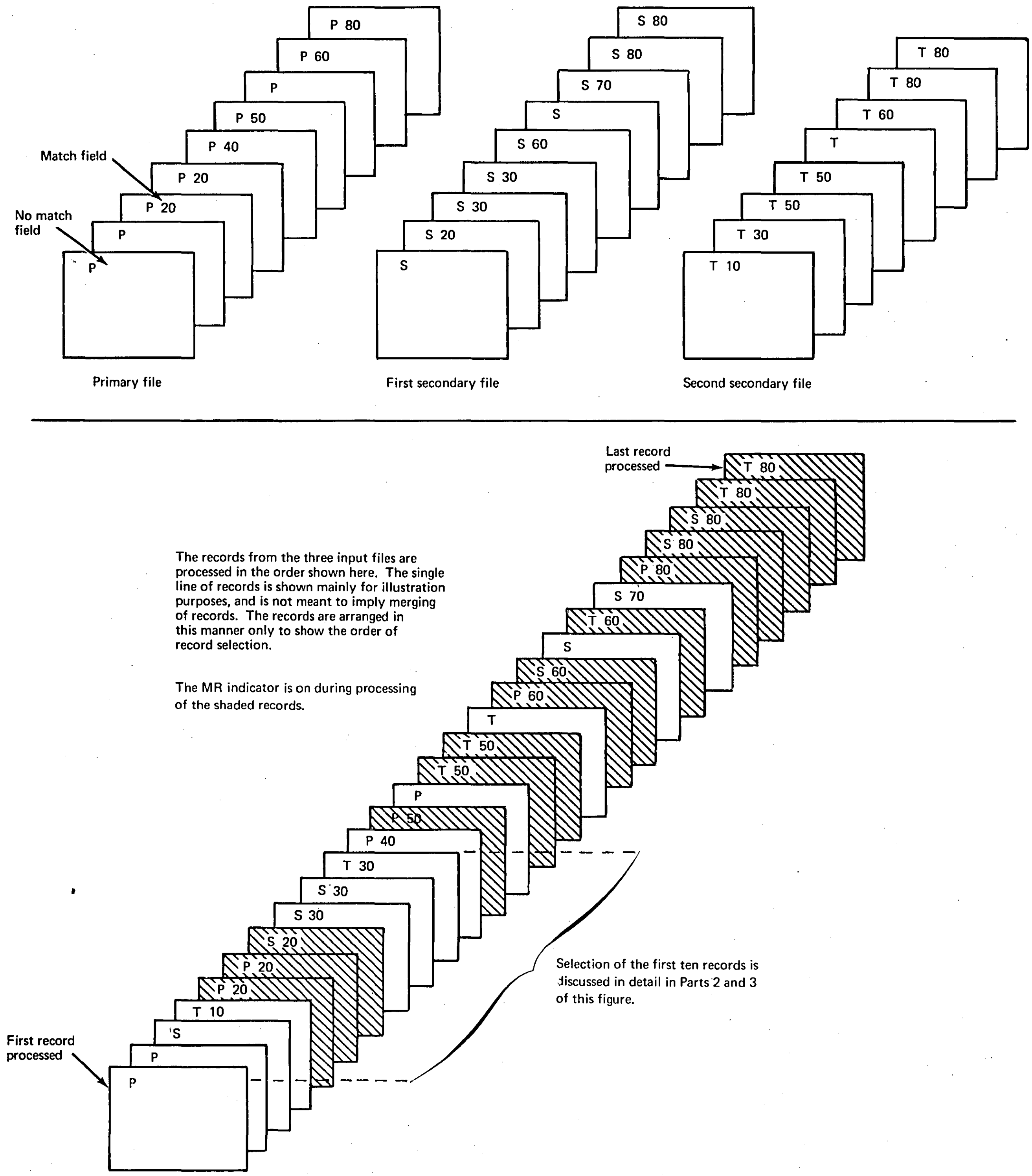
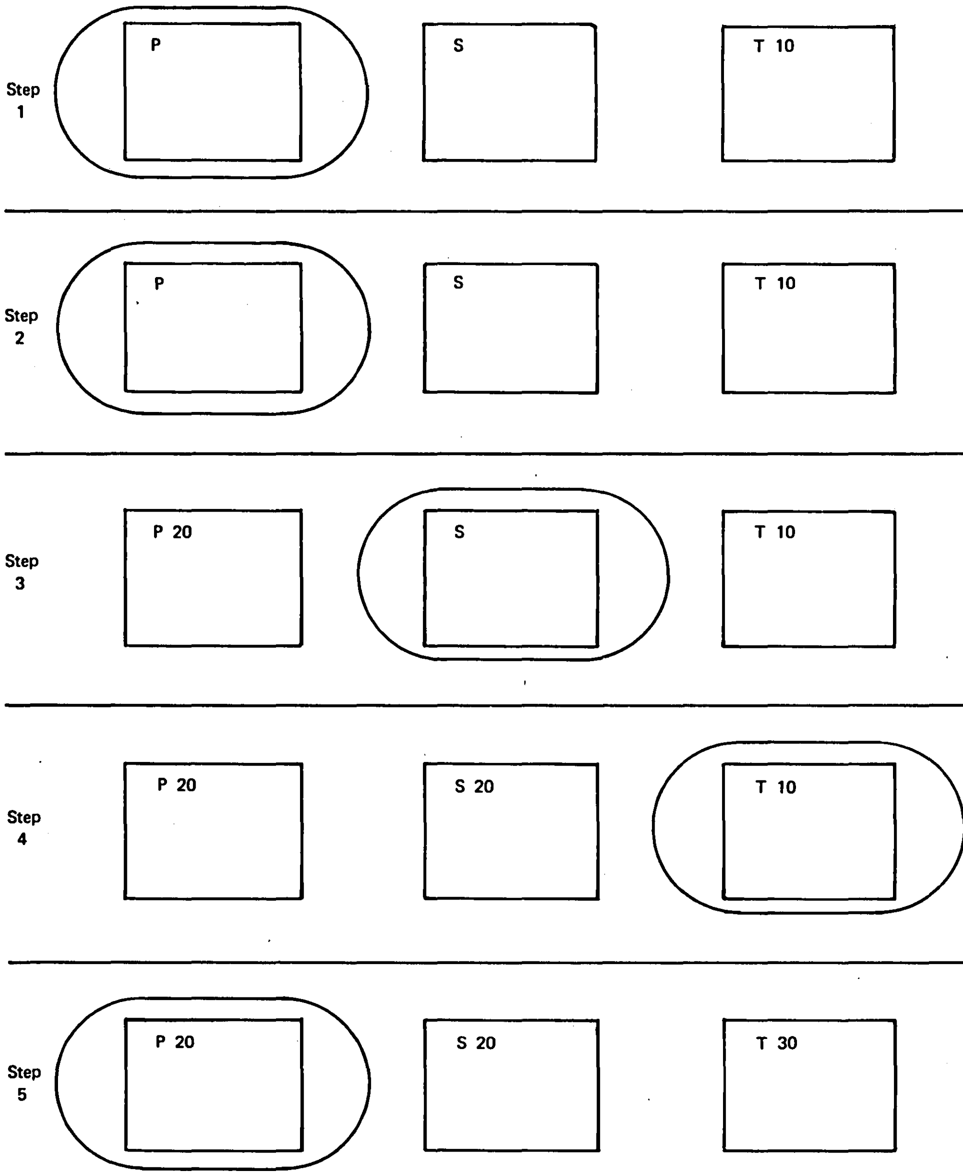


Figure 104 (Part 1 of 3). Normal Record Selection from Three Files



The first record from each file is read. The P and S records have no match field, so they are processed before the T record which has a match field. Because the P record comes from the primary file, it is selected for processing first.

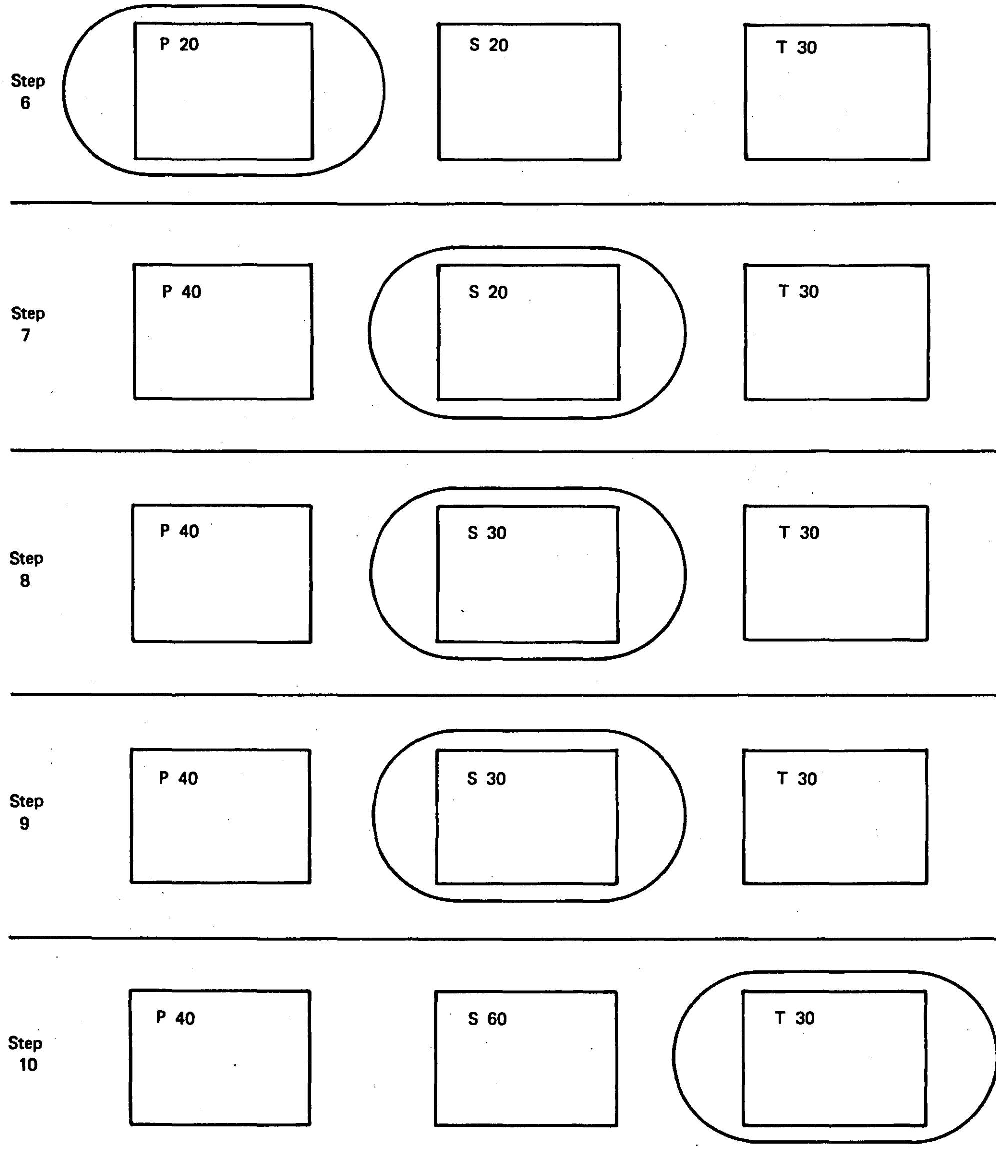
The next P record is read. It contains no match field, and comes from the primary file, so the new P record is also selected for processing before the S record.

The next P record read has a match field. The S record has no match field, so it is selected for processing.

The next S record is read. All three records have match fields. Because the value in the match field of the T record is lower than the value in the other two, the T record is selected for processing.

The next T record is read. The matching P and S records both have the low match field value, so they are processed before the T record. Since the matching P record comes from the primary file, it is selected for processing first. MR indicator is turned on at this point.

Figure 104 (Part 2 of 3). Normal Record Selection from Three Files



The next P record is read. Because it contains the same match field and comes from the primary file, the new P record is selected instead of the S record.

The next P record is read. The value of the match field in the S record is the lowest of the three, so the S record is selected for processing.

The next S record is read. Because the S and T records match and have the lowest match field, they are selected before the P record. Because the S record comes from the first secondary file, it is selected for processing before the T record.

The next S record is read. Because it also has the same match field as the S record just selected, it too is selected before the T record.

The next S record is read. The T record contains the lowest match field value, and is selected for processing.

Figure 104 (Part 3 of 3). Normal Record Selection from Three Files

MR (Matching Record Indicator)

Use the MR indicator to condition calculation and output operations which are to be done only when records match.

The MR indicator turns on when a primary file record matches any secondary file record on the basis of the matching fields indicated by M1-M9. The matching record indicator is always set before detail calculations. It retains this setting for one complete cycle. If all primary file records match all secondary file records, the MR indicator is always on. If record types for which no matching fields have been specified are read, MR is turned off.

A record selected by FORCE causes the MR indicator to remain off for one cycle while the forced record is processed.

Sequence Checking

Make an entry in columns 61-62 when you want to sequence check records within one input, update, or combined file. This entry causes sequence checking of the data in the fields to which M1-M9 have been assigned (see *Columns 15-16* in this chapter for sequence checking of record types).

You may use as many as nine fields (M1-M9) to sequence check. The sequence (ascending or descending) of your record file must be specified in the file description specifications (see *Column 18* in Chapter 4). An entry in columns 61-62 indicates that the records are to be checked to see if they really are in the sequence specified (see *Examples, Example 3*).

MULTIFILE PROCESSING

Multifile processing applies to programs that read records from a primary file and one or more secondary files. It is the name given to the methods by which programs select records for processing. The method used depends upon whether or not match fields are used in the records.

No Match Fields

When no match fields are used, records are selected from one file at a time. When the records from one file have all been processed, records from the next file are selected. The *files* are processed in this order:

1. Primary file.

2. Secondary files in the order in which they are described in the file description specifications.

Match Fields

When match fields are used, records are selected according to the contents of the match fields. One record is read from every file, and the match fields in the records are compared. If the records are in ascending order, the record with the lowest match field is selected for processing. If the records are in descending order, the record with the highest match field is selected.

When a record is selected from a file and processing from that file takes place, the next record from the file is read. At the beginning of the next program cycle, the new record is compared with the records that had not been selected during the previous cycle, and one is selected.

Records without match fields can be included in the files. Such records are selected before records with match fields. If two or more of the records being compared have no match fields, selection of those records is determined by the priority of the files from which the records came.

When the primary record matches one or more of the secondary records, the MR indicator is turned on. The indicator can be used to condition calculations or output for the record that is selected. If one of the matching records must be selected, the selection is determined by the priority of the files from which the records came.

For a discussion of multifile processing at end-of-file, see *Column 17 (End of File)* in Chapter 4.

Assigning Matching Field Values

1. Sequence checking is automatically done for all record types with matching field specifications. The contents of the fields to which M1-M9 have been assigned are checked for correct sequence. An error in sequence stops the program. The record which caused the halt is not processed. When the machine is restarted, the next record from the same file is read. Thus, all matching fields must be in the same order, either all ascending or all descending (see *Column 18* in Chapter 4).

2. Not all files used in the job must have matching fields. Not all record types within one file must have matching fields either. However, at least one record type from two files must have matching fields if files are ever to be matched.

3. The same number of matching fields must be specified for all record types which are used in matching. The same matching record values must also be used for all types (see *Examples, Example 1*).

4. All match fields given the same matching record value (M1-M9) must be the same length and type (alphameric or numeric).

   Note: When using packed fields the unpacked length [(2 x packed length)-1] is regarded as the length of the matched field.

5. Record columns of different matching fields may overlap, but the total length of all fields must not exceed 144 characters.

6. If more than one matching field is specified for a record type, all the fields are combined and treated as one continuous matching field (see *Examples, Example 2*). They are combined according to descending sequence (M9-M1) of matching record values.

7. Matching fields may not be split. This means that the same matching field value cannot be used twice for one type of record.

8. Matching fields may be either alphameric or numeric (but not binary). However, all matching fields given the same matching record value (M1-M9) are considered numeric if any of those matching fields is described as numeric. Numeric matching fields contain only the digits 0-9. Thus, matching fields of 050 and b50 (where ƀ denotes blank) will compare equal.

9. When numeric fields having decimal positions are matched, they are treated as if they had no decimal position.

10. Only the digit portions of numeric match fields are compared. Even though a field is negative it is considered to be positive since the sign of the numeric field is ignored. Thus, a *-5* will match with a *+5*.

11. Whenever more than one matching record value is used, all match fields must match before the MR indicator turns on. For example, if matching fields M1, M2, M3 are specified, all three fields from the primary file must match all three fields from the other record. A match on only the M1 and M2 fields will not turn on the MR indicator (see *Examples, Example 1*).

12. Field names are ignored in matching record operations. Therefore, fields from different record types which have been assigned the same match level may have the same name.

13. If you have defined an alternate collating sequence for your program, alphameric fields are matched according to the sequence you have specified. Matching fields contain a corresponding initial alternate collating sequence value; that is, they are set to the lowest alternate sequence value if ascending sequence is specified, and to the highest alternate sequence value if descending sequence is specified.

14. Matching is not allowed with demand or chained files.

15. If a program contains files with match fields as well as files without match fields, the files without match fields are processed before the files with match fields.

    Note: Additional rules applying to matching records when used with entries in the Field Record Relation columns are discussed in *Columns 63-64* in this chapter.

Processing Matching Records—Two or More Files

1. Whenever a record from the primary file matches a record from the secondary file, the primary file record is processed first. Then the matching secondary file record is processed unless another file is forced (see *Operation Codes, FORCE* in Chapter 8). Remember, the record identifying indicator which identifies the record type just selected is on at the time the record is processed. This indicator is often used to control the type of processing that takes place.

2. Whenever records from ascending files do not match, the record having the lowest match field content is processed first (Figure 103). Whenever records from descending files do not match, the record having the highest match field content is processed first.

3. A record type which has no matching field specification is processed immediately after the record it follows. The MR indicator is off. If this record type is first in the file, it is processed first even if it is not in the primary file (Figure 103).

4. The matching of records makes it possible to enter data from primary records into their matching secondary records since the primary record is processed before the matching secondary record. However, the transfer of data from secondary records into matching primary records can only be done through look ahead fields (see *Columns 19-20* in Chapter 7).

For additional information on matching records from more than two files see *Operation Codes, FORCE* in Chapter 8.

Examples

Example 1: Figure 105 shows three record types that are used in matching records. All record types have three matching fields specified and all use the same values (M1, M2, M3) to indicate which fields must match. The MR indicator turns on only if all three match fields in either of the record types from the MASTER file are the same as all three fields from the record in the WEEKLY file.

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Position (1)	C/Z/D (1)	Character (1)	Field Location From	Field Location To	Decimal Positions	Field Name	Matching Fields
01	I	MASTER	AA	Ø1	95	C	S					
02	I							1	5		EMPLNO	M1
03	I							1Ø	15	2	PAYRAT	
04	I							36	4Ø		DIVSON	M3
05	I							31	35		DEPT	M2
06	I		BB	Ø2	95	C	T					
07	I							1	5		EMPLNO	M1
08	I							1Ø	15	Ø	HRSWKD	
09	I							2Ø	24		DEPT	M2
10	I							25	29		DIVSON	M3
11	I	WEEKLY	DL	Ø3								
12	I							1	5		EMPLNO	M1
13	I							6	1Ø		DIVSON	M3
14	I							11	15		DEPT	M2
15	I							2Ø	25	Ø	HRSWKD	
16	I											
17	I											
18	I											
19	I											
20	I											

Figure 105. Match Fields

Example 2: Figure 105 indicates three matching fields on one record. These three are combined and treated as one matching field organized as follows:

DIVSON	DEPT	EMPLNO
M3	M2	M1

The order in which the fields are specified on the input specifications does not affect the organization of the match fields in the computer.

Example 3: An input file called MASTER is to be sequence checked using three fields (Figure 106). Data from two records is shown below:

Data from First Record		*Data from Second Record*	
DEPT	008	DEPT	003
REGION	051	REGION	025
DIVSON	003	DIVSON	005

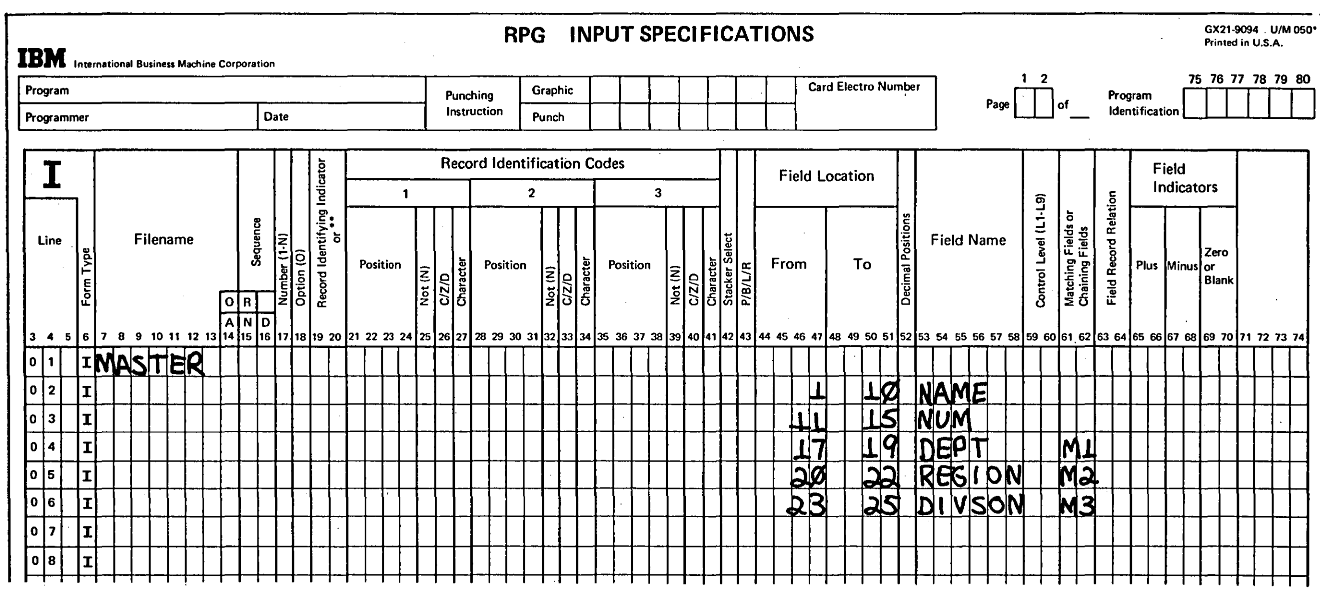
In sequence checking, all fields are treated as one continuous field. Thus, the matching fields look like:

	M3	M2	M1
Record 1	003	051	008
Record 2	005	025	003

The matching field from record 1 is compared with the matching field from record 2. If the file is specified to be in ascending sequence, the records are in order since 005025003 is higher than 003051008. However, if the file is specified as having a descending sequence, card 2 is out of order and a halt occurs.

COLUMNS 63-64 (FIELD RECORD RELATION)

Entry	*Explanation*
01-99	Record identifying indicator assigned to a record type.
L1-L9	Control level indicator previously used.
MR	Matching record indicator.
U1-U8	External indicator previously set.
H1-H9	Halt indicator previously used.


RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9094 U/M 050* Printed in U.S.A.

Line	Form Type	Filename	Field Location From	Field Location To	Field Name	Matching Fields or Chaining Fields
01	I	MASTER				
02	I		1	10	NAME	
03	I		11	15	NUM	
04	I		17	19	DEPT	M1
05	I		20	22	REGION	M2
06	I		23	25	DIVSON	M3
07	I					
08	I					

Figure 106. Match Fields (Sequence Checking Within a File)

Columns 63-64 have several uses which are discussed after these general rules:

1. All fields, including matching or control fields, that have no field record relation specification should come before those that do.

2. All fields related to one record type (that is, having the same Field Record Relation entry) should be entered as a group in specification lines following one another for more efficient use of core storage. These fields could, however, be entered in any order.

3. All portions of a split control field *must* be assigned the same field record relation indicator and must be entered as a group in specification lines following one another (see *Examples, Example 1*). For more information on split control fields, see *Columns 59-60* in this chapter.

4. When used with match or control fields, the field record relation indicator must match a record identifying indicator for this file, and the match or control fields must be grouped according to the field record relation indicator.

5. When any match value (M1-M9) is specified without field record relation, all match values used must be specified once without field record relation. If all match fields are not common to all records, a dummy match field should be used.

Record Identifying Indicators (01-99)

Columns 63-64 are commonly used when several record types have been specified in an OR relationship. Fields which have no field record relation indicator are associated with all the record types in the OR relationship. This is fine when all record types have the same fields. But if the record types in the OR relationship have some fields that are the same and some that are not the same, you do not want to associate every field with all records. Therefore, you must have some way of relating a field to a certain record. To do this, place in columns 63-64 the record identifying indicator found in columns 19-20 of the record type on which the field is found, or specify an indicator (01-99) which was previously defined in your program and which you now wish to use to condition data movement from the input area to the storage area (see *Examples, Example 2*).

Control fields (indicated by entries in columns 59-60) and matching fields (indicated by entries in columns 61-62) may also be related to a particular record type in an OR relationship by a field record relation entry. Control fields or matching fields that are not related to any particular record type in the OR relationship by the field record relation indicator are used with all record types in the OR relationship.

When two control fields have the same control level indicator or two matching fields have the same matching level entry, it is possible to assign a field record relation indicator to just one of the control fields or to just one of the matching fields. In this case, only the specification having the field record relation indicator is used when that indicator is on. If none of the field record relation indicators are on for that control field or matching field, the specification without a field record relation indicator is used. Control fields and matching fields cannot have an L1-L9, U1-U8, or MR entry in columns 63-64.

Control Level (L1-L9) and Matching Record (MR) Indicators

Another situation for which you may use these columns is when you wish to accept and use data from a particular field only when a certain condition (such as matching records or a control break) occurs. You indicate the conditions under which you accept data from a field by indicator L1-L9 or MR. Data from the field named in columns 53-58 is accepted only when the indicator is on (see *Examples, Example 3*).

External Indicators (U1-U8)

You may also use these columns to condition a specification by an external indicator (U1-U8). The external indicator which you set prior to processing conditions whether a field is to be used in the program. When the indicator is on, the field is read; when the indicator is off, the field is not read.

External indicators are primarily used when file conditioning is done by an entry in columns 71-72 in the file description specifications. However, they may also be used to condition when a specification should or should not be done even though file conditioning is not specified. See *Columns 71-72* in Chapter 4.

Halt Indicators (H1-H9)

A halt indicator is used to relate a field to a record that is in an OR relationship and also has a halt indicator specified in columns 19-20.

Examples

Example 1: Split control fields on one record type must have the same record relation entry. Figure 107, insert A, shows several record types with split control fields in each. The record identified by a *1* in column 95 has two split control fields:

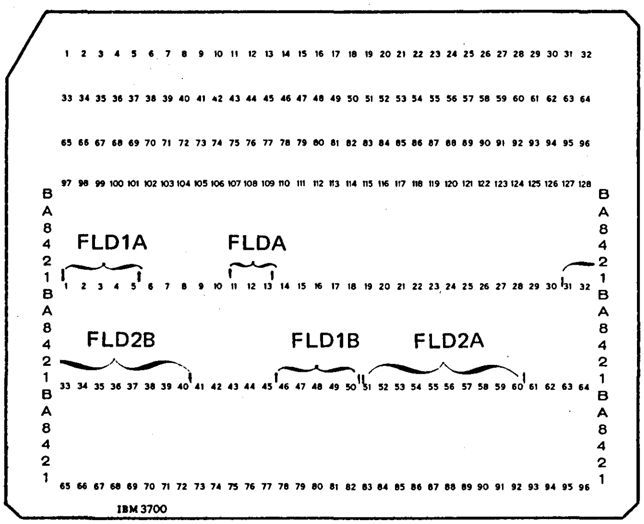
FLD1A and FLD1B
FLD2A and FLD2B

The record with a *2* in column 95 has three split control fields.

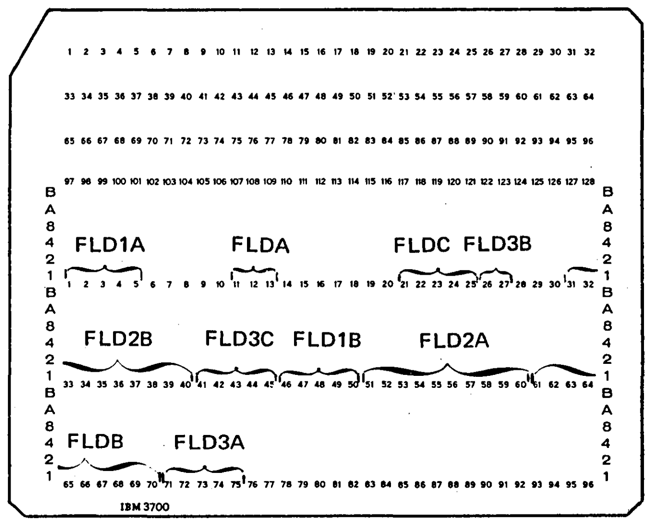
FLD1A and FLD1B
FLD2A and FLD2B
FLD3A, FLD3B, and FLD3C

The third record type, identified by the *3* in column 95, also has three split control fields:

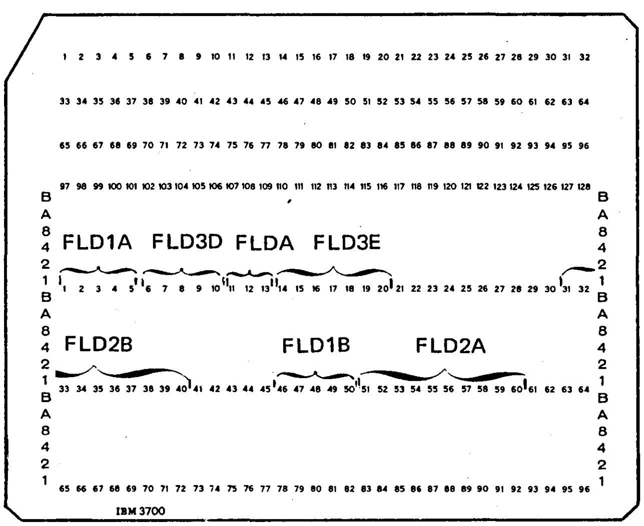
FLD1A and FLD1B
FLD2A and FLD2B
FLD3D and FLD3E



Record identification code = 1



Record identification code = 2

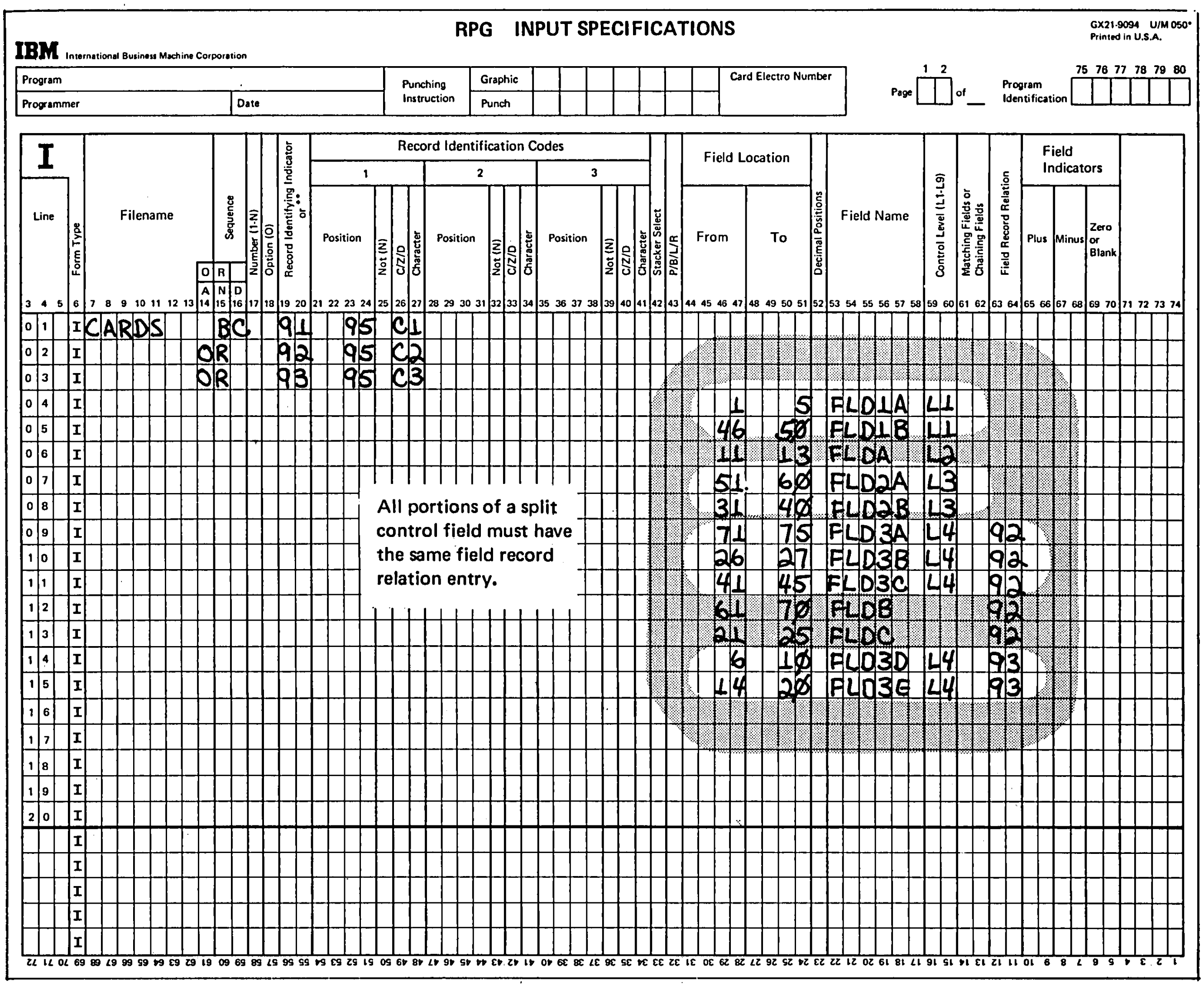


Record identification code = 3

A

Figure 107 (Part 1 of 2). Field Record Relation (Split Control Fields)

All portions of the split control field must be assigned the same control level indicator and all must have the same field record relation entry. Figure 107, insert B, shows the field record relation required for the three record types.



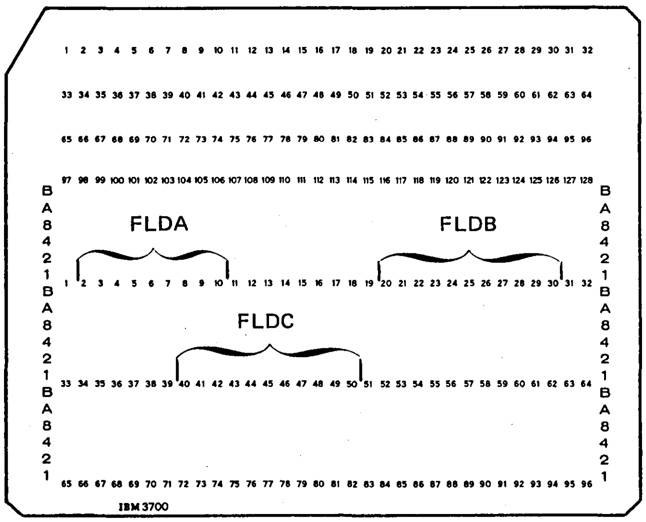
Line	Form Type	Filename	Sequence	Record Identifying Indicator	Position	C/Z/D	Character	From	To	Field Name	Control Level (L1-L9)	Field Record Relation
01	I	CARDS	BC	91	95	C	1					
02	I	OR		92	95	C	2					
03	I	OR		93	95	C	3					
04	I							1	5	FLD1A	L1	
05	I							46	50	FLD1B	L1	
06	I							11	13	FLDA	L2	
07	I							51	60	FLD2A	L3	
08	I							31	40	FLD2B	L3	
09	I							71	75	FLD3A	L4	92
10	I							26	27	FLD3B	L4	92
11	I							41	45	FLD3C	L4	92
12	I							61	70	FLDB		92
13	I							21	25	FLDC		92
14	I							6	10	FLD3D	L4	93
15	I							14	20	FLD3E	L4	93

B

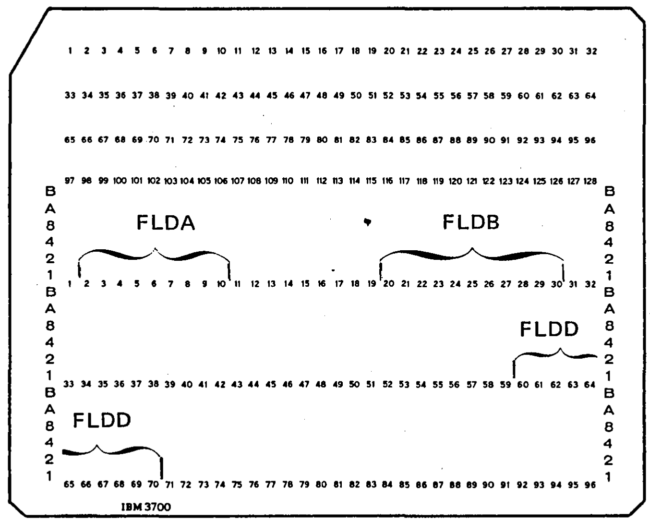
Figure 107 (Part 2 of 2). Field Record Relation (Split Control Fields)

Example 2: Figure 108 shows how record identifying indicators are used to relate a field to a record. The file contains two different types of records, one identified by a *5* in column 1 and the other by a *6* in column 1. FLDC is related by record identifying indicator 14 to the record type which is identified by a *5* in column 1. FLDD is related to the record type having a *6* in column 1 by record identifying indicator 16. This means that FLDC is found on only one type of record (that identified by *5* in column 1) and FLDD is found only on the other type. FLDA is conditioned by indicator 07 which you had previously defined elsewhere in your program. FLDB is found on both types since it is not related to any one type by a record identifying indicator.

Example 3: Suppose you were printing a monthly report showing all items sold in each department in your company. You also want the report to list the name of the manager of each department. Each input record then has the department number (DEPT), the manager's name (MANAGR), and one item (ITEM) that was sold by that department. Fields are described as shown in Figure 109. The records are arranged in order by department.



Record identification code = 5



Record identification code = 6

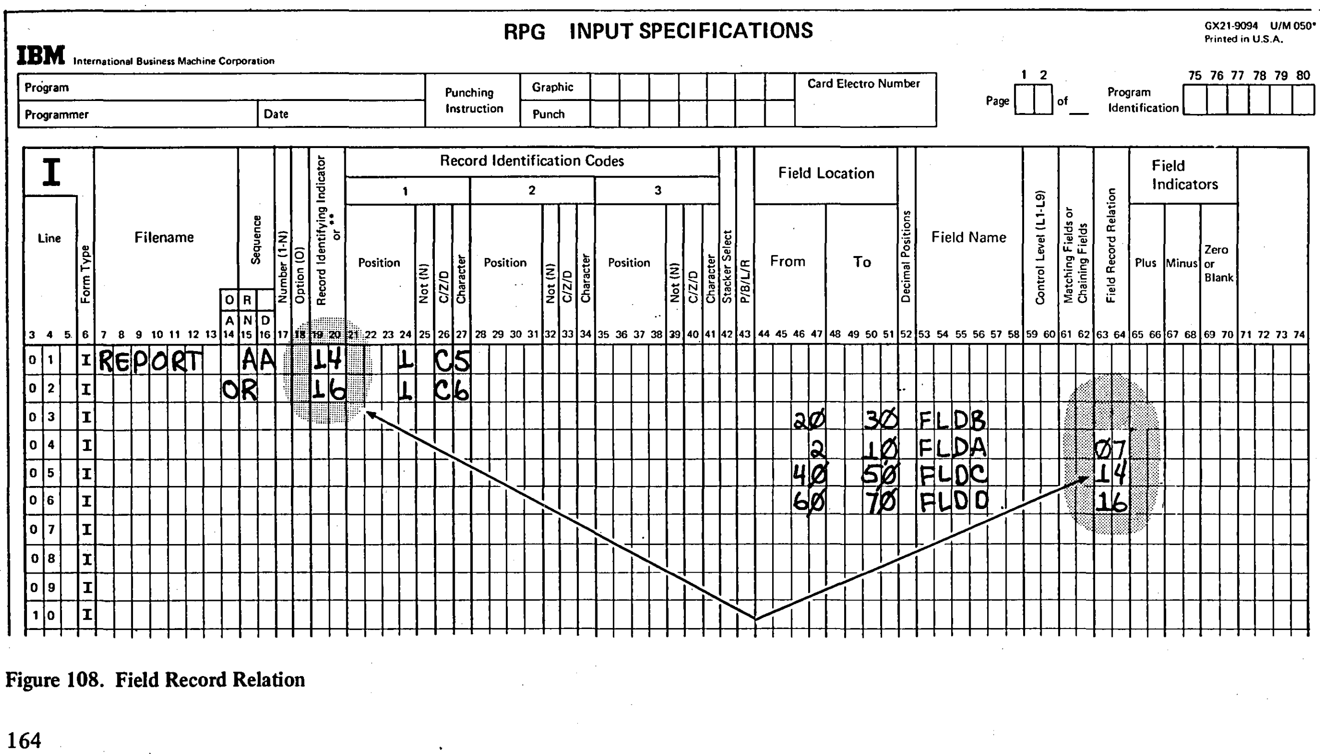
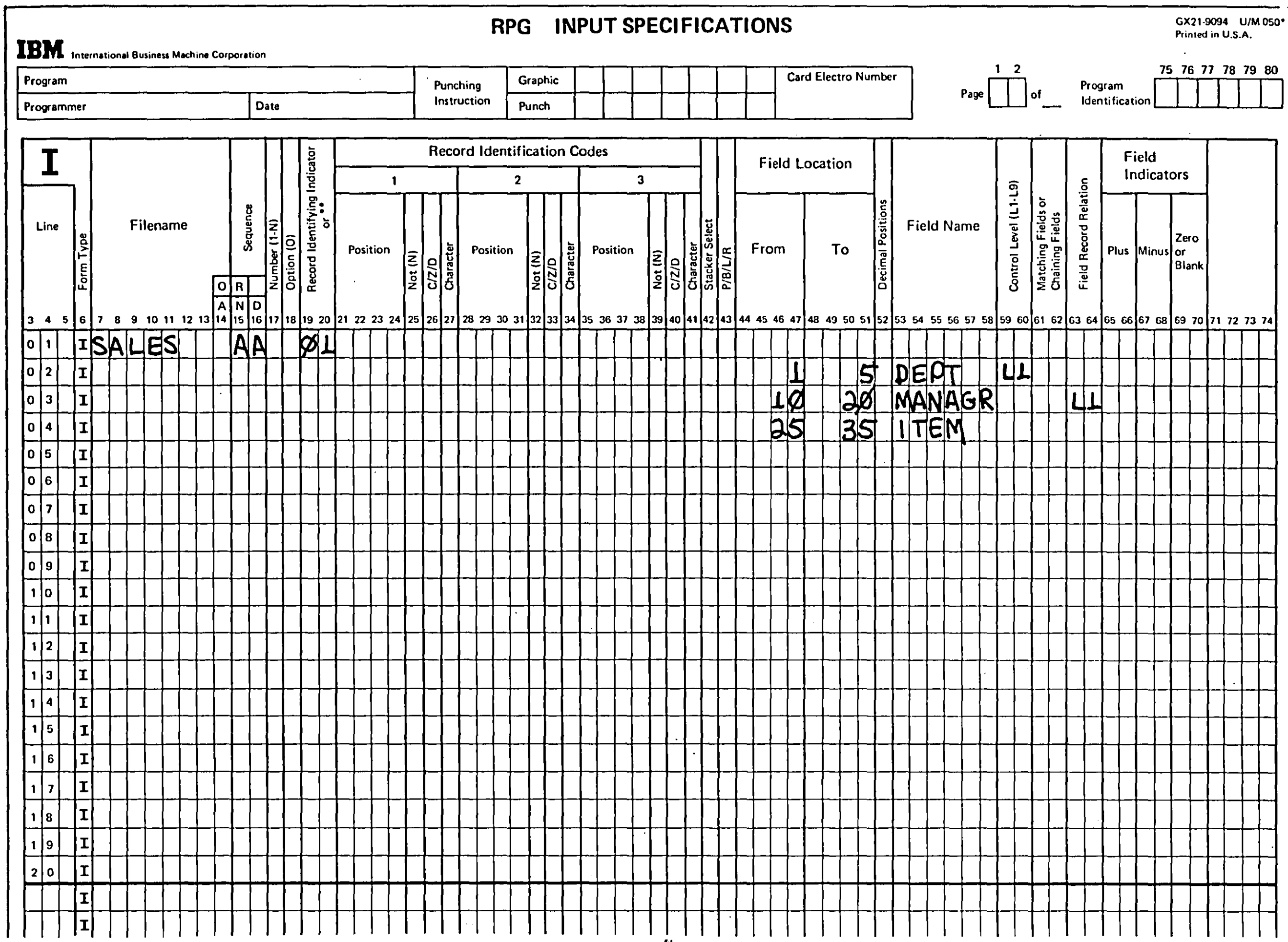


Figure 108. Field Record Relation



Line	Form Type	Filename	Sequence	Record Identifying Indicator	Field Location From	Field Location To	Field Name	Control Level (L1-L9)	Field Record Relation
01	I	SALES	AA	01					
02	I				1	5	DEPT	L1	
03	I				10	20	MANAGR		L1
04	I				25	35	ITEM		

Figure 109. Field Record Relation: Accepting Data From a Field

In the report it is not necessary to print the manager's name for every item that was sold in his department. Instead, it should be printed only when the first record containing an item sold in a different department is read. The field called DEPT is established as a control field.

Remember that the manager's name is printed only when information in the control field changes. Thus the information from the field called MANAGR is not used often. It would be wasted time to accept that information every time a record is read. The L1 entry in columns 63-64 indicates that the data from the field called MANAGR is to be accepted only when a control break occurs.

COLUMNS 65-70 (FIELD INDICATORS)

Entry	*Explanation*
01-99	Field indicator.
H1-H9	Halt indicator (when checking for an error condition in the data).

Use field indicators 01-99 when you wish to test a field for a condition of either plus, minus, zero, or blank. The indicator specified turns on if the condition is true for the input record; it remains off or turns off if the condition is not true for the input record. You usually use these same indicators to control certain calculation or output operations (see *Columns 9-17* in Chapter 8 or *Columns 23-31* in Chapter 9).

The three conditions you may check for are:

1. Plus (columns 65-66). Any valid indicator entered here is turned on if the numeric field named in columns 53-58 is greater than zero.

2. Minus (columns 67-68). Any valid indicator entered here is turned on if the numeric field in columns 53-58 is less than zero.

3. Zero or blank (columns 69-70). Any valid indicator entered here is turned on if a numeric field named in columns 53-58 is all zeros or if an alphameric field is all blanks.

   A numeric field (from the record), which is all blanks, turns on an indicator specified for all zeros. However, if an alphameric field is all zeros, the field will not turn on an indicator specified for all blanks.

In the input specifications, you specify the indicators that will be used to condition operations. In the calculation specifications and output specifications, you actually use these indicators. When conditioning operations, you must know when the indicators will be off and when they will be on. When assigning and using field indicators in columns 65-70, remember:

1. Indicators for plus or minus are off at the beginning of the program. They are not turned on until the condition (plus or minus) is satisfied by the field being tested on the card just read.

2. An indicator assigned to zero or blank is off at the beginning of the program. It remains off until the field being tested is zero or blank.

3. One input field may be assigned two or three field indicators. However, only the one which signals the result of the test turns on; the others are turned off.

4. If the same field indicator is assigned to fields in different record types, its status is always based on the last record type selected.

5. When different field indicators are assigned to fields in different record types, a field indicator turned on will remain on until another record of that type is read. Similarly, a field indicator assigned to more than one field within a single record type will always reflect the status of the last field defined.

6. Field indicators assigned in these columns may be SETON or SETOF in calculation specifications.

Halt Indicators

Specify any halt indicator (H1-H9) in columns 65-70 when you wish to check for an error condition in your data. For example, if a field should not be zero, you can specify a halt indicator to check for that zero condition. If a zero field is found, the halt indicator turns on and the job stops after the record with the zero field has been processed.

Indicators H1-H9 cause the program to halt after the record which caused the indicator to turn on is completely processed.

COLUMNS 71-74

Columns 71-74 are not used.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

See Chapter 2.

Calculation specifications describe the calculations you want performed on your data and the order in which you want them performed. Each calculation specification can be divided into three parts that indicate:

1. When the operation is to be performed (columns 7-17). The indicators entered in these columns determine under what conditions the operation specified is to be done.

2. What kind of operation (columns 28-32) is to be performed on the data in columns 18-27 and/or columns 33-42. Entries in these fields describe the kind of operation to be done. They also specify the data upon which the operation is to be performed.

3. What tests are to be made on the results of the operation (columns 54-59). The indicators entered here signal the result of the operation and may serve to condition other operations.

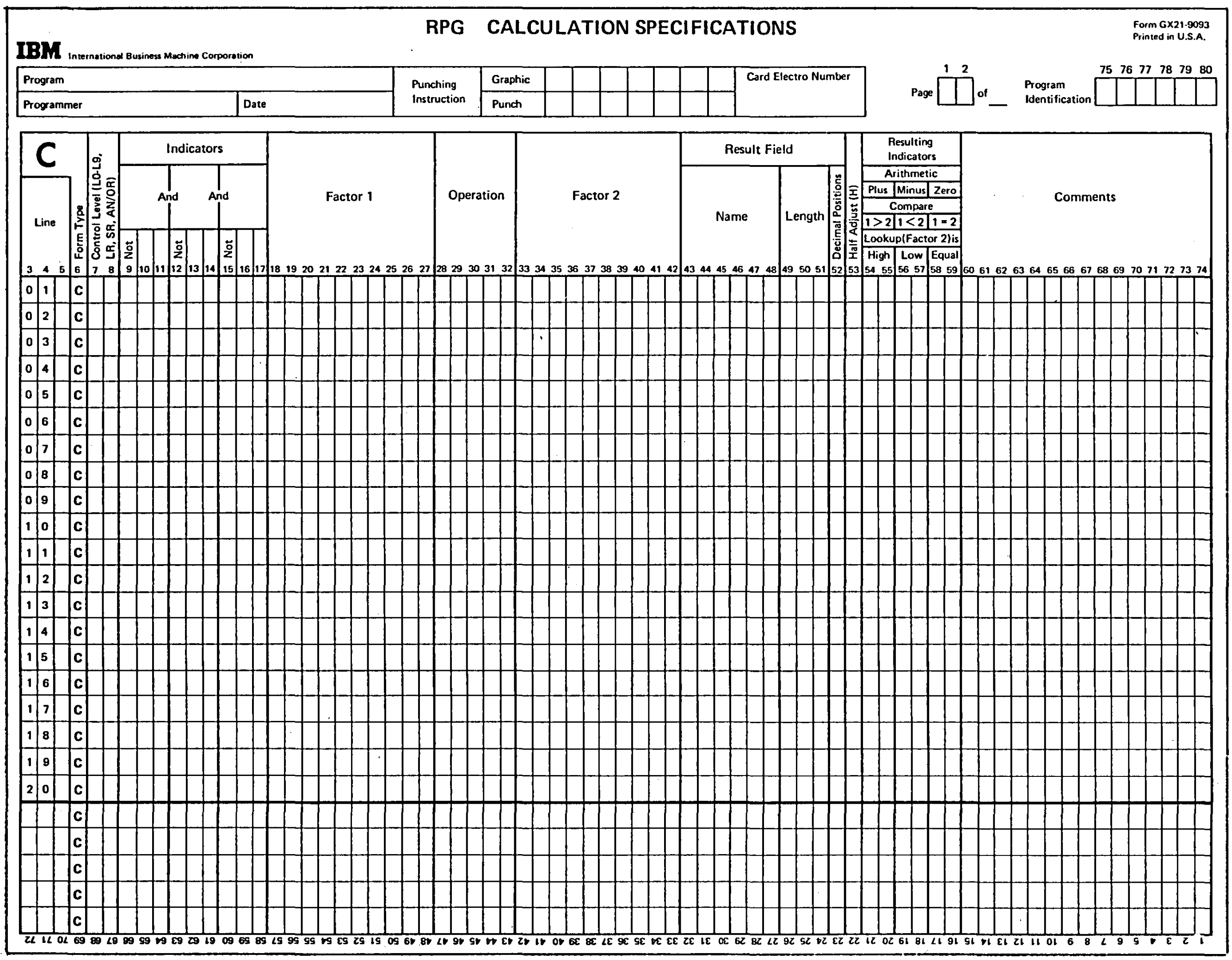
Write these specifications on the Calculation Sheet (Figure 110).

COLUMNS 1-2 (PAGE)

See Chapter 2.

COLUMNS 3-5 (LINE)

See Chapter 2.



IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Program | Punching Instruction | Graphic | Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Programmer | Date

C | Line | Form Type | Control Level (L0-L9, LR, SR, AN/OR) | Indicators (Not, And, Not, And, Not) | Factor 1 | Operation | Factor 2 | Result Field (Name, Length) | Decimal Positions | Half Adjust (H) | Resulting Indicators (Arithmetic: Plus, Minus, Zero; Compare: 1>2, 1<2, 1=2; Lookup(Factor 2)is: High, Low, Equal) | Comments

Figure 110. Calculation Sheet

COLUMN 6 (FORM TYPE)

A *C* must appear in column 6.

COLUMNS 7-8 (CONTROL LEVEL)

Entry	*Explanation*
Blank	Calculation operation is not part of a subroutine and may only be performed for detail calculations.
L0, L1-L9	Calculation operation is done when the appropriate control break occurs or an indicator is set on (L0 is always on).
LR	Calculation operation is done after the last record has been processed or after the LR indicator has been set on by a SETON operation.
SR	Calculation operation is part of a subroutine.
AN,OR	Establishes AN and OR relationships between lines of indicators.

If you leave columns 7-8 blank, the operation specified on the same line is done every time a record is read, provided indicators in columns 9-17 of that line or AN/OR lines associated with that line allow it (see *Columns 9-17* in this chapter).

Calculations must be specified in the following order:

1. Detail (blank in columns 7-8).

2. Total (L0 or L1-L9 in columns 7-8).

3. Last record (LR in columns 7-8). LR calculations must appear after L1-L9 calculations.

4. Subroutine (SR in columns 7-8).

AN/OR lines can appear within any of the above calculations.

Control Level Indicators (L0, L1-L9)

The L0 indicator is on during the entire program. You need never assign this indicator, but you may use it. The indicator is often used when no control fields have been assigned. Remember that when a control break occurs, all operations conditioned by control level indicators are done before those that are not conditioned. If you have no control field but want total calculations to be done and total output records to be written or punched, you may use the L0 indicator to condition those operations (see *Examples, Example 1*).

Use control level indicators L1-L9 to signal when certain operations are to occur. If you specify a control level indicator (L1-L9) in columns 7-8, the operation described on the same specifications line is done only when that indicator is on. Remember that a control level indicator turns on when information in a control field changes (see *Columns 59-60* in Chapter 7).

A control break for a certain level causes all lower control level indicators to turn on. Thus, if you used indicators L3, L2, and L1 in your program, and L3 turns on, L1 and L2 will also turn on. All operations conditioned by L3, L2, and L1 will be done. Exceptions are as follows:

1. When a control level indicator used as a record identifying indicator turns on to reflect the type of record read, only that one control level indicator turns on.

2. When a control level indicator is turned on by the SETON instruction, only that one control level indicator turns on.

Note: In one program cycle, all operations conditioned by control level indicators in columns 7-8 are done at total calculation time. Operations that are conditioned by control level indicators in columns 9-17 are done at detail calculation time immediately following the control break.

LR (Last Record Indicator)

Use the LR indicator to condition all operations that are to be done only at the end of the job. The LR and L1-L9 indicators automatically turn on after the last record of the input file has been processed. It is also possible to turn the LR indicator on by a SETON operation. This does not, however, cause all other control level indicators used to turn on at that point. (LR cannot be turned off by a SETOF operation.) If LR is on, the job ends after all total operations have been performed.

Subroutine Lines (SR)

Use columns 7-8 to indicate that a line is part of a subroutine (see *Subroutines* in Chapter 8). Subroutine lines must be specified last.

AN/OR Lines

Columns 7-8 can be used to specify that lines of indicators are in an AN/OR relationship. By using the AN/OR relationship, many lines of indicators may be grouped together to condition an operation. A maximum of seven AN, OR or AN/OR lines may be used to condition an operation.

The first line of such a group contains blanks in columns 7-8, or an L0-L9, LR, or SR entry if the group of lines is conditioned by a control level indicator or is part of a subroutine. All lines after the first line in the group must have an AN or OR entry in columns 7-8. The indicators on each line are in an AND relationship. It is not necessary to have three indicators on each AN and OR line, but an AN/OR group must have at least one indicator. The last line of the group contains the operation and the necessary operands. All lines in the group prior to the last line must contain blanks in the columns for Factor 1, Factor 2, Operation, Result Field, and Resulting Indicator (see *Examples, Example 2 and 3*).

Note: If LR is used as a resulting indicator, NLR should be used as a conditional indicator to avoid setting off LR.

Examples

Example 1: Figure 111 shows the format of the report printed by the job described in Figure 112. The job shows how total operations can be performed even though there is no control field (no L1-L9 indicators). The job requires:

1. A list of items sold in each district.
2. A total of all sales for each district.
3. A grand total of all sales in all districts.

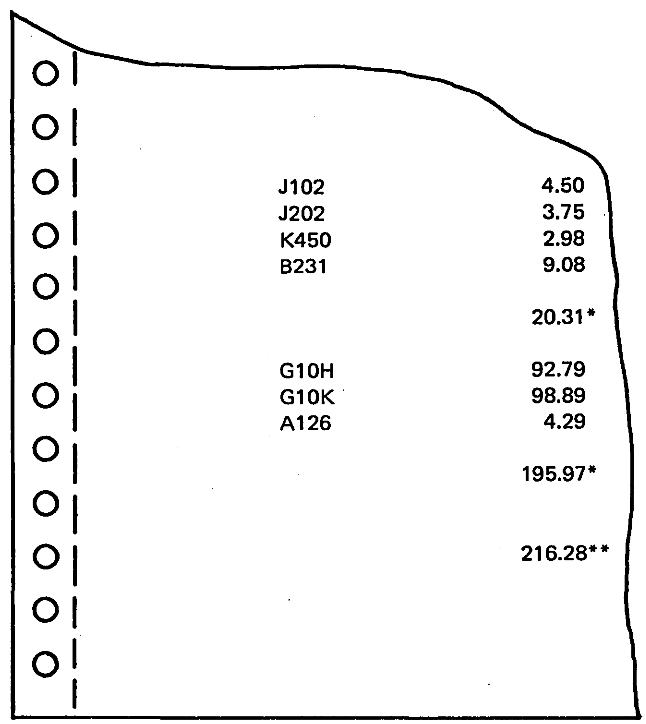


Figure 111. Format of a Printed Report

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050*
Printed in U.S.A.

Program | Punching Instruction | Graphic | Punch | Card Electro Number | Page 02 of | Program Identification
Programmer | Date

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Position (1)	C/Z/D	Character	From	To	Decimal Positions	Field Name
01	I	INPUT	AA	01	1	C	1				
02	I	OR			1	C	M				
03	I							10	15	2	COST
04	I							20	30		ITEM
05	I		BB	02	1	C	2				
06	I	OR			1	C	N				
07	I							30	35	2	COST
08	I							40	50		ITEM
09	I										
10	I										

Ⓐ

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Program | Punching Instruction | Graphic | Punch | Card Electro Number | Page 03 of | Program Identification
Programmer | Date

Line	Form Type	Control Level	Indicators	Indicators (And)	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Resulting Indicators (Plus)
01	C		01		COST	ADD	DISTOT	DISTOT	6	2	
02	C		02		COST	ADD	DISTOT	DISTOT			
03	C					SETOF					21
04	C		01			SETON					21
05	C	L0	02	21	DISTOT	ADD	GDTOT	GDTOT	6	2	
06	C	LR			DISTOT	ADD	GDTOT	GDTOT			
07	C										

Ⓑ

Figure 112 (Part 1 of 2). Use of the L0 Indicator

Line	Form Type	Filename	Type (H/D/T/E)	Space Before	Space After	Output Indicators		Field Name	Edit Codes	B/A/C/1-9/R	End Position in Output Record	Constant or Edit Word
01	O	OUT	D		1	Ø1						
02	O	OR				Ø2						
03	O							ITEM			3Ø	
04	O							COST	1		5Ø	
05	O		T	2	2	Ø2	21					
06	O	OR				LR						
07	O							DISTOT	1	B	5Ø	
08	O										51	'*'
09	O		T	2		LR						
10	O							GDTOT	1		5Ø	
11	O										52	'**'
12	O											
13	O											

Figure 112 (Part 2 of 2). Use of the L0 Indicator

The input records have ITEM and COST fields and a one column record identification field. The records are grouped in ascending sequence by district. The record identification code is used to tell which district a record is from. For example, records from district one are identified either by a *1* or an *M* in column 1. Records from district two are identified either by a *2* or an *N* in column 1 (Figure 112, insert A).

No field on the records can serve as a control field. Certainly, ITEM and COST cannot. The record identifying field cannot either since one district can be identified by two different codes. This means that the contents of this one column identifying field can change even though the district number cannot. Therefore, in order to get total operations without the use of a control field, L0 must be used (See line 05 of Figure 112, insert B). Assume that the five records shown in Figure 113 are read. Refer to Figure 112 as you read the description of operations performed for each record read.

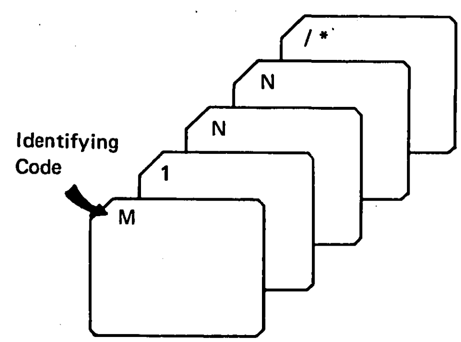


Figure 113. Data Records with No Control Fields

Record	*Indicators On*	*Operations Performed*
(1)	L0	01 turns on. No total operations are performed because conditions in lines 5 and 6 (Calculations Sheet) are not met. (Remember that operations conditioned by control level indicators in columns 7-8 are performed first, but are bypassed on the first RPG II cycle). COST is added to DISTOT. 21 is set on. ITEM and COST are printed out. 01 is turned off. 21 remains on.
(2)	L0, 21	01 is turned on. No total operations are performed. COST is added to DISTOT. ITEM and COST are printed out. 01 is turned off. 21 remains on.
(3)	L0, 21	02 turns on. DISTOT is added to GDTOT. (Conditions for the total operation in line 5 have been met.) DISTOT is printed out. COST is added to DISTOT. 21 is set off. ITEM and COST are printed out. 02 is turned off.
(4)	L0	02 is turned on. No total operations are performed. COST added to DISTOT. ITEM and COST are printed out. 02 is turned off.
(5)	LR	DISTOT added to GDTOT (LR indicator is on). DISTOT and GDTOT printed out.

Example 2: Figure 114, insert A shows the use of AN and OR entries to group lines of indicators. When indicators 01, 02, 03 and 04 are on, or when indicators 01, 02, 03 and 05 are on, the calculation will be performed.

Example 3: Figure 114, insert B illustrates a case in which three additional conditions will cause the L4 total calculations to be performed: 01 and 02 are on, but not 03; or 01 and 03 are on, but not 02; or 02 and 03 are on but not 01.

COLUMNS 9-17 (INDICATORS)

Entry	*Explanation*
Blank	Operation is performed for every record read if columns 7-8 are not L0 or L1-L9 or SR.
01-99	Resulting indicators used elsewhere in the program.
L1-L9	Control level indicators previously assigned.
LR	Last record indicator.
MR	Matching record indicator.
H1-H9	Halt Indicators assigned elsewhere.
U1-U8	External indicators previously set.
OA-OG, OV	Overflow indicator previously assigned.

Use columns 9-17 to assign indicators that control when an operation is or is not to be done. You may use from one to three indicators on a line. By using AN or OR entries in columns 7-8, many indicators can be used to condition one operation. A maximum of seven AN or OR lines in any combination are allowed.

There are three separate fields (9-11, 12-14, and 15-17) on each line, one for each indicator. If the indicator must not be on in order to condition the operation, place an *N* before the appropriate indicator (columns 9, 12, 15).

All three indicators on one line are in an AND relationship with each other. The indicators on one line, or indicators in grouped lines, plus the control level indicator (if used in columns 7-8) must all be exactly as specified before the operation is done (see *Examples, Example 1*).

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page __ of __ | Program Identification

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators (Plus / Minus / Zero)	Comments
01	C		Ø1 Ø2 Ø3									
02	C	AN	Ø4									
03	C	OR	Ø1 Ø2 Ø3									
04	C	AN	Ø5	FIELDA	SUB	FIELDB	QTY	4	Ø		23 (Minus)	

Ⓐ

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page __ of __ | Program Identification

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators	Comments
01	C	L4	Ø1 Ø2 NØ3									
02	C	OR	Ø1 NØ2 Ø3									
03	C	OR	NØ1 Ø2 Ø3	SUM	ADD	SUMTOT	SUMTOT	8	2	H		

Ⓑ

Figure 114. Use of AND/OR Lines for Indicators

Indicators are used as follows in columns 9-17:

- Use any record identifying indicators previously specified in columns 19-20 on the Input Sheet to condition an operation that is to be done only for a certain type of record (see *Examples, Example 1*).

- Use any field indicators previously specified in columns 65-70 on the Input Sheet to condition an operation that is to be done only after the status of a field has been checked and has met certain conditions (see *Examples, Example 3*).

- Use any resulting indicators specified in columns 54-59 on the Calculation Sheet to condition operations according to the results of previous calculation operations (see the example in *Columns 54-59* in this chapter).

- Use any halt indicators previously used in columns 65-70 on the Input Sheet or in columns 54-59 on the Calculation Sheet to prevent the operation from being done when a specified error condition has been found in the input data (see *Columns 19-20* in Chapter 7) or on previous calculations. This is necessary because the record that causes the halt condition will be completely processed before your program stops. Thus, if the operation is performed even on an error condition, the results are in error. It is also possible to use a halt indicator to condition an operation that is to be done only when an error occurs.

- Use the matching record (MR) indicator to condition an operation that is to be done only when matching records have been found.

- Use any external indicator, including any previously specified in columns 71-72 on the File Description Sheet, to condition which operations should be done and which files should be used for a specific job.

- To condition operations to be performed at end of job, use the last record (LR) indicator in columns 9-17 only if LR is turned on during calculations. If LR is off during calculations, then all operations to be performed at end of job should be conditioned by LR in columns 7-8.

  Note: If LR is used as a resulting indicator, then NLR should be used as a conditioning indicator to avoid setting off LR.

- Use any control level indicators specified in columns 59-60 on the Input Sheet, or in columns 54-59 on the Calculation Sheet. If control level indicators are used in these columns instead of in columns 7-8, the operation is performed on only the first record of a new control group at detail calculations time.

- Use any overflow indicators previously specified in columns 33-34 on the File Description Sheet to condition operations that are to be done when overflow occurs. See *Columns 33-34* in Chapter 4 for a discussion of overflow.

The relationship between columns 7-8 and columns 9-17 is as follows:

- When a control level indicator (L1-L9) is specified in columns 7-8 and MR is specified in columns 9-17, MR indicates the matching condition of the previous record and not the one just read that caused the control break. After all operations conditioned by control level indicators (specified in columns 7-8 of the Calculation Sheet) are done, MR then indicates the matching condition of the record just read.

- When a control level indicator is used in columns 9-17 and columns 7-8 are not used, the operation conditioned by the indicator is done only on the record that causes that control break or any higher level control break.

- In one program cycle all operations conditioned by control level indicators in columns 7-8 are done before operations that are conditioned by control level indicators in columns 9-17 (see *Examples, Example 4*).

Examples

Example 1: Figure 115 shows the use of control level indicators to condition calculation operations. The operation in line 02 may be done when the L2 indicator is on, provided indicator 10 is on and L3 is not on.

The operation conditioned both by L2 and NL3 is done only when a control level 2 break occurs. These two indicators are used together because this operation is not to be done when a control level 3 break occurs, even though L2 is also on.

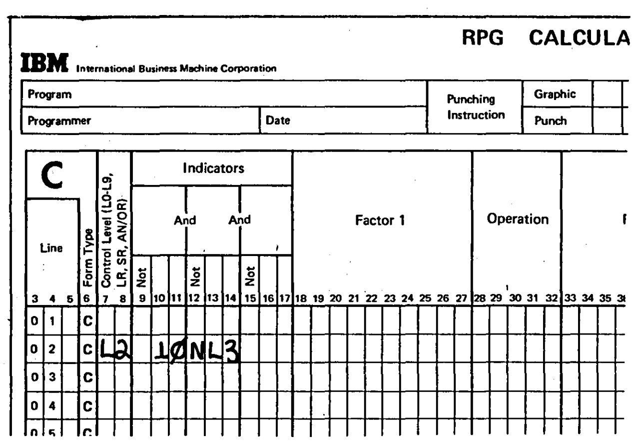


Figure 115. Conditioning Calculations (Control Level Indicators)

Example 2: Figure 116 shows how a record identifying indicator is used to condition an operation. When a record is read that has a *T* in column 1, the 01 indicator turns on. If this indicator is on, the field named SAVE is added to SUM. When a record having no *T* in column 1 is read, the 02 indicator is on. The subtract operation, since it is conditioned by 02, is then done instead of the add operation.

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position (21-24)	Not (N) (25)	C/Z/D (26)	Character (27)	2: Position (28-31)	Not (N) (32)	C/Z/D (33)	Character (34)	3: Position (35-38)	Not (N) (39)	C/Z/D (40)	Character (41)	Stacker Select (42)	P/B/L/R (43)	Field Location From (44-47)	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)	(71-74)
01	I	CARDS	AA			Ø1	1		C	T																					
02	I																				1Ø	15	2	SAVE							
03	I		BB			Ø2	1	N	C	T																					
04	I																				1Ø	15	2	SAVE							
05	I																														

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR, AN/OR) (7-8)	Indicators: Not (9)	(10-11)	And: Not (12)	(13-14)	And: Not (15)	(16-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators (Arithmetic: Plus / Minus / Zero; Compare: 1>2 / 1<2 / 1=2; Lookup (Factor 2) is: High / Low / Equal) (54-59)	Comments (60-74)
01	C			Ø1					SUM	ADD	SAVE	SUM	8	2			
02	C			Ø2					SUM	SUB	SAVE	SUM	8	2			
03	C																
04	C																
05	C																
06	C																
07	C																
08	C																
09	C																
10	C																

Record identifying indicators 01 and 02 are assigned on the input specifications. They are used here to condition calculation operations.

Figure 116. Conditioning Operations (Resulting Indicators)

Example 3: Figure 117 shows the use of field indicators to condition operations. Assume the job is to find weekly earnings including overtime. The overtime field is checked to see if any overtime has been put in. If the employee has worked overtime, the field is positive and indicator 10 turns on. In all cases the weekly regular wage is calculated. However, overtime pay is calculated only if indicator 10 is on (lines 02 and 03).

Example 4: Line 02 of Figure 118 shows the use of a control level indicator in columns 9-17. Assume that indicator 25 represents a record type and that a control level 2 break occurred when record type 25 was read. L1 and L2 are both on. All operations conditioned by the control level indicators in columns 7-8 are performed before operations conditioned by control level indicators in columns 9-17. Thus, the operation in line 03 occurs before the operation in line 02. The operation in line 02 is done on the first record of the new control group indicated by 25, whereas the operation in line 03 is a total operation done for all records of the previous control group.

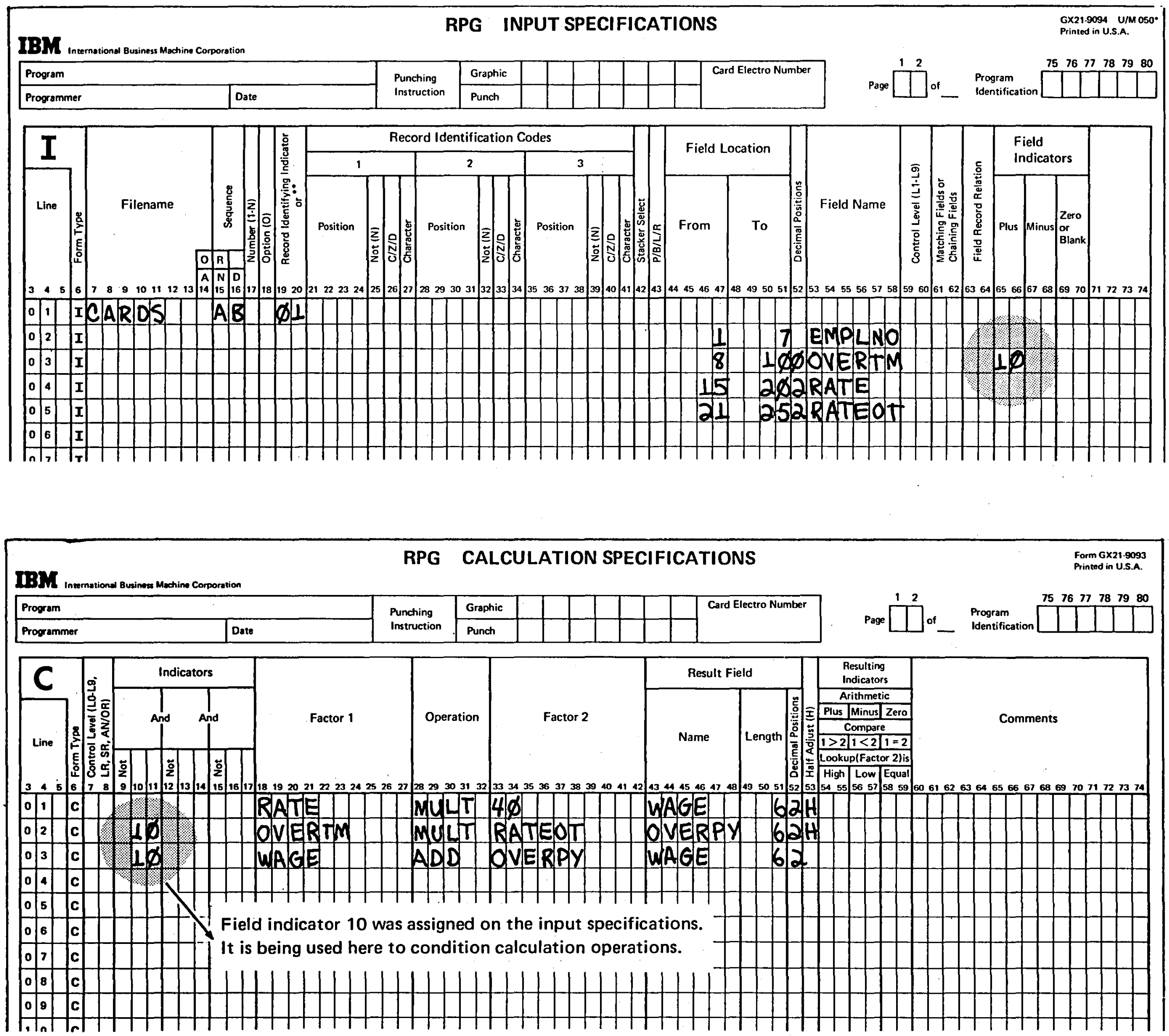


Figure 117. Conditioning Operations (Field Indicator)

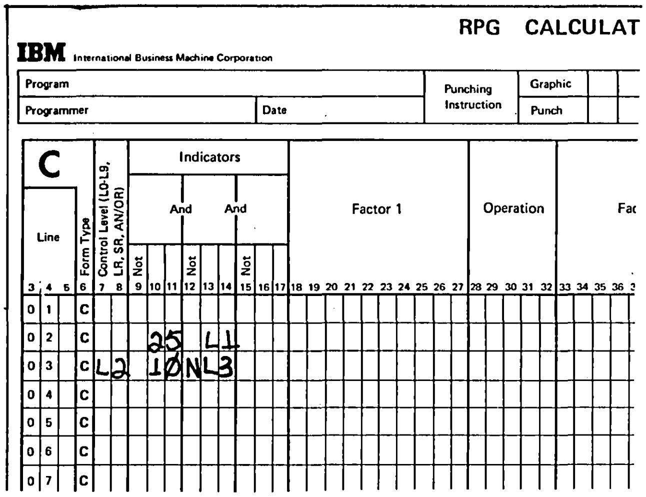


Figure 118. Conditioning Calculations (Control Level Indicators)

COLUMNS 18-27 (FACTOR 1) AND COLUMNS 33-42 (FACTOR 2)

Use columns 18-27 and 33-42 to name the fields or to give the actual data (literals) on which an operation is to be performed. The entries you can use are:

1. The name of any field that has been defined.

2. Any alphameric or numeric literal.

3. Any subroutine, table array name, or array element.

4. Any date field names (UDATE, UMONTH, UDAY, UYEAR).

5. The special names, PAGE, PAGE1, or PAGE2.

6. A label for a TAG, BEGSR, or ENDSR operation (Factor 1 only). A label for a GOTO or EXSR operation (Factor 2 only).

7. A filename for a CHAIN, DEBUG, DSPLY, READ, SETLL, or FORCE operation (Factor 2 only).

An entry in Factor 1 must begin in column 18; an entry in Factor 2 must begin in column 33.

The entries you use depends upon the operation you are describing. Some operations need entries in both sets of columns, some need entries in only one, and some need no entries at all. See *Columns 28-32* in this chapter for more information on operation codes. If you are naming a subroutine, see *Subroutines* in this chapter.

Literals

A literal is the actual data used in an operation rather than the field name representing that data. A literal may be either alphameric or numeric.

Consider the following rules when using an alphameric literal (Figure 119, insert A):

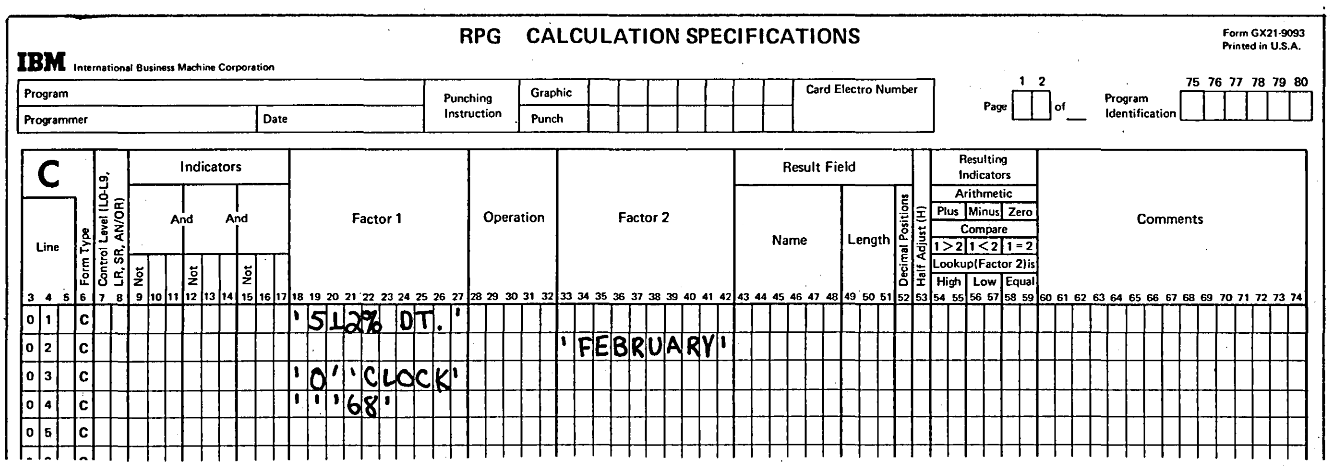
1. Any combination of characters may be used in an alphameric literal. Blanks are also valid.

2. Alphameric literals must be enclosed by apostrophes (').

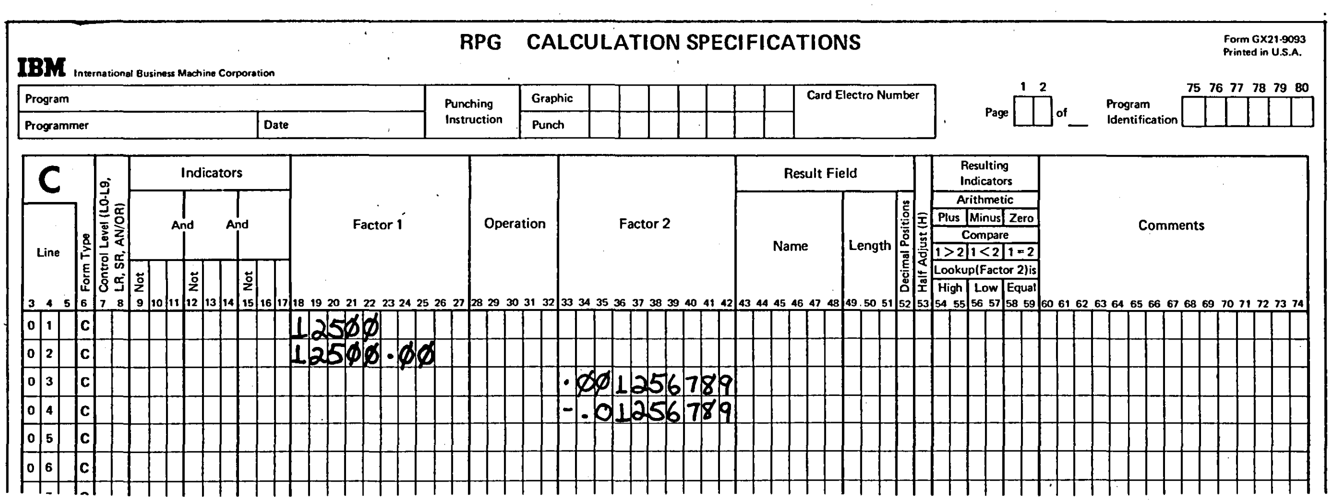
3. The maximum length of an alphameric literal is eight characters excluding the two enclosing apostrophes.

4. An apostrophe required as part of a literal is represented by two apostrophes. For example, the literal 'O'CLOCK' would be written as 'O''CLOCK'.

5. Alphameric literals may not be used for arithmetic operations.



Ⓐ



Ⓑ

Figure 119. Alphameric and Numeric Literals

Consider the following rules when using a numeric literal (Figure 119, insert B):

1. A numeric literal consists of any combination of the digits 0-9. A decimal point or sign may also be included.

2. The maximum total length of a literal is 10 characters including signs and decimal points.

3. Blanks may not appear in the literal.

4. The sign, if present, must be the leftmost character. An unsigned literal is treated as a positive number.

5. Numeric literals must not be enclosed by apostrophes (').

6. Numeric literals are used in the same way as a numeric field.

7. Decimal comma or decimal period is controlled by the Inverted Print option on the Control Record (see Chapter 3, *Column 21*).

COLUMNS 28-32 (OPERATION)

Use columns 28-32 to specify the kind of operation to be performed using Factor 1, Factor 2, and/or the Result Field and resulting indicators. The operation code must begin in column 28. A special set of operation codes have been defined which you must use to indicate the type of operation desired. Every operation code used requires certain entries on the same specification line. See *Appendix D*, Table D-1 for a summary of all possible codes and the additional entries required for each code. For further information on the operations that can be performed, see *Operation Codes* in this chapter.

The operations are performed in the order specified on the Calculation Sheet.

All operations conditioned by control level indicators in columns 7-8 must follow those that are not conditioned by control level indicators. All operations which are part of a subroutine (SR in column 7-8) must follow all other calculations in a program.

COLUMNS 43-48 (RESULT FIELD)

Entry	*Explanation*
Result Field	Field, table, array, or array element.

Use columns 43-48 to name the field, table, array, or array element that will hold the result of the operation specified in columns 28-32. You may use the name of a field, table, array, or array element that has already been defined either on extension specifications, input specifications, or elsewhere in the calculation specifications. (See *Tables and Arrays* in Chapter 5 for more information on arrays.)

Otherwise you may define a new field by entering a field name that has not already been used. Any field you define here will be created at the time the program is compiled. The field you name may be either numeric or alphameric. A field used in arithmetic operations (see *Columns 28-32* in this chapter) or numeric compare, or a field edited or zero suppressed in output specifications must be numeric.

The result field name must begin with an alphabetic character in column 43 and contain no blanks or special characters.

If you are entering the name of a field that has not been defined elsewhere, columns 49-52 should also contain entries.

If you are entering the name of a field that has been defined, entries in columns 49-52 are not necessary but if specified must agree with the previous definition of that field.

COLUMNS 49-51 (FIELD LENGTH)

Entry	*Explanation*
Blank	Alphameric or numeric field described elsewhere.
1-256	Result Field length.

Use columns 49-51 to give the result field length for any result field. If you are naming a new field (one that has not been used before), you must consider the form your data will be in and the length it will have after the operation has been performed.

Whenever the field length is specified for a result field, you should be careful to make the result field long enough to hold the largest possible result. If the result field is too small, significant digits may be lost. For example, you may wish to add field A (eight characters long, four decimal places) to field B (ten characters long, six decimal positions). Fields A and B have four characters to the left of the decimal, but the result field, field C, must allow for more characters to the left of the decimal.

9999.0000	Field A
0001.111111	Field B
10000.111111	Field C (result field)

In this case, field C was defined as 11 characters long with six decimal positions. Some of the numbers to the right of the decimal could be lost without changing the meaning of the result greatly. However, if field C were defined as 10 characters long with six decimal positions, a significant digit to the left of the decimal would be lost. Field C in this case would be 0000.111111 and the meaning of the result has greatly changed.

Numeric fields have a maximum length of 15 characters. Alphameric fields may be up to 256 characters long. You may indicate the length of a field that has been previously described either in the input specifications or in calculation specifications. However, if you do so, you must specify the same field length and number of decimal positions as was previously given to the field.

If the result field contains the name of a table or array, an entry in these columns is optional. If used, it must agree with the length described in the extension specifications.

COLUMN 52 (DECIMAL POSITIONS)

Entry	*Explanation*
Blank	Alphameric or numeric field described elsewhere.
0-9	Number of decimal places in a numeric result field.

Use column 52 to indicate the number of positions to the right of the decimal in a numeric result field. If the numeric result field contains no decimal positions, enter zero.

This column must be left blank if the result field is alphameric. It may also be left blank if the result field is numeric but has been previously described in the extension, input, or calculation specifications. In this case, Field Length (columns 49-51) must also be blank.

The number of decimal positions must never be greater than the length of the field. The number may, however, be larger or smaller than the number of decimal positions that actually result from an operation. If the number of decimal positions specified is greater than the number of decimal places that actually result from an operation, zeros are filled in to the right. If the number specified is smaller than the number that results from the operation, the rightmost digits are dropped.

Figure 120 shows how the contents of a result field after a multiplication operation may change according to the Decimal Positions (column 52) and Field Length (columns 49-51) specifications.

COLUMN 53 (HALF ADJUST)

Entry	*Explanation*
Blank	Do not half adjust.
H	Half adjust.

Use column 53 to indicate that the contents of the result field are to be half adjusted (rounded). In essence, half adjusting is done by adding a *5* (*-5* if the field is negative) to the number at the right of the last decimal position specified for this field. All decimal positions to the right of the position specified for that field are then dropped (see *Example*).

The half adjust entry is allowed only with arithmetic operations (see *Columns 28-32* in this chapter). This entry cannot be specified for an MVR operation, or for a DIV operation followed by an MVR operation.

Example

Figure 121 shows a result field being half adjusted to two decimal positions (*2* in column 52 and *H* in column 53). The result field is half adjusted as follows:

35.7968	Result of an add operation.
5	Add 5 to the number at the right of the last decimal position specified.
35.8018	Drop all decimal positions to the right at the position specified.
35.80	Result after half adjusting

Multiplication: 98.76 x 1.234 = 121.86984 *

Decimal Positions (column 52)	Field Length (columns 49-51)									
	10	9	8	7	6	5	4	3	2	1
9	1.869840000	.869840000								
8	21.86984000	1.86984000	.86984000							
7	121.8698400	21.8698400	1.8698400	.8698400						
6	0121.869840	121.869840	21.869840	1.869840	.869840					
5	00121.86984	0121.86984	121.86984*	21.86984	1.86984	.86984				
4	000121.8698	00121.8698	0121.8698	121.8698	21.8698	1.8698	.8698			
3	0000121.869	000121.869	00121.869	0121.869	121.869	21.869	1.869	.869		
2	00000121.86	0000121.86	000121.86	00121.86	0121.86	121.86	21.86	1.86	.86	
1	000000121.8	00000121.8	0000121.8	000121.8	00121.8	0121.8	121.8	21.8	1.8	.8
0	0000000121	000000121	00000121	0000121	000121	00121	0121	121	21	1

Not permitted

Permitted but inaccurate

Recommended

* A field length of 8 with 5 decimal positions gives all significant digits without adding zeros to either the left or right.

Figure 120. Result Field Contents Based on Various Field Length and Decimal Position Specifications

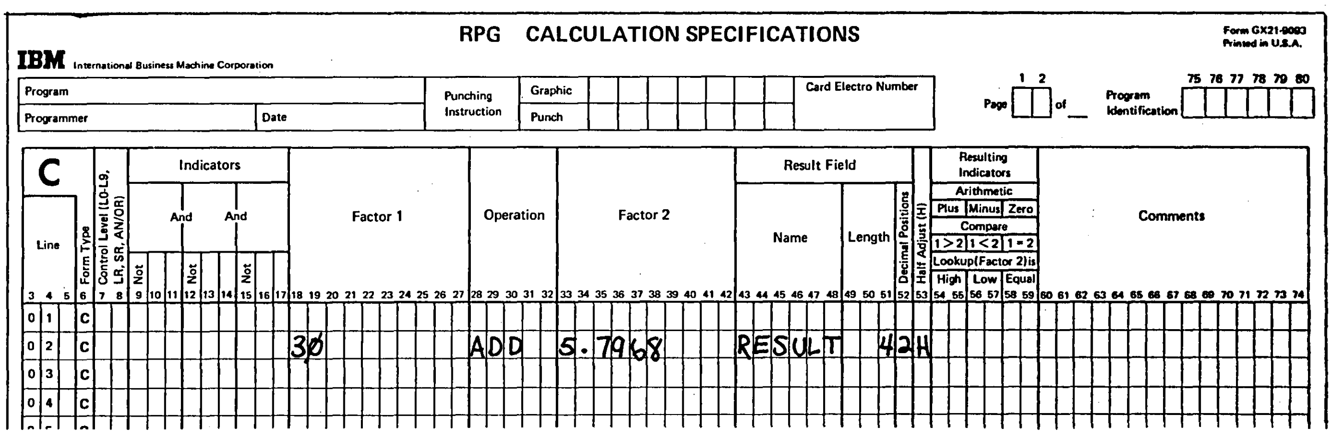


Figure 121. Specifying Half Adjust

COLUMNS 54-59 (RESULTING INDICATORS)

Entry	*Explanation*
01-99	Any numeric indicator.
H1-H9	Any halt indicator.
L1-L9	Any control level indicator.
LR	Last record indicator.
OA-OG, OV	Any overflow indicator (if specified on File Description Sheet).

Columns 54-59 are used for four different purposes:

1. To test the value of the result field after an arithmetic operation.
2. To check the outcome of a CHAIN, LOKUP, COMP, TESTB, or TESTZ operation (see *Operation Codes* in this chapter).
3. To specify which indicators to SETON or SETOF.
4. To indicate end of file for the READ operation code.

Test Results

By entering an indicator in columns 54-59, you specify that the result field is to be tested after the operation specified in columns 28-32 has been performed. (Normally, only indicators 01-99 and H1-H9 are used for testing.) The indicator specified is turned on only if the result field satisfies the condition being tested for (see *Examples, Examples 1-3*). This indicator may then be used to condition following calculations or output operations (see *Examples, Example 4*). If the same indicator is used to test the result of more than one operation, the operation last performed determines the setting of the indicator.

Notice that three fields (columns 54-55, 56-57, and 58-59) can be used for this purpose. Each field is used to test for different conditions: columns 54-55, plus or high; columns 56-57, minus or low; columns 58-59, zero or equal. You can test for more than one of the conditions.

Columns 54-55 (Plus or High): Place an indicator in these columns when testing to find:

1. If the Result Field in an arithmetic operation is positive.
2. If Factor 1 is higher than Factor 2 in a compare operation.
3. If Factor 2 is higher than Factor 1 in a table or array lookup operation.
4. The results of a CHAIN (not found), TESTB (all 0's), or TESTZ (*C* zone) operation.

Columns 56-57 (Minus or Low): Place an indicator in these columns when testing the Result Field to find:

1. If the Result Field in an arithmetic operation is negative.
2. If Factor 1 is lower than Factor 2 in a compare operation.
3. If Factor 2 is lower than Factor 1 in a table or array lookup operation.
4. The results of a TESTB (mixed), or TESTZ (*D* zone) operation.

Columns 58-59 (Zero or Equal): Place an indicator in these columns when testing the Result Field to find:

1. If the Result Field in an arithmetic operation is zero.
2. If Factor 1 is equal to Factor 2 in a compare operation.
3. If Factor 2 is equal to Factor 1 in a table or array lookup operation.
4. The results of a READ (end of file), TESTB (all ones), or TESTZ (not *C* or *D* zone) operation.

Note: If the LR indicator is used as a resulting indicator, NLR ("not LR") in columns 9-17 should be used as a conditional indicator to avoid setting off the LR indicator.

Setting Indicators

You may enter the indicators that you want to turn on or off by the operations SETON or SETOF. See *Operation Codes, Setting Indicators* in this chapter for more information on these operations. Any indicators to be turned on or off by the SETON or SETOF operations are specified from left to right in the three resulting indicators fields (Figure 122). Column headings in columns 54-59 have no meaning for SETON, or SETOF operations.

01-99 (Field Indicators, Record Identifying Indicators, Resulting Indicators, and Conditioning Indicators)

You may assign any of the numbers 01-99 to indicate such things as:

1. The type of record read (see *Columns 19-20* in Chapter 7).
2. The status (plus, minus, zero/blank) of an input field (see *Columns 65-70* in Chapter 7).
3. The results of a calculation operation. See *Examples, Example 1* and *Example 2.*

Any of these indicators which you have assigned may then also be used to:

1. Condition calculation operations (see *Columns 9-17* in this chapter).
2. Condition output operations (see *Columns 23-31* in Chapter 9).
3. Establish field record relations (see *Columns 63-64* in Chapter 7).

Indicators reflect only one condition at a time. When one indicator is used to reflect two or more conditions, it is always set to reflect the condition in the last operation performed. Therefore, it is not usual practice to assign the same number as a field indicator and/or resulting indicator more than once in a program. When you use such an indicator to condition other operations, you may get wrong results since the indicator may not always reflect the condition you think it does (see *Examples, Example 3*).

If any indicator 01-99 is set on or off by the operation codes SETON or SETOF, it remains on or off until an instruction in a specification line containing that same indicator is performed. The indicator is then set to reflect a condition from the operation performed.

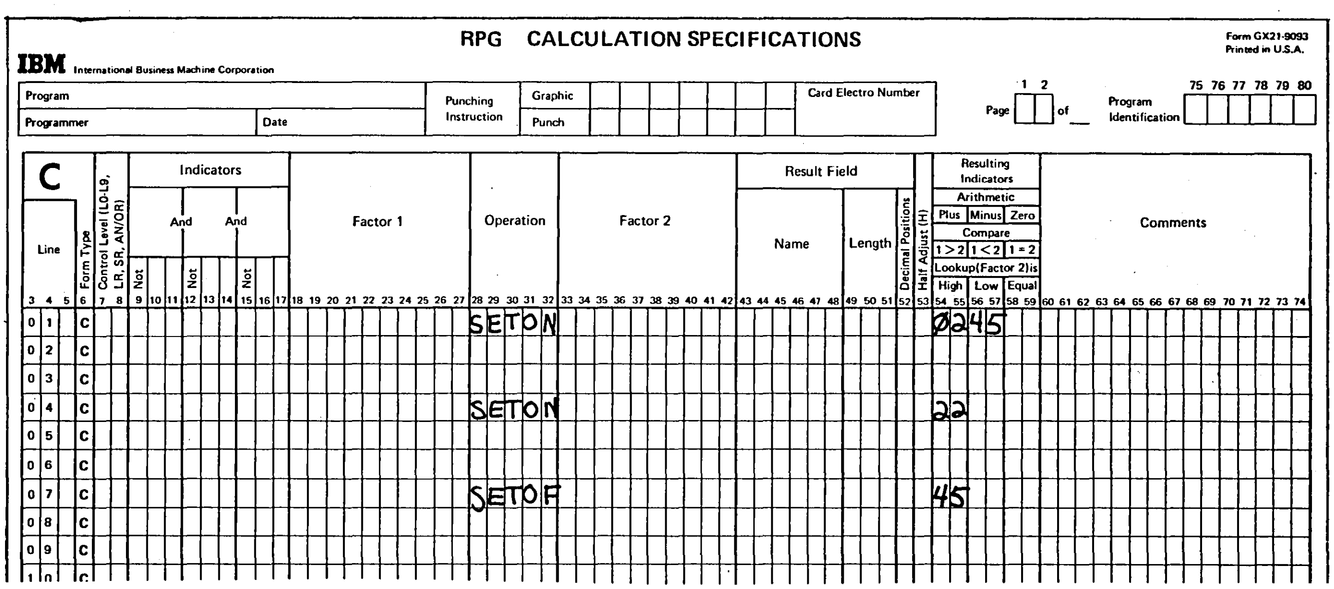


Figure 122. Setting Indicators

H1-H9 (Halt Indicators)

You may use any halt indicator to:

1. Cause the program to stop after finding an unacceptable condition.

2. Condition calculation or output operations that are not to be performed when such an unacceptable condition has occurred. This is necessary because all calculation and detail output operations are still performed for the record that caused the error before processing stops.

3. Establish field record relations (see *Columns 63-64* in Chapter 7).

Using the same indicator to test for two or more error conditions is not usually good practice (see *Examples, Example 5*).

Any assigned halt indicator is off at the beginning of the program.

All halt indicators that are turned on during a cycle are displayed at the end of detail output. Each halt must be responded to separately. After all halt indicators are displayed, an HO halt is displayed. All halt indicators are turned off when option 0 is selected in response to the HO halt.

Examples

Example 1: Figure 123, insert A shows that resulting indicator 10 has been assigned to signal when a minus condition occurs. Indicator 10 turns on if the result after the subtraction operation has been performed is negative. It then remains on (or off depending upon the result) until the same operation is performed again. It is always set to reflect the result of the subtraction operation each time it is done.

Example 2: Figure 123, insert B shows the same operation as insert A. However, this operation is conditioned by indicator 01. The operation is done only when indicator 01 is on. Resulting indicator 10 is set on only when the result of the operation is negative.

Example 3: Figure 123, insert C shows the use of the same indicator (10) in two lines. The status of this indicator reflects the result of each operation. For instance, indicator 10 turns on after the operation in line 05 has been done if the result of the operation is negative. However, if the result of the operation in line 07 is positive or zero, indicator 10 turns off. It is then reset only when the operation in line 05 is done again.

Note: Halt indicators are valid when used with the LR indicator. Even though the halt condition may be satisfied, the halt indicator is never displayed because the end of job occurs before the halt can occur.

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page 1 2

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust	Resulting Indicators Plus	Minus	Zero	
01	C			SALES	SUB	RETRNS	NETSLS	9	2			10		(A)
02	C													
03	C		01	SALES	SUB	RETRNS	NETSLS	9	2			10		(B)
04	C													
05	C			SALES	SUB	RETRNS	NETSLS	9	2			10		(C)
06	C			NETSLS	ADD	TOTSLS	TOTSLS	10	2					
07	C			PROFIT	SUB	RTRPRF	NETPRF	8	2			10		
08	C													
09	C													

Figure 123. Indicators 01-99

Example 4: Figure 124 shows the entry of two indicators that are used to test for the different conditions in a compare operation. These indicators are used to condition the calculations which might be performed for a payroll job. Indicator 10 is turned on if the hours worked (HRSWKD) are greater than 40 and is then used to condition all operations necessary to find overtime pay. Indicator 20 is turned on if HRSWKD is less than 40. It is also used to condition other operations. In line 03 if 20 is not on (the employee worked 40 or more hours), regular pay based on a 40 hour week is calculated. In line 06 if 20 is on (employee worked less than 40 hours), pay based on less than a 40 hour week is calculated.

IBM International Business Machine Corporation — RPG CALCULATION SPECIFICATIONS — Form GX21-9093 Printed in U.S.A.

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators High (1>2)	Low (1<2)	Equal (1=2)	Comments
01	C			HRSWKD	COMP	40					10	20		
02	C		10	HRSWKD	SUB	40	OVERTM	3	0					
03	C		N20	PAYRAT	MULT	40	PAY	6	2	H				
04	C		10	OVERTM	MULT	OVRRAT	OVRPAY	6	2	H				
05	C		10	OVRPAY	ADD	PAY	PAY							
06	C		20	PAYRAT	MULT	HRSWKD	PAY							
07	C													
08	C													

Figure 124. Conditioning Operations (Resulting Indicators)

Example 5: Figure 125, insert A shows the use of H1 in two different specification lines. If the result of the calculation operation in line 01 is negative, H1 turns on. This is an error condition. Processing continues, however, until this program cycle is completed. Thus, the operation in line 03 is done. If the result of this subtraction operation is positive, H1 turns off. The program does not stop because H1 is not on, even though an error condition has been found in line 01.

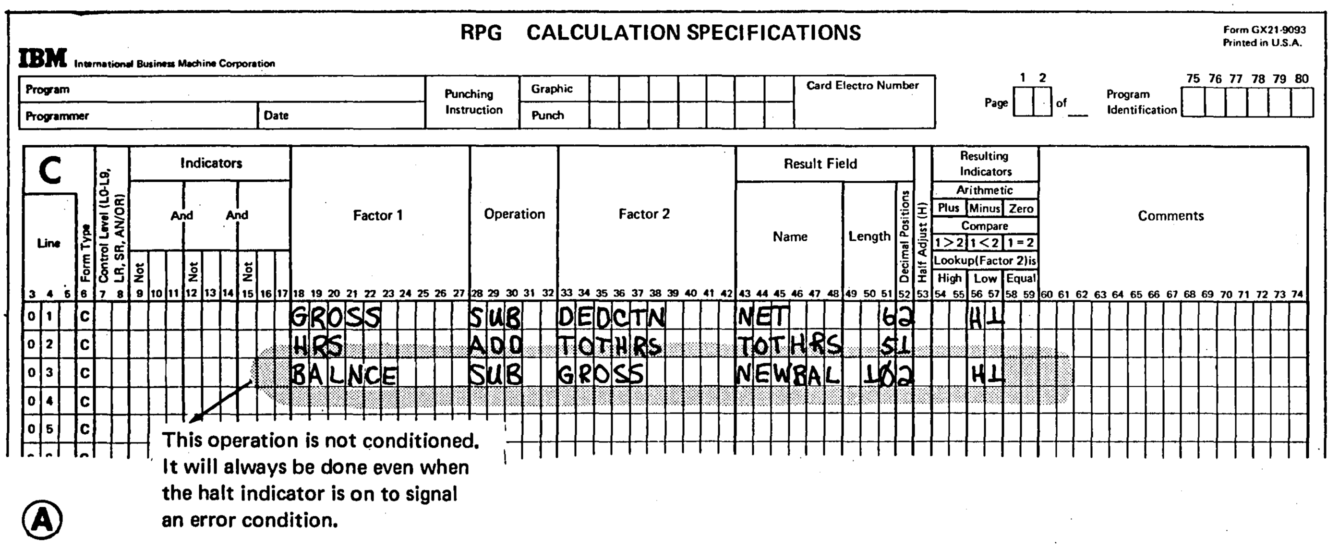
The use of two different halt indicators as shown in Figure 125, insert B does not allow a situation like the one just described to occur.

COLUMNS 60-74 (COMMENTS)

Enter in columns 60-74 any meaningful information you wish. The comments you use should help you understand or remember what you are doing on each specification line. Comments are not instructions to the RPG II program. They serve only as a means of documenting your program.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

See Chapter 2.



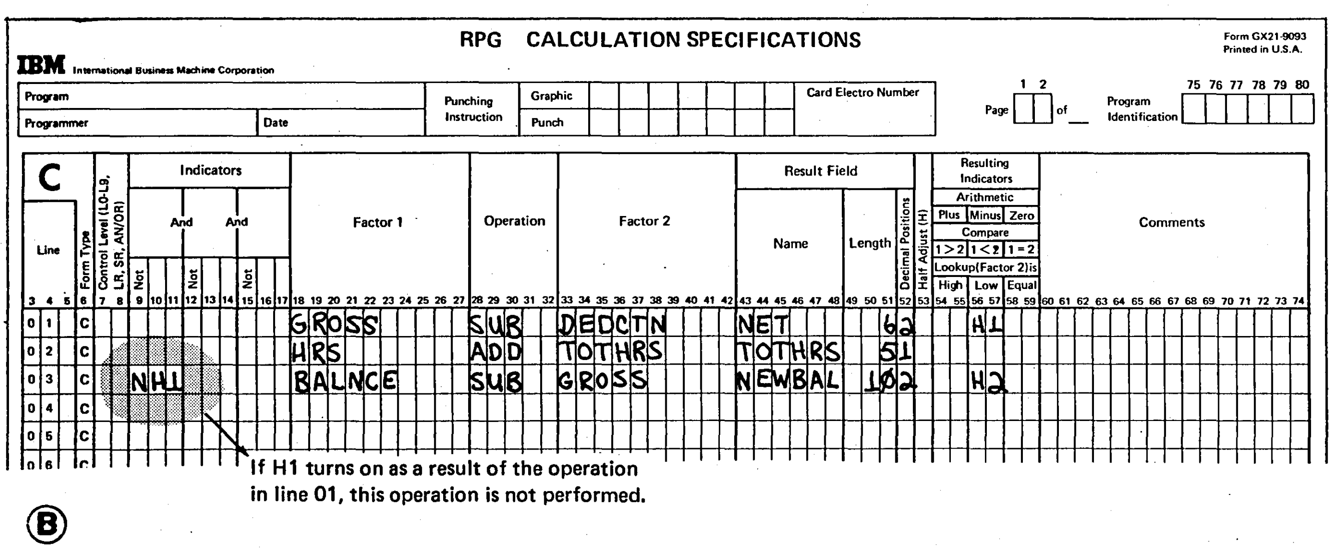


Figure 125. One Halt Indicator Testing for Two Error Conditions

Operation Codes

You are able to perform many different types of operations on your data using the RPG II language. Special codes have been set up which indicate the operation to be performed. Usually these are just abbreviations of the name of the operation. You must use these codes to specify the operation to be performed.

Operations may be divided into nine categories; all codes in each category are explained in this section. Examples are also given for many codes. *Appendix D,* Table D-1 provides a summary of the operation codes. It also shows what other specifications need to be used with each code.

ARITHMETIC OPERATIONS

Arithmetic operations can be performed only on numeric fields or literals. The result field must also be numeric. For arithmetic operations in which all three fields are used:

1. Factor 1, Factor 2, and the Result Field may all be different fields.

2. Factor 1, Factor 2, and the Result Field may all be the same field.

3. Factor 1 and Factor 2 may be the same field but different from the Result Field.

4. Either Factor 1 or Factor 2 may be the same as the Result Field.

The length of any field involved in an arithmetic operation cannot exceed 15 characters. If the result exceeds 15 characters, characters may be dropped from either or both ends depending on the location of the decimal point. The results of all operations are signed (+,–). Any data placed in the result field replaces the data that was there previously.

Add (ADD)

Factor 2 is added to Factor 1. The sum is placed in the Result Field. Factor 1 and Factor 2 are not changed by the operation.

Zero and Add (Z-ADD)

Factor 2 is added to a field of zeros, and the sum is placed in the Result Field.

Subtract (SUB)

Factor 2 is subtracted from Factor 1. The difference is placed in the Result Field. Factor 1 and Factor 2 are not changed by the operation.

Note: Subtracting two fields which are the same is a method of setting the result field to zero.

Zero and Subtract (Z-SUB)

Factor 2 is subtracted from a field of zeros. The difference is placed in the Result Field. This actually places the negative of Factor 2 in the Result Field. This operation can be used to change the sign of a field. Factor 1 is not used.

Multiply (MULT)

Factor 1 is multiplied by Factor 2. The product is then placed in the Result Field. Factor 1 and Factor 2 are not changed. When you use (as a factor) a field which is described as the Result Field, you must be sure the Result Field is large enough to hold the product.

Divide (DIV)

Factor 1 (dividend) is divided by Factor 2 (divisor). The result (quotient) is placed in the Result Field. Factor 1 and Factor 2 are not changed.

If Factor 1 is 0, the result of the divide operation will be 0. Factor 2 cannot be 0. If it is, the job stops immediately and a halt code is displayed (see *RPG II Halt Procedures* in Appendix A). You may continue processing, by resetting the halt. When processing is continued, the result and remainder are set to zero.

Any remainder resulting from the divide operation is lost unless the move remainder operation is specified as the next operation. If move remainder is the next operation, the result of the divide operation cannot be half adjusted (rounded).

Move Remainder (MVR)

This operation moves the remainder from the previous divide operation to a separate field named under Result Field. Factor 1 and Factor 2 must not be used. This operation must immediately follow the divide operation and should be conditioned by the same indicators. Half adjust cannot be specified with this operation. The maximum length of the remainder is 15, including decimal positions. The number of significant decimal positions is the greater of:

1. The number of decimal positions in Factor 1 of the previous divide operation.
2. The sum of the decimal positions in Factor 2 and the Result Field of the previous divide operation.

The maximum whole number positions in the remainder is equal to the whole number positions in Factor 2 of the previous divide operation.

Figure 126 shows the use of the move remainder operation.

Square Root (SQRT)

This operation derives the square root of the field named in Factor 2. The square root of Factor 2 is placed in the Result Field. Factor 1 is not used.

Factor 2 and the Result Field can be numeric fields up to fifteen digits long overall, including up to nine decimal places. Figure 127 is a table which can be used to determine Result Field contents for various field lengths and decimal positions.

For every digit left of the decimal place in the Result Field, there should be two digits left of the decimal place in Factor 2; for every digit right of the decimal place in the Result Field, there should be two digits right of the decimal place in Factor 2.

A whole array can be used in a SQRT operation if Factor 2 and Result Field contain array names. In this case, the square root of each element of the array named in Factor 2 will be placed in the corresponding element of the array named in the Result Field.

When using the SQRT operation, remember:

1. The Result Field (root) is automatically half-adjusted.
2. The Result Field length must be greater than or equal to the decimal positions entry.
3. Factor 2 cannot be a negative number. A negative number causes a halt (see *RPG II Halt Procedures* in Appendix A).
4. Resulting indicators are not allowed.

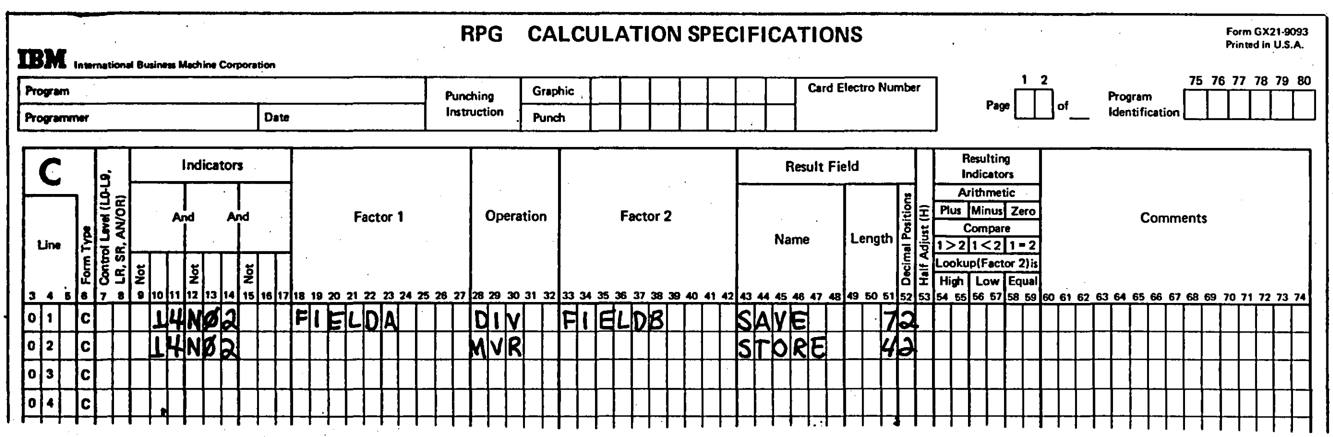


Figure 126. Move Remainder Operation

Field Length

Decimal Positions	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
9	0	0	0	0	6	3	8	1	1	2	0	5	9	1	2
8	0	0	0	0	0	6	3	8	1	1	2	0	5	9	1
7	0	0	0	0	0	0	6	3	8	1	1	2	0	5	9
6	0	0	0	0	0	0	0	6	3	8	1	1	2	0	6
5	0	0	0	0	0	0	0	0	6	3	8	1	1	2	1
4	0	0	0	0	0	0	0	0	0	6	3	8	1	1	2
3	0	0	0	0	0	0	0	0	0	0	6	3	8	1	1
2	0	0	0	0	0	0	0	0	0	0	0	6	3	8	1
1	0	0	0	0	0	0	0	0	0	0	0	0	6	3	8
0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	4

Numbers in the table represent the square root of 4071.87.

Notes:

1. Shaded areas are decimal positions.
2. To find the Result Field contents for any field length and decimal positions, read all digits on the desired decimal positions line which are below and to the right of the desired field length. For example:

   Field length = 8; decimal positions = 4

   Result Field contents = 0063.8112

Figure 127. Result Field Contents for Various Field Lengths and Decimal Positions

Crossfoot (XFOOT)

This operation is used only on arrays with numeric elements. It adds all the elements of the array together and puts the sum into a separate field specified as the Result Field. Factor 1 is not used. Factor 2 contains the name of the array. You can half-adjust the total in the Result Field and use resulting indicators if you wish.

If the Result Field is an element of the same array used in Factor 2, the value of that element prior to the XFOOT operation is used in arriving at a total.

MOVE OPERATIONS

Move operations move part or all of Factor 2 to the Result Field. Factor 2 remains unchanged.

Factor 1 is not used in any move operations. It must always be blank. No resulting indicators may be used. Numeric fields may be changed to alphameric fields and alphameric fields may be changed to numeric fields by the move operations. 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.

When move operations are specified to move data into numeric fields, decimal positions are ignored. For example, if the data 1.00 is moved into a numeric field with one decimal position, the result is 10.0.

Move (MOVE)

This operation causes characters from Factor 2 to be moved to the rightmost positions in the result field. Moving starts with the rightmost character.

If Factor 2 is longer than the Result Field, the excess leftmost characters of Factor 2 are not moved. If the Result Field is longer than Factor 2, the characters to the left of the data just moved in are unchanged.

An alphameric field or constant may be changed into a numeric field by moving it into a numeric field. When this is specified, the digit portion of each character is converted to its corresponding numeric character and then moved to the result field. Blanks are transferred as zeros. However, the zone portion of the rightmost alphameric character is converted to a corresponding sign and is moved to the rightmost position of the numeric field where it becomes the sign of the field. A numeric field may also be changed into an alphameric field by moving it into an alphameric field. All digits are transferred. The digit and zone of the rightmost character are transferred. The MOVE operation is summarized in Figure 128.

When moving data into an indexed array, only one element in the array has data moved into it. This is true even if the field being moved is larger than the array element.

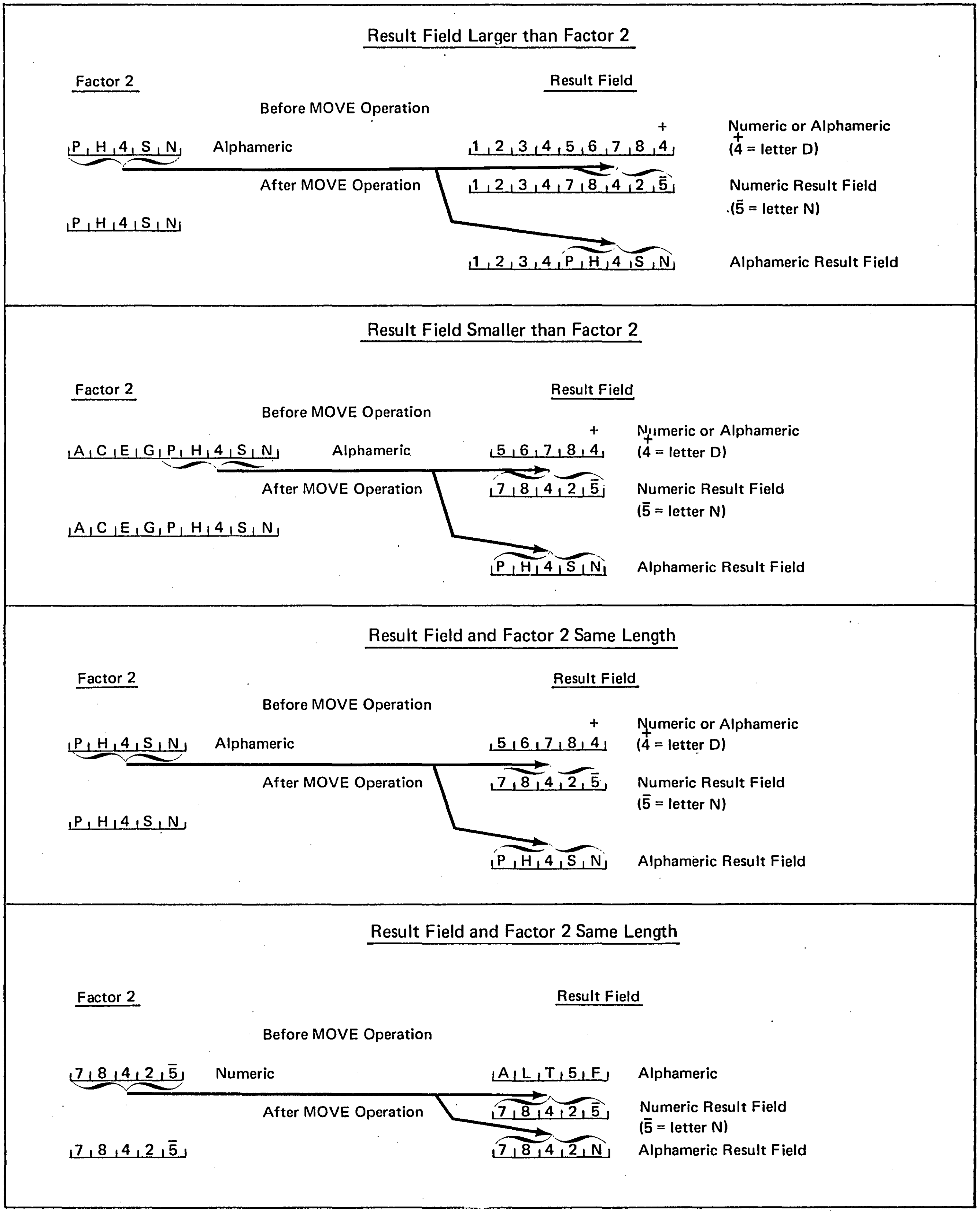


Figure 128. MOVE Operations

Move Left (MOVEL)

This operation causes characters from Factor 2 to be moved to the leftmost position in the Result Field. Moving begins with the leftmost character.

If Factor 2 is longer than the Result Field, the excess rightmost characters of Factor 2 are not moved. If the Result Field is longer than Factor 2, the characters to the right of the data just moved in are unchanged. In this case the sign of a numeric field is not changed either.

An alphameric field or constant may be changed into a numeric field by moving it into a numeric field. When this is specified, the digit portion of each character is converted to its corresponding numeric character and then moved into the result field.

Blanks are transferred as zeros. If the rightmost character is moved, the zone is also converted and used as the sign of the field. When the rightmost character is not transferred, the zone is, nevertheless, still transferred and used as the sign of the result field.

A numeric field may also be changed into an alphameric field by moving it into an alphameric field. All digits are transferred. Both digit and zone portions of the rightmost character are transferred if that character is to be moved.

A summary of rules for MOVEL transfers are as follows (see also Figure 129):

1. Factor 2 is the same length as the Result Field.

   a. Factor 2 and Result Field numeric: the sign is moved with the rightmost digit.
   b. Factor 2 numeric, Result Field alphameric: the sign is moved with the rightmost digit. Only digits are moved for other positions.
   c. Factor 2 alphameric, Result Field numeric: zone and digit portions of rightmost digit are moved. Zones in other positions are not moved.
   d. Factor 2 and Result Field alphameric: all characters are moved.

2. Factor 2 is longer than the Result Field.

   a. Factor 2 and Result Field numeric: the sign from the rightmost position of factor 2 is moved over the rightmost digit of the result field.
   b. Factor 2 numeric, Result Field alphameric: the Result Field contains only digits.
   c. Factor 2 alphameric, Result Field numeric: zone from the rightmost character of Factor 2 is moved over the rightmost digit of the Result Field; other Result Field positions contain only digits.
   d. Factor 2 and Result Field alphameric: only the number of characters needed to fill the Result Field are moved.

3. Factor 2 is shorter than the Result Field.

   a. Factor 2 either numeric or alphameric, Result Field numeric: digit portion of Factor 2 replaces the contents of the leftmost positions in the Result Field. The sign in the rightmost position of the Result Field is not changed.
   b. Factor 2 either numeric or alphameric, Result Field alphameric: characters in Factor 2 replace the equivalent number of leftmost positions in the Result Field. No change is made in the zone of the rightmost position of the Result Field.

Factor 2 and Result Field Same Length

	Factor 2		Result Field	
a. Numeric	7 8.4 2 $\bar{5}$	Before MOVEL Operation	5 6 7.8 $\overset{+}{4}$	Numeric
	7 8.4 2 $\bar{5}$	After MOVEL Operation	7 8 4.2 $\bar{5}$	
b. Numeric	7 8.4 2 $\bar{5}$	Before MOVEL	A K T 4 D	Alphameric
	7 8.4 2 $\bar{5}$ ($\bar{5}$ = letter N)	After MOVEL	7 8 4 2 N	
c. Alphameric	P H 4 S N	Before MOVEL	5 6 7 8 $\overset{+}{4}$	Numeric
	P H 4 S N	After MOVEL	7 8 4 2 $\bar{5}$	
d. Alphameric	P H 4 S N	Before MOVEL	A K T 4 D	Alphameric
	P H 4 S N	After MOVEL	P H 4 S N	

Factor 2 Longer Than Result Field

	Factor 2		Result Field	
a. Numeric	0 0 0 0 0 8 4 2 $\bar{5}$	Before MOVEL Operation	5·6 7 8 $\overset{+}{4}$	Numeric
	0 0 0 0 0 8 4 2 $\bar{5}$	After MOVEL Operation	0·0 0 0 $\bar{0}$	
b. Numeric	9 0 3 1 7 8 4 2 $\bar{5}$	Before MOVEL	A K T 4 D	Alphameric
	9 0 3 1 7 8 4 2 $\bar{5}$	After MOVEL	9 0 3 1 7	
c. Alphameric	B R W C X H 4 S N	Before MOVEL	5 6 7 8.$\overset{+}{4}$	Numeric
	B R W C X H 4 S N	After MOVEL	2 9 6 3.$\bar{7}$	
d. Alphameric	B R W C X H 4 S N	Before MOVEL	A K T 4 D	Alphameric
	B R W C X H 4 S N	After MOVEL	B R W C X	

Factor 2 Shorter Than Result Field

		Factor 2		Result Field	
a.	Numeric	7 8 4 2 $\bar{5}$	Before MOVEL Operation	1.3 0 9 4 3 2 1 $\overset{+}{0}$	Numeric
		7 8 4 2 $\bar{5}$	After MOVEL Operation	7.8 4 2 5 3 2 1 $\overset{+}{0}$	
	Alphameric	C P T 5 N	Before MOVEL	1 1 3 0 9 4 3 2 1 $\bar{0}$	Numeric
		C P T 5 N	After MOVEL	3 7 3 5 5 3 2 1 $\bar{0}$	
b.	Numeric	7 8 4 2 $\bar{5}$	Before MOVEL	B R W C X H 4 S A	Alphameric
		7 8 4 2 $\bar{5}$	After MOVEL	7 8 4 2 N H 4 S A	
	Alphameric	C P T 5 N	Before MOVEL	B R W C X H 4 S A	Alphameric
		C P T 5 N	After MOVEL	C P T 5 N H 4 S A	

Figure 129. MOVEL Operations

Move Array (MOVEA)

This operation moves the data starting from the leftmost position of Factor 2 to the leftmost position of the Result Field. The shorter field (Factor 2 or the Result Field) determines the length of the move. If Factor 2 is longer than the Result Field, the excess rightmost characters of Factor 2 will not be moved. If the Result Field is longer than Factor 2, the characters to the right of the data moved into the Result Field will remain unchanged. All arrays and fields referenced by a MOVEA instruction must be alphameric.

The MOVEA operation makes it possible to move:

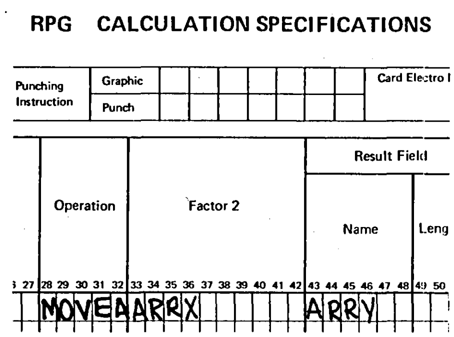
- Several contiguous array elements to a single field.
- A single field to several contiguous array elements.
- Contiguous elements of one array to contiguous elements of another array.

The movement of data starts with the first element of an array or field. If the array is indexed, the move starts with the element referenced. The movement of data is terminated when the last array element has been moved or filled, or when the number of characters moved equals the length of the shorter field specified in Factor 2 or the Result Field. This may cause the move to terminate in the middle of an array element.

Note: Both Factor 2 and the Result Field cannot reference the same array.

The MOVEA is summarized in Figure 130.

Example: Array to array move. No indexing; different length arrays, same element length.



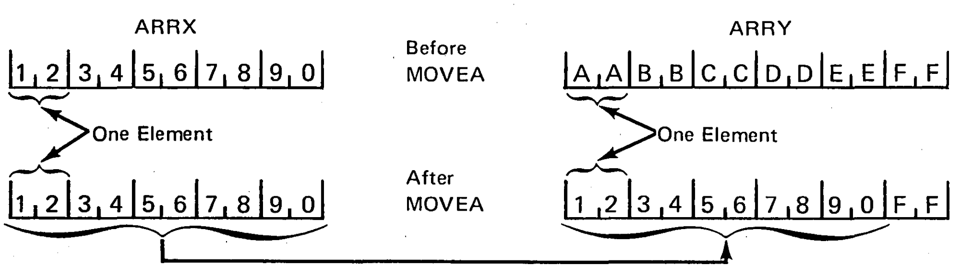
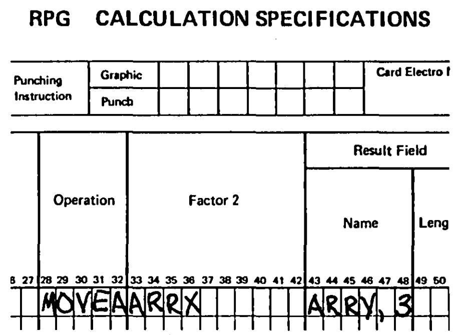
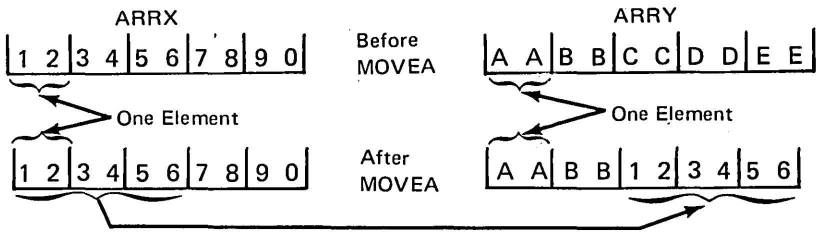


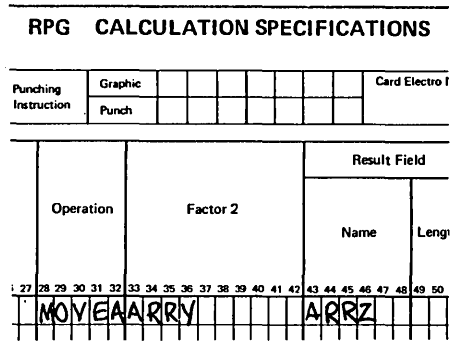
Figure 130 (Part 1 of 3). MOVEA Operation

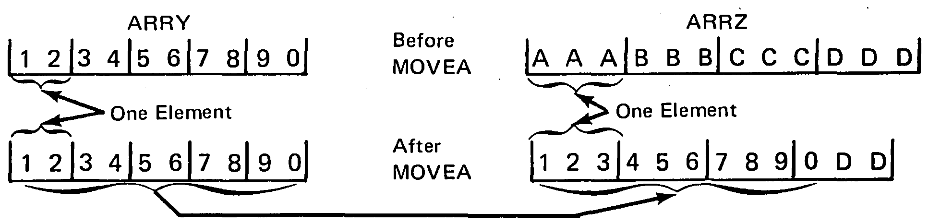
Example: Array to array move. Index result field.



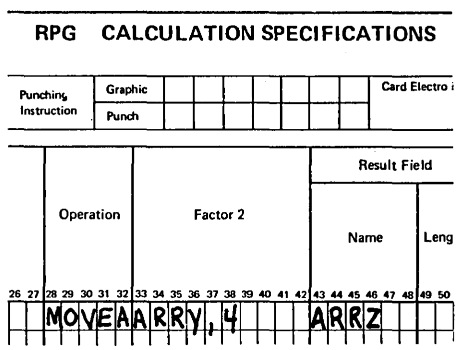


Example: Array to array move. No indexing, different length array elements.





Example: Array to array move. Index factor 2, different length array elements.



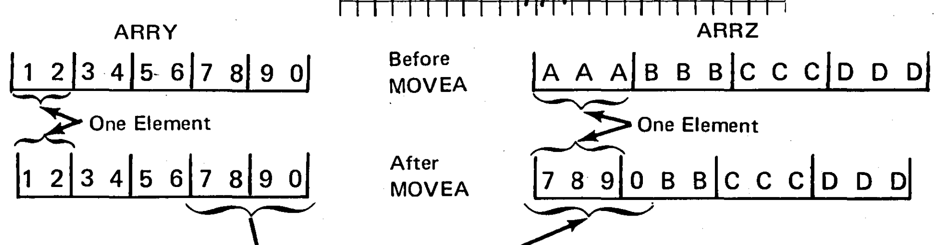
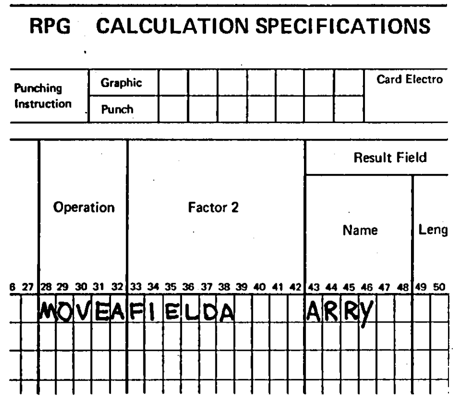
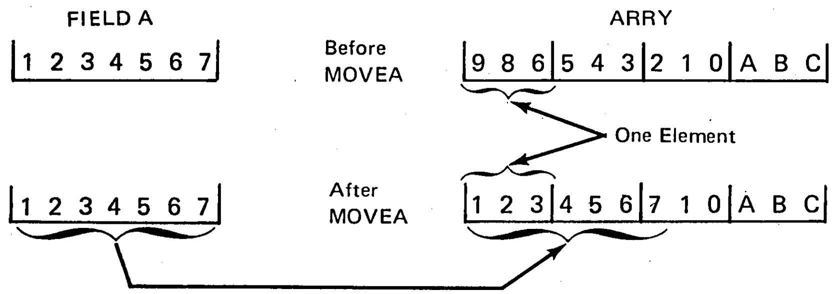


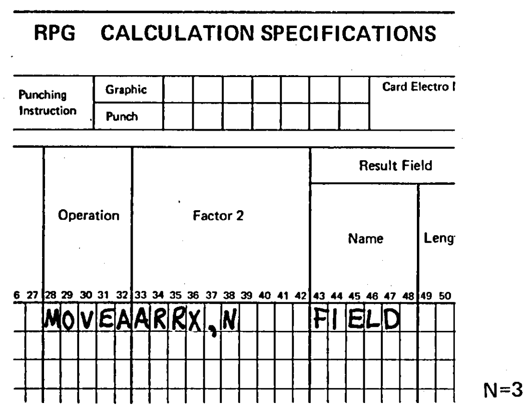
Figure 130 (Part 2 of 3). MOVEA Operation

Example: Field to array move.
No indexing on array.





Example: Array to field move.
Variable indexing.



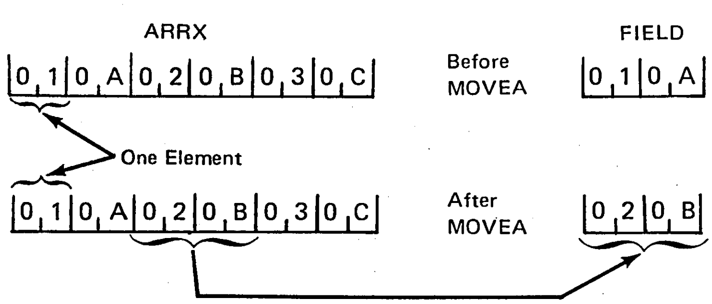


Figure 130 (Part 3 of 3). MOVEA Operation

MOVE ZONE OPERATIONS

These operations are used only to move the zone portion of a character. There are four varieties of the move zone operation (Figure 131).

Using a minus (-) sign in a move zone operation will not yield a negative character in the result field, since minus is represented by a X'60' internally and a D zone is required for a negative character. Characters J-R have D zone representations and can be used to obtain a negative value (J = X'D1', . . . ,R = X'D9').

Note: Generally, whenever the word high is used, the field involved must be alphameric; whenever low is used, the field involved may be either alphameric or numeric.

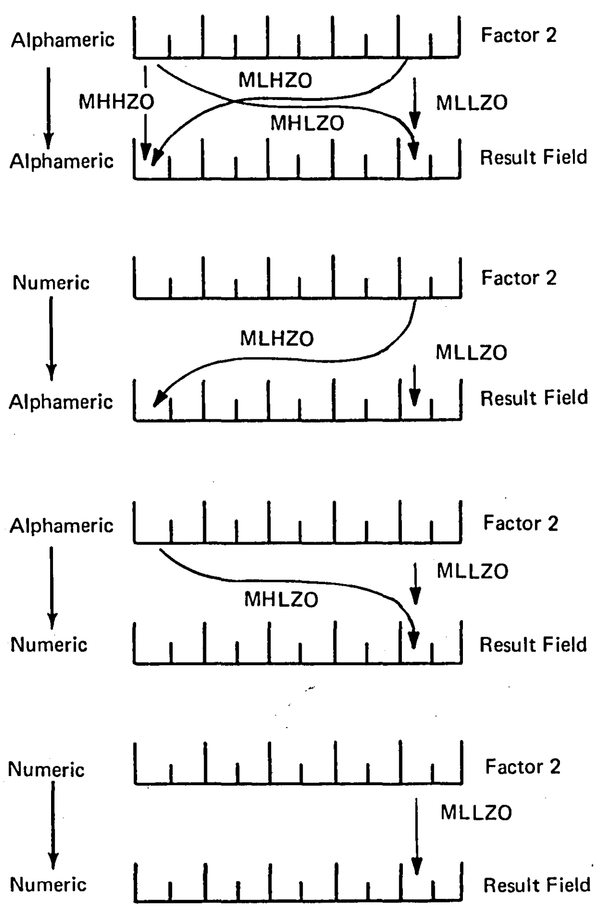


Figure 131. Function of Move Zone Operations

Move High to High Zone (MHHZO)

This operation moves the zone from the leftmost position of Factor 2 to the leftmost position of the Result Field. Factor 2 and the Result Field must be alphameric.

Move High to Low Zone (MHLZO)

This operation moves the zone from the leftmost position of Factor 2 to the zone of the rightmost position of the Result Field. Factor 2 can be only alphameric. The Result Field may be either alphameric or numeric.

Move Low to Low Zone (MLLZO)

This operation moves the zone from the rightmost position of Factor 2 to the zone of the rightmost position of the Result Field. Factor 2 and the Result Field may be either alphameric or numeric.

Move Low to High Zone (MLHZO)

This operation moves the zone from the rightmost position of Factor 2 to the leftmost position of the Result Field. Factor 2 can be numeric or alphameric, but the Result Field can only be alphameric.

COMPARE AND TESTING OPERATIONS

These operations test fields for certain conditions. The result of the test is shown by the resulting indicators assigned in columns 54-59. No fields are changed by these operations.

Compare (COMP)

This operation causes Factor 1 to be compared with Factor 2. As a result of the compare, indicators are turned on as follows:

High	Factor 1 is greater than Factor 2.
Low	Factor 1 is less than Factor 2.
Equal	Factor 1 equals Factor 2.

Factor 1 and Factor 2 must either be both alphameric or both numeric.

The fields are automatically aligned before they are compared. If the fields are alphameric, they are aligned to their leftmost character. If one is shorter, the unused positions are filled with blanks (Figure 132).

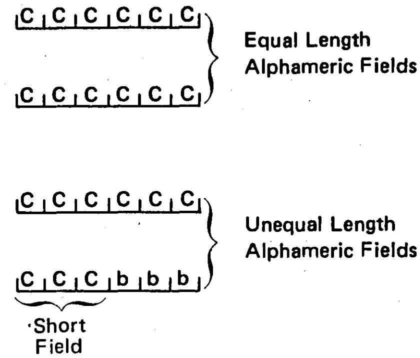


Figure 132. Comparison of Alphameric Fields

If the fields which are to be compared are numeric, they are aligned according to the decimal point. Any missing digits are filled in with zeros (Figure 133). The maximum field length for numeric fields which are to be compared is 15 digits.

If an alternate collating sequence is defined, alphameric fields are compared according to that sequence. Entire arrays cannot be used with the compare operation.

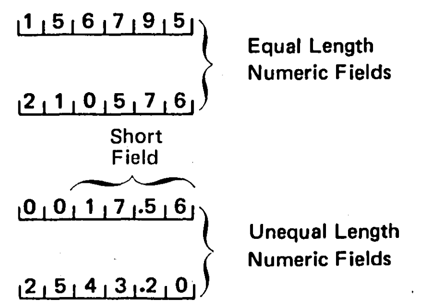
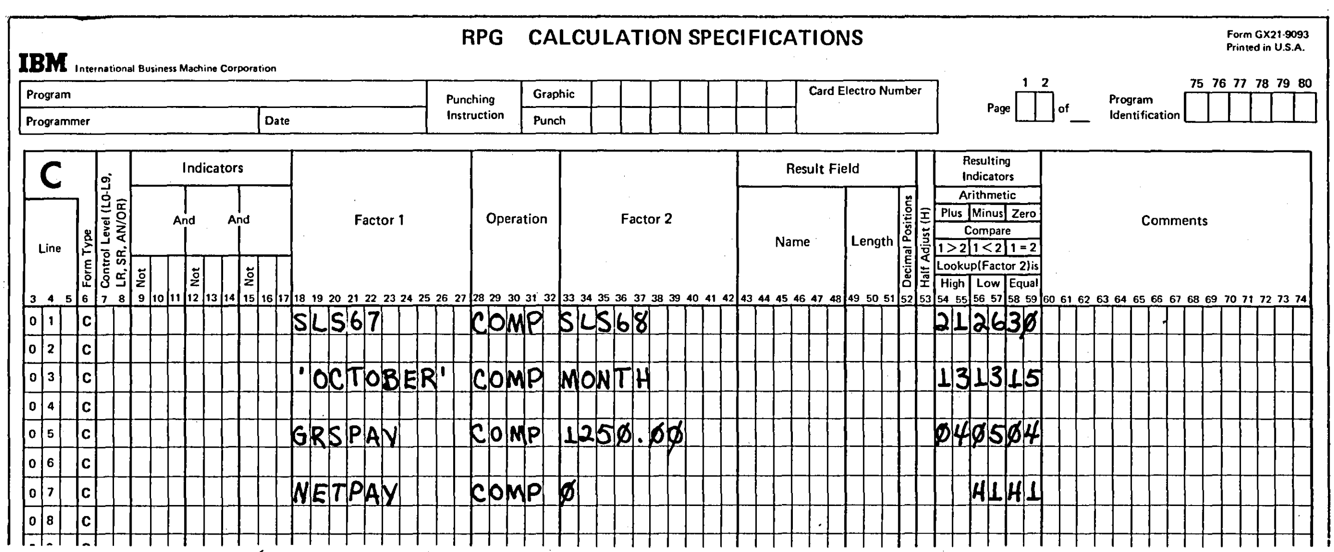


Figure 133. Comparison of Numeric Fields

Figure 134 shows some specifications for compare operations. In specification line 01, the contents of the field SLS67 (1967 sales) are compared with the contents of SLS68. If 1967 sales exceed 1968 sales, resulting indicator 21 turns on; if they are less, resulting indicator 26 turns on; if the two years had equal sales, 30 turns on. In line 03 the alphameric constant OCTOBER is compared against the contents of the field named MONTH (which must also be defined as alphameric). If the MONTH field does not contain the word OCTOBER, indicator 13 turns on; if it does, indicator 15 turns on after the compare operation. In line 05 the contents of the field named GRSPAY (which must be defined as numeric) is decimal-aligned with numeric


IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Line	Form Type	Factor 1	Operation	Factor 2	High (1>2)	Low (1<2)	Equal (1=2)
01	C	SLS67	COMP	SLS68	21	26	30
02	C						
03	C	'OCTOBER'	COMP	MONTH	13	13	15
04	C						
05	C	GRSPAY	COMP	1250.ØØ	Ø4	Ø5	Ø4
06	C						
07	C	NETPAY	COMP	Ø		H1	H1
08	C						

Figure 134. Compare Operations

constant 1250.00 and then compared to it. If the value in field GRSPAY is greater than or equal to 1250.00, indicator 04 turns on; if its value is less than 1250.00, indicator 05 turns on. In line 07 the contents of the field NETPAY (which must be defined as numeric) is decimal-aligned with numeric constant 0 and then compared to it. If NETPAY is greater than zero, indicator H1 remains off after the compare operation. If NETPAY is zero or negative, indicator H1 turns on.

Test Zone (TESTZ)

This operation tests the zone of the leftmost character in the result field (see *Character Structure* under *Columns 21-41* in Chapter 7). The Result Field must be alphameric since this operation can be done only on alphameric characters. Resulting indicators are used to determine the results of the test. The zone portion of characters *&* and *A-I* causes the plus indicator to turn on. The zone portion of the characters } (bracket), – (minus), and J-R causes the minus indicator to turn on. All other characters, when tested, cause the blank indicator to turn on. Factor 1 and Factor 2 are not used in this operation.

BIT OPERATIONS

Three operation codes, BITON, BITOF, and TESTB, are provided to set and test individual bits. The individual bits can be used as switches in a program.

All data fields are initialized by the system at the beginning of a job:

- Alphameric fields are initialized to hexadecimal '40'.
- Numeric fields are initialized to hexadecimal 'F0'.

If a program uses alphameric fields for bit operations and requires the fields to be initialized to binary zero (hexadecimal '00'), then the program must initialize those fields.

In binary field operations, the operation code, BITON, BITOF, or TESTB, must appear in columns 28-32. Factor 2 can contain:

- *Bit numbers 0-7:* One or more bits (maximum of eight) may be set on, set off, or tested per operation. The bits are numbered from left to right and are enclosed in apostrophes. The order of specification of the bits is not restricted. For example, to specify the first bit in a field, enter *'0'* in Factor 2 (in columns 33-35). To specify bits 0, 2, and 5, enter *'025'* in Factor 2 (in columns 33-37). Bits not specified in Factor 2 are not changed.
- *Field Name:* The name of a one-position, alphameric field or table or array element can be entered. In this case, the bits which are on in the field or array element are set on, set off, or tested in the Result Field; bits which are not on are not affected.

Any field named in Factor 2 or the Result Field must be a one-position, alphameric field (no entries in the decimal positions columns on the Input or Calculation Sheet).

Set Bit On (BITON)

This operation code causes bits identified in Factor 2 to turn on (set to one) in a field named as the Result Field. The operation code BITON must appear in columns 28-32. Conditioning indicators can be used in columns 7-17. Any entry under Field Length must be *1*. See the preceding discussion in *Bit Operations.*

Factor 1, Decimal Positions, Half-Adjust, and Resulting Indicators are not used with the BITON operation. See Figure 135 for a summary of BITON operations.

IBM International Business Machine Corporation — RPG CALCULATION SPECIFICATIONS — Form GX21-9093 Printed in U.S.A.

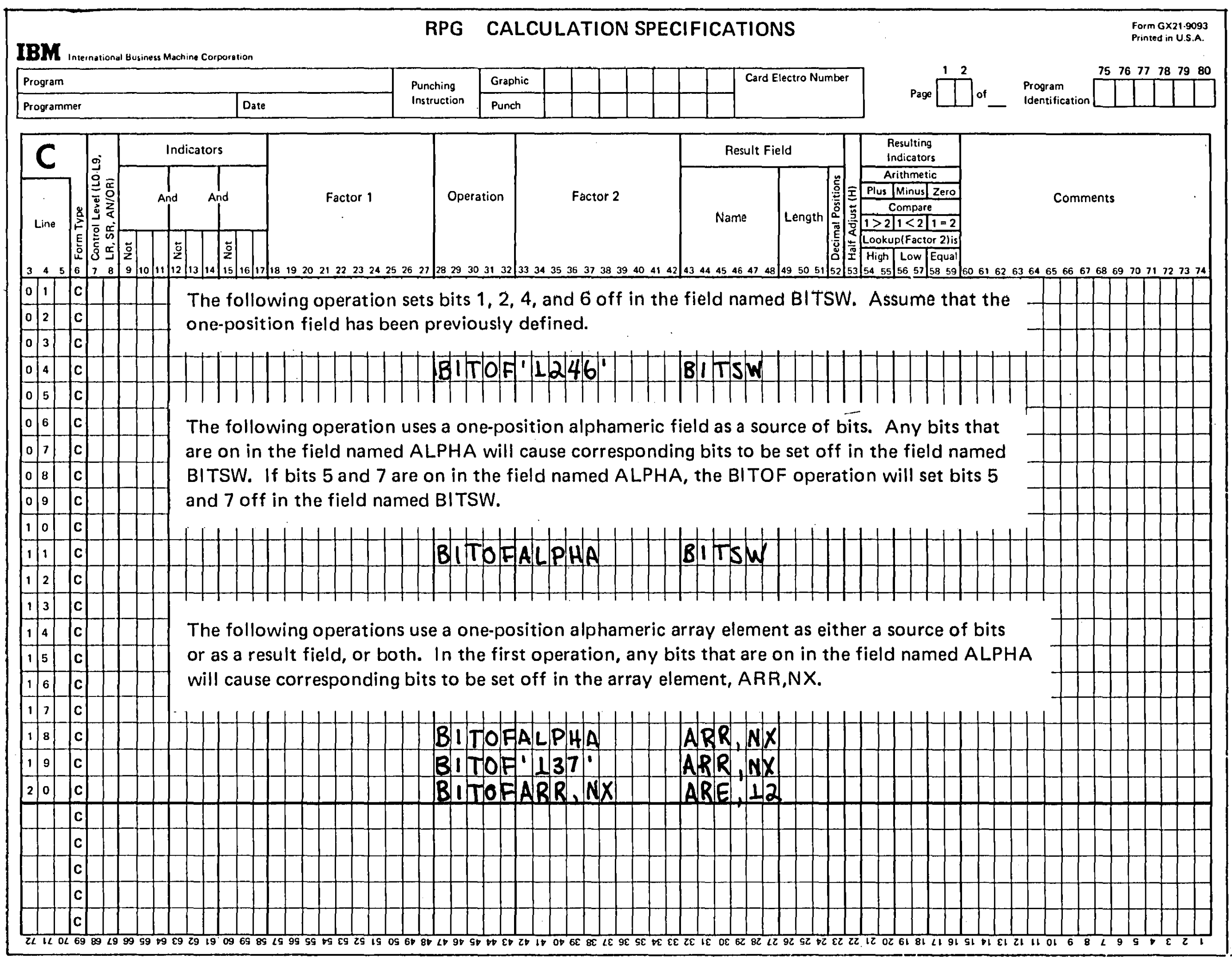
Line	Form Type	Factor 1	Operation	Factor 2	Result Field Name
01	C				
02	C	The following operation sets bits 0, 3, 5, and 7 on in the field named BITS. Assume that the one-position field has been previously defined.			
03	C				
04	C		BITON	'Ø357'	BITS
05	C				
06	C	The following operation uses a one-position alphameric field as a source of bits. Any bits that are on in the field named ALPHA will cause corresponding bits to be set on in the field named BITS. If bits 5 and 7 are on in the field named ALPHA, the BITON operation will set bits 5 and 7 on in the field named BITS.			
07	C				
08	C				
09	C				
10	C				
11	C				
12	C		BITON	ALPHA	BITS
13	C				
14	C	The following operations use a one-position alphameric array element as either a source of bits or as a result field, or both. In the first operation, any bits that are on in the array element, ARR,NX will cause corresponding bits to be set on in the array element ARE,LZ.			
15	C				
16	C				
17	C				
18	C		BITON	ARR,NX	ARE,LZ
19	C		BITON	'Ø246'	ARR,NX
20	C		BITON	ALPHA	ARR,NX

Figure 135. Set Bit On (BITON) Operations

Set Bit Off (BITOF)

This operation code causes bits identified in Factor 2 to turn off (set to zero) in a field named as the Result Field.

The operation code BITOF must appear in columns 28-32. All other specifications are the same as those for the BITON operation. See Figure 136 for a summary of BITOF operations.


IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Line	Form Type	Operation	Factor 1	Factor 2	Result Field
01	C	The following operation sets bits 1, 2, 4, and 6 off in the field named BITSW. Assume that the one-position field has been previously defined.			
04	C	BITOF		'1246'	BITSW
06	C	The following operation uses a one-position alphameric field as a source of bits. Any bits that are on in the field named ALPHA will cause corresponding bits to be set off in the field named BITSW. If bits 5 and 7 are on in the field named ALPHA, the BITOF operation will set bits 5 and 7 off in the field named BITSW.			
11	C	BITOF		ALPHA	BITSW
14	C	The following operations use a one-position alphameric array element as either a source of bits or as a result field, or both. In the first operation, any bits that are on in the field named ALPHA will cause corresponding bits to be set off in the array element, ARR,NX.			
18	C	BITOF		ALPHA	ARR,NX
19	C	BITOF		'137'	ARR,NX
20	C	BITOF		ARR,NX	ARE,12

Figure 136. Set Bit Off (BITOF) Operations

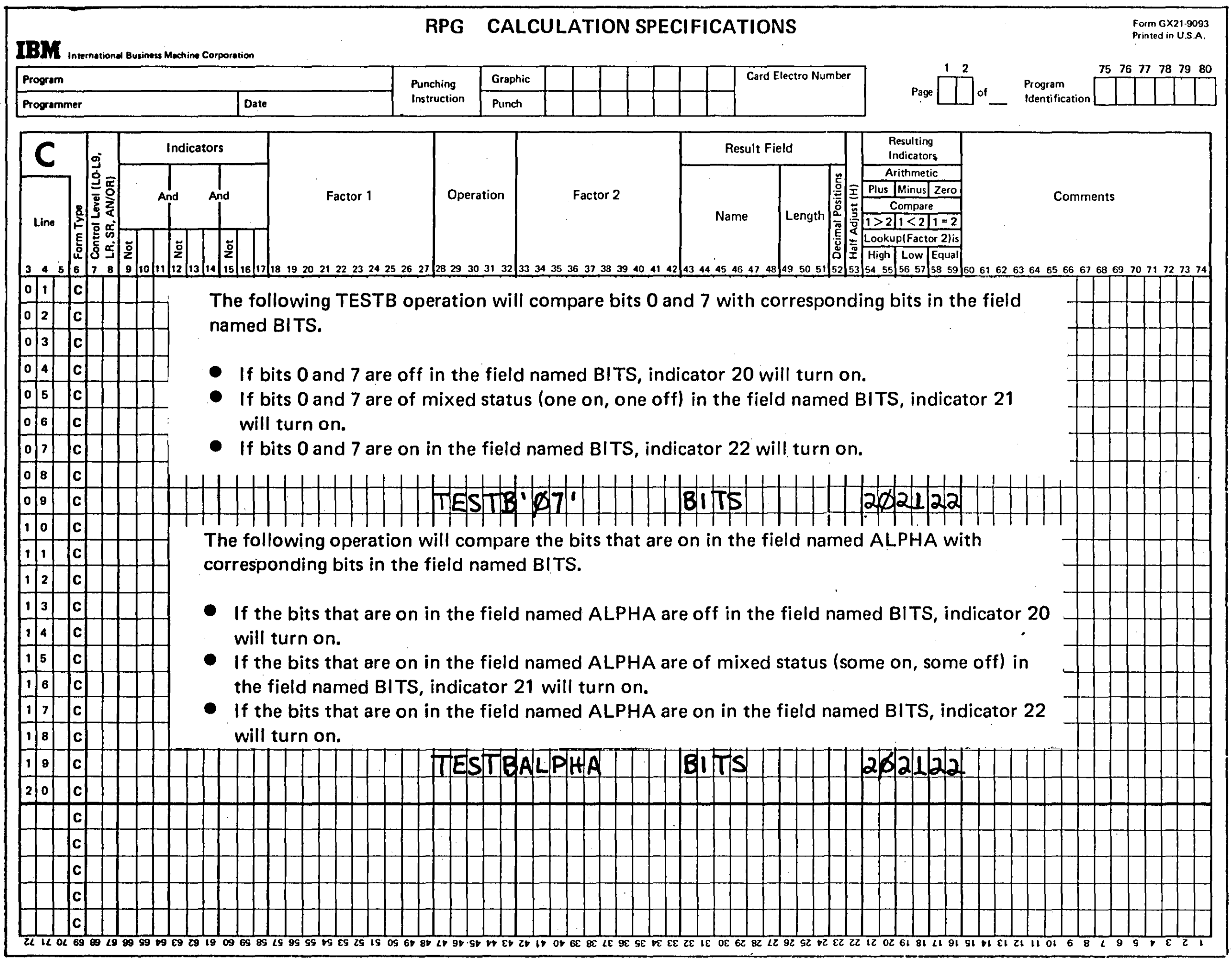
Test Bit (TESTB)

This operation code causes bits identified in Factor 2 to be tested for an on or off condition in the field named as the Result Field. The condition of the bits is known by resulting indicators in columns 54-59. All other specifications are the same as those for BITON and BITOF. See Figure 137 for a summary of TESTB operations.

At least one resulting indicator must be used with the TESTB operation; as many as three can be named for one operation. Two indicators may be the same for one TESTB operation, but not three. If a field specified in Factor 2 contains bits which are all off (binary 0), no resulting indicators are turned on. A resulting indicator has the following meanings for these columns:

- *Columns 54-55:* An indicator in these columns is turned on if each bit specified in Factor 2 is off (0) in the Result Field.
- *Columns 56-57:* An indicator in these columns is turned on if two or more bits were tested and found to be of mixed status; that is, some bits on and other bits off. It is the programmer's responsibility to ensure that the field named in Factor 2 contains more than one bit which is on if an indicator appears in columns 56-57.
- *Columns 58-59:* An indicator in these columns is turned on if each bit specified in Factor 2 is *on* (1) in the Result Field.

Note: If the field in Factor 2 has no bits on, then this indicator will be turned on.


IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093 Printed in U.S.A.

Line	Form Type	Operation	Factor 2	Result Field Name	Resulting Indicators
01–08	C				
09	C	TESTB	'07'	BITS	202122
10–18	C				
19	C	TESTB	ALPHA	BITS	202122
20	C				

The following TESTB operation will compare bits 0 and 7 with corresponding bits in the field named BITS.

- If bits 0 and 7 are off in the field named BITS, indicator 20 will turn on.
- If bits 0 and 7 are of mixed status (one on, one off) in the field named BITS, indicator 21 will turn on.
- If bits 0 and 7 are on in the field named BITS, indicator 22 will turn on.

The following operation will compare the bits that are on in the field named ALPHA with corresponding bits in the field named BITS.

- If the bits that are on in the field named ALPHA are off in the field named BITS, indicator 20 will turn on.
- If the bits that are on in the field named ALPHA are of mixed status (some on, some off) in the field named BITS, indicator 21 will turn on.
- If the bits that are on in the field named ALPHA are on in the field named BITS, indicator 22 will turn on.

Figure 137 (Part 1 of 2). Test Bit (TESTB) Operations

SETTING INDICATORS

These operation codes are used to turn indicators off or on. Any indicator to be turned on or off is specified in columns 54-59. The headings in the Resulting Indicators field (Plus or High, Minus or Low, Zero or Equal) have no meaning in these operations. When setting indicators, remember:

1. The following indicators may not be turned on by the SETON operation: 1P, MR, L0, U1-U8.

2. The following indicators may not be turned off by the SETOF operation: 1P, MR, LR, L0, U1-U8.

3. If the LR indicator is turned on by a SETON operation which is conditioned with a control level indicator (columns 7-8 of the Calculation Sheet), processing stops after all total output operations are finished. If it is turned on by a SETON operation not so conditioned, processing stops after the next total output operation is completed.

4. If the halt indicators (H1-H9) are set on and not turned off before the detail output operations are complete, the system stops. Processing may be continued by pressing the start key on the Processing Unit once for every halt indicator that is on.

5. Setting on or setting off a control level indicator (L1-L9) does not automatically set on the lower control level indicators.

6. Indicators L1-L9 and the record identifying indicators are always turned off after detail output operations are completed, regardless of the previous SETON or SETOF operation.

7. Whenever a new record is read, record identifying indicators (01-99) and field indicators are set to reflect conditions on the new record. The setting from any previous SETON or SETOF operation does not apply then.

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Program ____ Programmer ____ Date ____ Punching Instruction: Graphic / Punch ____ Card Electro Number ____ Page ☐ of __ Program Identification ____

The following operations use a one-position array element as either a source of bits or as a result field, or both. In the first operation, the bits that are on in the field named ALPHA are compared to corresponding bits in the array element, ARR,NX. For example, assume that bits 1 and 4 are on in the field named ALPHA:

- If bits 1 and 4 are off in array element ARR,NX, indicator 20 will turn on.
- If bits 1 and 4 are of mixed status (one on, one off) in array element ARR,NX, indicator 21 turns on.
- If bits 1 and 4 are on in array element ARR,NX, indicator 22 will turn on.

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust	Plus / High / 1>2	Minus / Low / 1<2	Zero / Equal / 1=2	Comments
01	C													
02	C													
03	C													
04	C													
05	C													
06	C													
07	C													
08	C													
09	C													
10	C													
11	C													
12	C				TESTB	ALPHA	ARR,NX				2Ø	21	22	
13	C													
14	C				TESTB	'24'	ARE,12				2Ø	21	22	
15	C													
16	C				TESTB	ARE,12	ARR,NX				2Ø	21	22	
17	C													
18	C													

Figure 137 (Part 2 of 2). Test Bit (TESTB) Operations

Set On (SETON)

This operation causes any indicators in columns 54-59 to be turned on.

Set Off (SETOF)

This operation causes any indicators in columns 54-59 to be turned off.

BRANCHING OPERATIONS

Operations are normally performed in the order that they appear on the Calculation Sheet. There may be times, however, when you do not want the operations performed in the order they are specified. For example, you may wish to:

1. Skip several operations when certain conditions occur.
2. Perform certain operations for several, but not all, record types.
3. Perform several operations over and over again.

Go To (GOTO)

This operation allows you to skip instructions by specifying some other instruction to go to (see *TAG*). You may branch to an earlier line or to a later specification line. However, you cannot skip from a calculation that is not conditioned by a control level indicator (columns 7-8) to one that is, or vice versa. Neither can you branch from a calculation within a subroutine to a calculation outside of that subroutine, or vice versa.

Factor 2 must contain the name of the point to which you wish to go (maximum of six characters). Factor 1 and the Result Field are not used in this operation. The GOTO operation may be conditioned by any indicators. If it is not conditioned, the operation is always done. See *Examples* for use of GOTO operations.

Tag (TAG)

This operation code names the point to which you are branching in the GOTO operation. Factor 1 contains this label (maximum of six characters). The name must begin in column 18. The same label may not be used for more than one TAG instruction.

Factor 2 and the Result Field are not used. No indicators may be entered in columns 9-17 for a TAG instruction. Control level indicators may be used, however, if branching is to occur at total time. See *Examples* for use of the TAG operation.

Examples

Example 1: Figure 138 shows how TAG and GOTO may be used to skip operations on certain conditions.

1. If the results of the subtraction in line 01 is minus (indicator 10 is on), a branch is taken to RTN1 (routine 1) named by the TAG operation code in line 09. Notice that both the GOTO (line 02) and TAG (line 09) are *not* conditioned by control level indicators.

2. If the branch is not taken in line 02, the multiplication in line 03 is performed. Then the branch to RTN1 (line 09) *must* be taken because this branch is not conditioned by indicators.

3. Operations in lines 10-12 are then done. If the operation in line 12 does not turn indicator 15 on, a branch is taken backwards to RTN2 (line 05).

4. Operations then go in the order specified again from lines 06-12. Nothing is done in line 09 since TAG only gives a name. These same operations are performed again and again until 15 does turn on.

5. When 15 is on, the branch to RTN2 is not taken. The TESTZ operation is then performed. If this operation causes 20 to turn on, a branch is taken to line 17 (GOTO END). If 20 is not on, the operation on line 16 is done.

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page of | Program Identification

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Plus (1>2, High)	Minus (1<2, Low)	Zero (1=2, Equal)	Comments
01	C			FIELDA	SUB	FIELDB	FIELDB	5	2			10		
02	C		10		GOTO	RTN1								
03	C			FIELDB	MULT	4	SAVE	8	2					
04	C				GOTO	RTN1								
05	C			RTN2	TAG									
06	C				Some Calculation									
07	C				Operations									
08	C													
09	C			RTN1	TAG									
10	C				Some Calculation									
11	C				Operations									
12	C												15	
13	C		N15		GOTO	RTN2								
14	C				TESTZ		FIELDC				20		20	
15	C		20		GOTO	END								
16	C				MHLZO	FIELDC	FIELDD							
17	C			END	TAG									
18	C													
19	C													
20	C													

Figure 138. Using GOTO and TAG (Skipping Operations)

Example 2: Figure 139 shows how TAG and GOTO may be used to eliminate coding when several operations have to be performed again and again.

Assume that you wish to make 20 mailing labels for every customer you have. The customer's name and address are found on an input card. Since you wish to write 20 labels for each card, you have to use exception lines and the operation EXCPT (see *EXCPT Operation* in this section for further information).

This can be coded as shown in Figure 139, insert A. You have to write the EXCPT operation code for every mailing label. However, by using branching, you can code it all in six lines (see Figure 139, insert B). An EXCPT line is printed out. One is added to COUNT in order to keep track of how many times the line has been printed. Then COUNT is compared to 20. If COUNT does not equal 20, a branch is taken back to the beginning (GOTO DOAGIN). If COUNT equals 20, the branch is not taken. Instead 20 is subtracted from the COUNT field so that it will be zero for the next cycle.

LOOKUP OPERATIONS

Lookup operations are used when searching through a table or an array to find a special element.

Lookup (LOKUP)

This operation code causes a search to be made for a particular item in a table or array. The table or array is Factor 2. Factor 1 is the search word (data for which you wish to find a match in the table or array named). Factor 1, the search word, may be:

1. An alphameric or numeric constant.
2. A field name.
3. An array element.
4. A table name.

Remember that when a table is named in Factor 1, it refers to the element of the table last selected in a LOKUP operation, not to the whole table.

Resulting indicators are always used in connection with LOKUP. They are used to first indicate the type of search desired and then to reflect the result of the search. A resulting indicator assigned to Equal (columns 58-59) instructs the program to search for an entry in the table or array equal to the search word. The indicator turns on only if such an entry is found. If there are several entries identical to the search word, the first one that is encountered is selected.

An indicator assigned to Low (columns 56-57) instructs the program to locate an entry in the table that is nearest to, yet lower in sequence than, the search word. The first such entry found causes the indicator assigned to Low to turn on.

The indicator assigned to High (columns 54-55) instructs the program to find the entry that is nearest to, yet higher in sequence than, the search word. The first higher entry found causes the indicator assigned to High to turn on. In all cases the resulting indicator turns on only if the search is successful.

A table or array that is not full is called a short table or array. When the program is searching a short table or array, all entries in the table (including blanks and zeros) are compared to the search word.

At least one resulting indicator must be assigned, but no more than two can be used. Resulting indicators can be assigned to Equal and High or Equal and Low. The program searches for an entry that satisfies either condition with Equal given precedence; that is, if no Equal entry can be found, the nearest lower or nearest higher entry is selected. If resulting indicators are assigned both to High and Low, the indicator assigned to Low is ignored. When using the LOKUP operation, remember:

1. The search word and each table or array item must have the same length and the same format (alphameric or numeric), but need not have the same alignment.
2. You may search on High, Low, High and Equal, or Low and Equal only if your table or array is in sequence, or the portion of an array being searched is in sequence.
3. No resulting indicator turns on if the entry searched for is not found.

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation

Form GX21-9093 Printed in U.S.A.

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust	Resulting Indicators	Comments
01	C				EXCPT							
02	C				EXCPT							
03	C				EXCPT							
04	C				EXCPT							
05	C				EXCPT							
06	C				EXCPT							
07	C				EXCPT							
08	C				EXCPT							
09	C				EXCPT							
10	C				EXCPT							
11	C				EXCPT							
12	C				EXCPT							
13	C				EXCPT							
14	C				EXCPT							
15	C				EXCPT							
16	C				EXCPT							
17	C				EXCPT							
18	C				EXCPT							
19	C				EXCPT							
20	C				EXCPT							

Ⓐ

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation

Form GX21-9093 Printed in U.S.A.

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust	Resulting Indicators (High / Low / Equal)	Comments
01	C			DOAGIN	TAG							
02	C				EXCPT							
03	C			1	ADD	COUNT	COUNT	2	0			
04	C			COUNT	COMP	20					Equal: 20	
05	C		N20		GOTO	DOAGIN						
06	C			COUNT	SUB	20	COUNT					

Ⓑ

Figure 139. Using GOTO and TAG (Eliminate Duplicate Coding)

Using the LOKUP Operation

LOKUP with One Table

When searching a single table, Factor 1, Factor 2, and at least one resulting indicator must be specified. Conditioning indicators (specified in columns 7-17) may also be used.

Whenever a table item is found that satisfies the type of search being made (Equal, High, Low), a copy of that table item is placed in a special storage area. Every time a search is successful, the newly found table item is placed in this area, destroying what was there before. If the search is not successful, no table item is placed in the storage area. Instead, the area keeps the contents it had before the unsuccessful search.

Resulting indicators are always set to reflect the result of the search. If the indicator is on, showing a successful search, you know that a copy of the item searched for is in the special storage area.

LOKUP with Two Tables

When two related tables are used in a search, only one is actually searched. When the search condition (High, Low, Equal) is satisfied, the corresponding data items from both tables are made available for use.

Factor 1 must be the search word and Factor 2 must name the table to be searched. The Result Field must name the related table from which data is made available for use. Resulting indicators must also be used. Conditioning indicators (specified in columns 7-17) may be specified if needed.

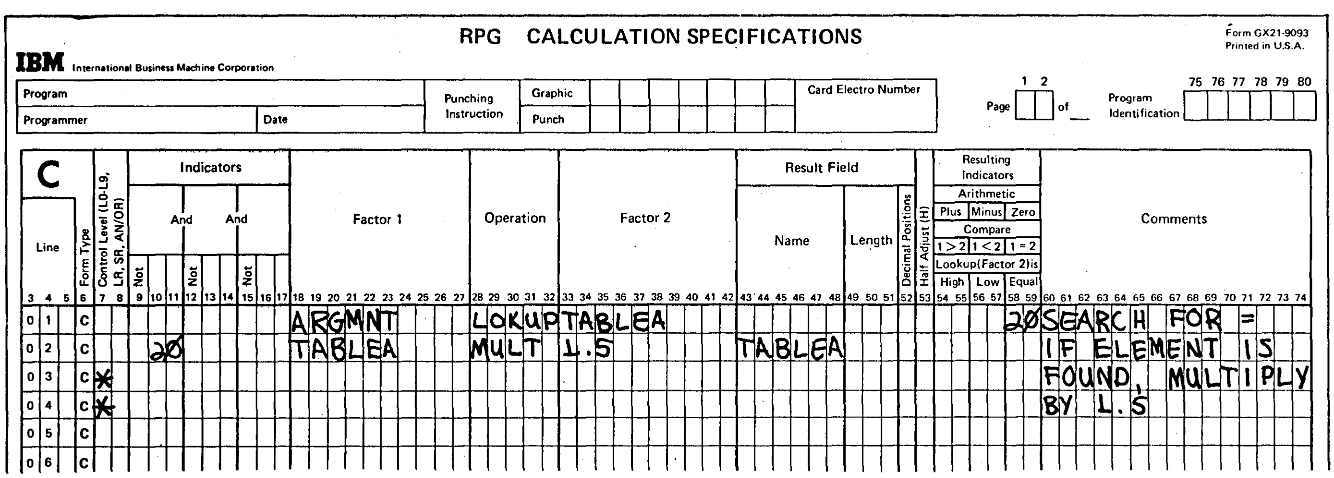
The two tables involved should be the same length. If the table that is searched is longer than its related table, the search stops at the end of the shorter table.

Referencing the Table Item Found in a LOKUP Operation

Whenever a table name is used in an operation other than LOKUP, the table name really refers to the data placed in the special storage area by the last successful search. Thus, by specifying the table name in this fashion, you can use data items from a table in calculation operations.

If the table is used as Factor 1 in a LOKUP operation, the contents of the special storage area are used as the search word. In this way a data item from a table can itself become a search word.

The table may also be used as the Result Field in operations other than the LOKUP operation. In this case the contents of the special storage area are changed by the calculation operation. The corresponding table item in the table itself is also changed. This is a way in which you can modify the contents of the table by calculation operations (Figure 140).



IBM International Business Machine Corporation — RPG CALCULATION SPECIFICATIONS — Form GX21-9093 Printed in U.S.A.

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Resulting Indicators (Equal)	Comments
01	C		ARGMNT	LOKUP	TABLEA		20	SEARCH FOR =
02	C	20	TABLEA	MULT	1.5	TABLEA		IF ELEMENT IS
03	C*							FOUND, MULTIPLY
04	C*							BY 1.5
05	C							
06	C							

Figure 140. Referencing the Table Item Found in a LOKUP Operation

Example of Table Lookup

Figures 141 and 142 show the use of the LOKUP operation. Figure 141, insert A shows the contents of four tables: TABLEA, TABLEB, TABLEC, and TABLED (loaded at compile time). Each table has five entries.

Figure 141, insert B shows the extension specifications for these tables. TABLEA and TABLEB are described separately and are, therefore, entered separately. TABLEC and TABLED are related tables and are entered in alternating format on the table input cards. Figure 142 shows the order in which the table input cards are loaded into the machine at compile time.

LOKUP with an Array

The LOKUP specifications for arrays are the same as for tables except that if Factor 2 is an array, the Result Field cannot be used. In addition if the desired item is found, the indicators reflect only that the desired item is in the array; the programmer does not have ready access to this item.

If you use just the array name in referencing the array, the search begins at the first element in the array. You must use indicators to determine if a match was found.

If you use the array name and an index (which may be a field name or a literal), the search begins at the element identified by the index. If a match is found, the number of the array element containing the match is placed in the field used as an index. If no match is found, the index field is set to 1.

If a literal was used as an index, indicators must be used to determine if a match was found. The content of the element referenced by the literal is not changed.

	First Entry	Second Entry	Third Entry	Fourth Entry	Fifth Entry
TABLEA	01	05	08	32	96
TABLEB	05.13	02.12	47.15	28.70	15.16
TABLEC	WWW	NNN	LLL	GGG	AAA
TABLED	7	8	3	2	5

A

G EXTENSION AND LINE COUNTER SPECIFICATIONS

Punching Instruction	Graphic		Card Electro Number	Page 1
	Punch			

Extension Specifications

me (24–26)	Table or Array Name (27–32)	Number of Entries Per Record (33–35)	Number of Entries Per Table or Array (36–39)	Length of Entry (40–42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46–51)	Length of Entry (52–54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	58 59
	TABLEA	2	5	2			A						
	TABLEB	5	5	4		2							
	TABLEC	1	5	3			D	TABLED	1		Ø		

B

Figure 141. Table Lookup (Tables Used)

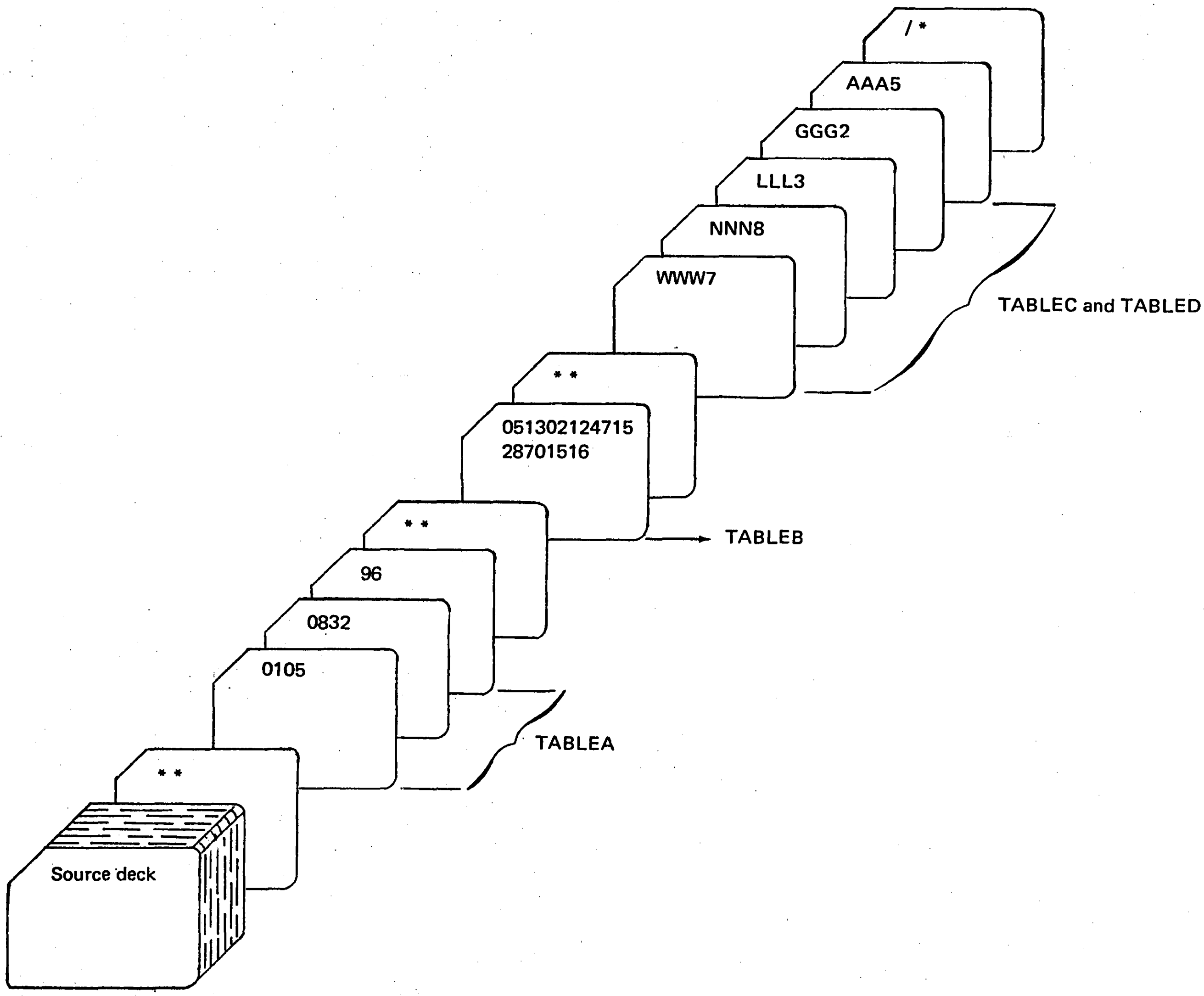


Figure 142. Order in which Tables are Loaded (Compile Time)

Figure 143 shows two LOKUP operations performed with an array. MANNOS, a 2100 element array of employee numbers, is read in at pre-execution time from file ARRFILE with 10 six-position elements per record; the array elements are in ascending order. Line 01 of the Calculation sheet shows a LOKUP of array MANNOS with the object of finding the element nearest to but higher in sequence than the search word '100336'. If this desired element is found in the array, indicator 20 turns on and the GOTO in line 02 is performed. Notice that the result of this LOKUP indicates only whether or not the desired element exists in the array. Line 05 of the Calculation Sheet shows essentially the same LOKUP operation–indicator 20 will turn on when the first element higher in sequence than '100336' is found. Note, however, that in this LOKUP operation, the array MANNOS is indexed by the field INX. This index field was set to 1 in line 04 so the LOKUP will begin at the first element of MANNOS. If the desired element is found, the number of this element (not its contents) is placed in the field INX. In this way, the actual element which satisfied the LOKUP can be used in subsequent calculation operations, as in line 07. If no element was found to satisfy the LOKUP, the field INX would be reset to 1.

RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

IBM International Business Machine Corporation — Form X21-9091, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Extension Specifications

Line (3-5)	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 or Array Name (27-32)	Number of Entries Per Record (33-35)	Number of Entries Per Table or Array (36-39)	Length of Entry (40-42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46-51)	Length of Entry (52-54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58-74)
01	E			ARRFILE		MANNOS	1Ø	21ØØ	6			A						
02	E																	
03	E																	

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093, Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR, AN/OR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators: Arithmetic Plus / Compare 1>2 / Lookup (Factor 2) is High (54-55)	Minus / 1<2 / Low (56-57)	Zero / 1=2 / Equal (58-59)	Comments (60-74)
01	C			'1ØØ336'	LOKUP	MANNOS					2Ø			
02	C		2Ø		GOTO	NEXT								
03	C													
04	C				Z-ADD	1	INX	4	Ø					
05	C			'1ØØ336'	LOKUP	MANNOS,INX					2Ø			
06	C		N2Ø		GOTO	END								
07	C				MOVE	MANNOS,INX	SAVE	6	Ø					
08	C			NEXT	TAG									
09	C													
10	C													
11	C					Calculation								
12	C					Operations								
13	C													
14	C													

Figure 143. LOKUP With an Array

Starting the Search at a Particular Array Item

It is possible, in order to save processing time, to start the LOKUP search at a particular item in the array. This type of search is indicated by additional entries in columns 33-42. Enter the name of the array to be searched in these columns followed by a comma and a numeric literal or the name of a numeric field (with no decimal positions). The numeric literal or numeric field tells the number of the item at which you wish to start the search (Figure 144). This numeric literal or field is known as the index because it points to a certain item in the array. All other columns are used as previously described for the normal lookup operation.

The search starts at the specified item and continues until the desired item is found or until the end of the array is reached. When an index field is used, an unsuccessful search causes the index field to contain the value of one. If, however, an item is found which satisfies the conditions of the LOKUP operation, the number of that array item (counting from the first item) is placed in the index field. A numeric literal used as an index is not changed to reflect the result of the search.

Note: If a literal or field index for an array is zero, or greater than the number of elements in the array, the following will result:

1. For a literal index a severe error occurs, and compilation will cease.
2. For a field index the job will halt, allowing the operator to cancel or restart the program. If the program is restarted, the field index is given a value of one (see *Appendix A, RPG II Halt Procedures*).

SUBROUTINE OPERATIONS

These operation codes are only used for subroutines. See *Subroutines* for information on subroutines. All subroutine operation codes must be written in specification lines following all detail and total calculations. Subroutine lines are always identified by an SR in columns 7-8.

Begin Subroutine (BEGSR)

This operation code serves as the beginning point of the subroutine. Factor 1 must contain the name of the subroutine.

End Subroutine (ENDSR)

This operation code must be the last statement of the subroutine. It serves to define the end of the subroutine. Factor 1 may contain a name. This name then serves as a point to which you can branch by a GOTO statement within the subroutine. The ENDSR operation ends the subroutine and automatically causes a branch back to the next statement after the EXSR operation.

Execute Subroutine (EXSR)

This operation causes all the operations in the subroutine to be performed. EXSR may appear anywhere in the program. Whenever it appears, the subroutine is executed. After all operations in the subroutine are done, the operation in the line following the EXSR operation is performed.

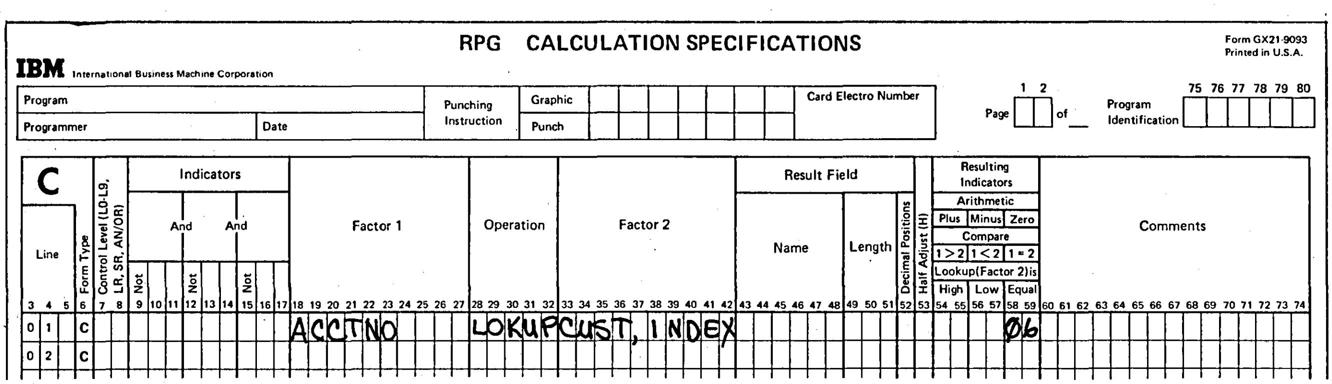


Figure 144. Array Lookup: Starting at a Particular Array Item

This operation may be conditioned by any indicators, meaning the subroutine is executed only when all conditions are satisfied. Factor 2 must contain the name of the subroutine that is to be executed. This same name must appear on a BEGSR instruction.

SUBROUTINES

A subroutine is a routine that is part of another main routine. A routine is something done over and over again. A program can be called a routine because the instructions in a program are done again and again (the program cycle). A subroutine is a group of instructions in that main routine (program) which may be done several times in one program cycle.

Sometimes it is necessary to write a program which at several points does the same operations. Instead of having to write these instructions every time they are needed, it is easier and less time consuming if they can be written just once and then referred to each time they are needed. You can do this by writing a subroutine which then consists of all those operations you have to do at several points in your program.

You might also have to do the same sequence of operations in several different programs. Instead of writing these specifications in each program, you can code the operations once as a subroutine. You then include this subroutine in as many different programs as you wish.

Coding Subroutines

Subroutines are coded and used on the Calculation Sheet. They are entered after all other calculation operations. Every subroutine must have a name, but no two subroutines used in the same program may have the same name.

Enter the name of the subroutine in Factor 1, and on the same line enter the operation code BEGSR (line 10 of Figure 145). The subroutine name can be 1-6 characters long and must begin in column 18 with an alphabetic character. The remaining characters can be any combination of alphabetic or numeric characters (no special characters). Blanks may not appear between characters in the name.

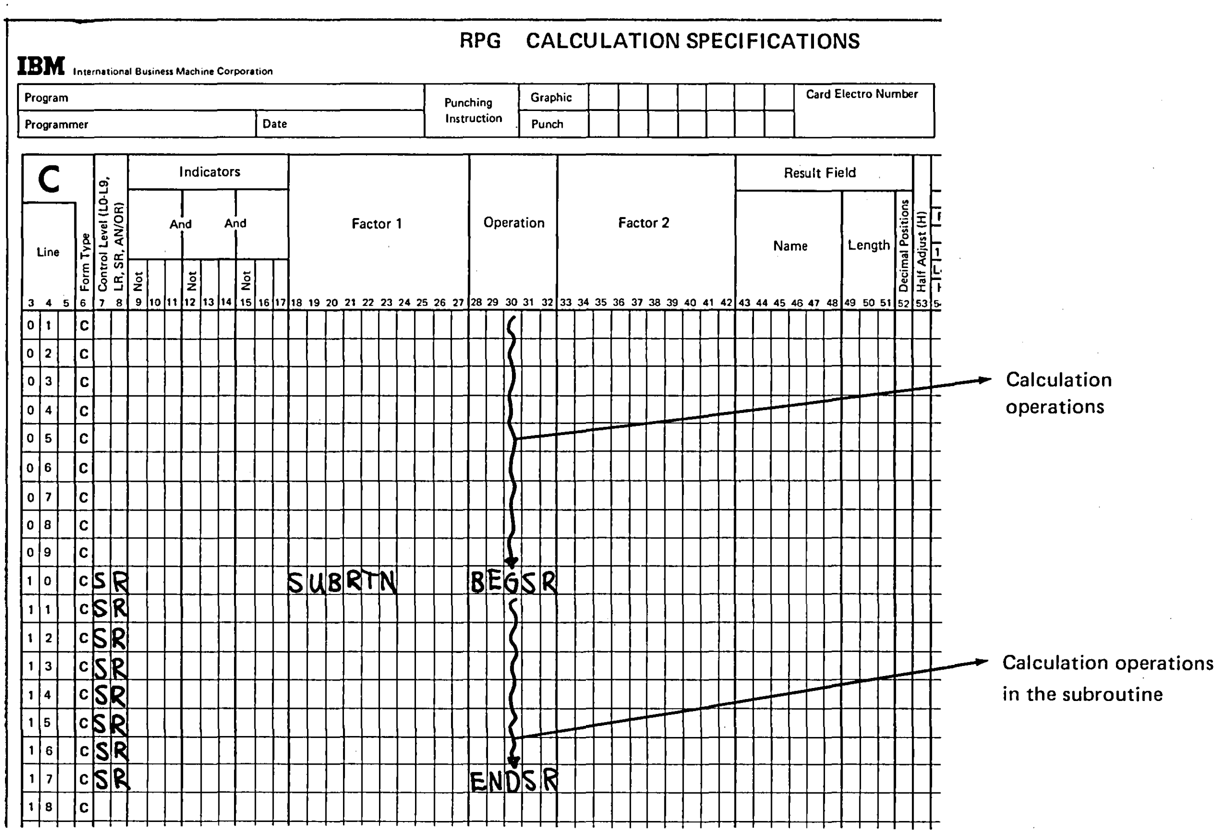


Figure 145. Subroutine Lines (SR)

Each specification line within the subroutine (except AN or OR lines) must have SR in columns 7-8 to identify it as a subroutine line (Figure 145). The last statement of the subroutine is indicated by the operation code ENDSR (line 17 of Figure 145). Factor 1 of the ENDSR statement may contain a name. This name indicates the point to which a GOTO within the subroutine can branch (Figure 146).

The subroutine, even though specified last on the Calculation Sheet, may be performed at any point in the calculation operations. Whenever the subroutine is to be used, enter the operation code EXSR (execute subroutine). The name of the subroutine to be used must also be entered in Factor 2 (lines 04 and 08 of Figure 147). Using the EXSR operation is known as calling a subroutine.

The operation code EXSR causes the operations in the subroutine named in Factor 2 to be performed. After all calculation operations in the subroutine are done, the next operation after the EXSR is performed. For example, when the EXSR operation (line 04 of Figure 147) is encountered, all subroutine operations (lines 11-15) are done. Then the operation in line 05 is performed.

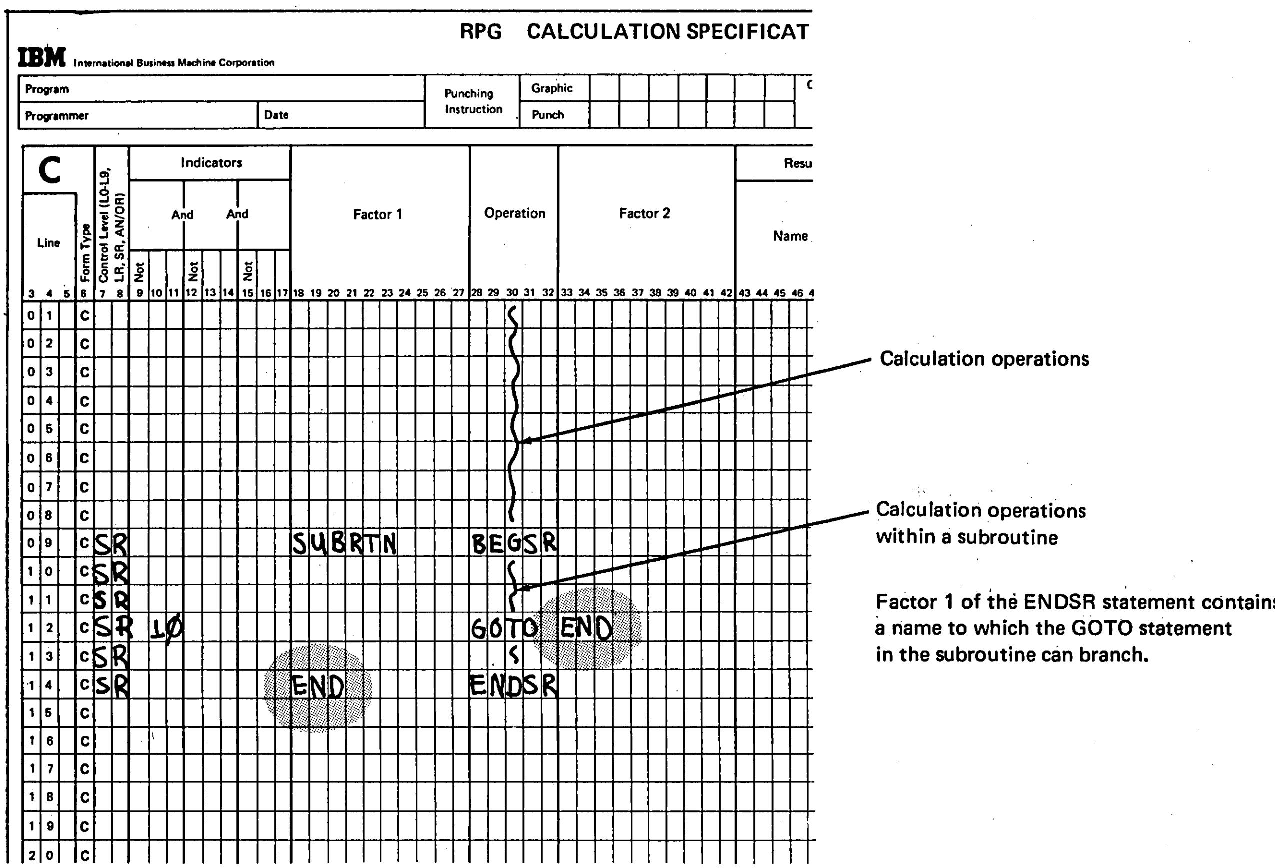


Figure 146. Subroutines (ENDSR)

Indicators may be used with EXSR code to condition when the subroutine should be executed. Any valid indicator may be used in columns 7-17. If no indicators are used, the subroutine is always executed.

All possible RPG II operations may be performed within a subroutine. Operations within the subroutine may be conditioned by any valid indicator in columns 9-17 (Figure 147). Since SR must appear in columns 7-8, control level indicators cannot be used in these columns.

This means that individual operations within the subroutine cannot be conditioned by a control level indicator used in columns 7-8. However, entire subroutines can be conditioned by control level indicators. This can be done by using the control level indicator with the EXSR operation (line 08 of Figure 147).

Fields used in the subroutine may be defined either inside or outside the subroutine. In either case, they can be used by both the main routine and the subroutine.

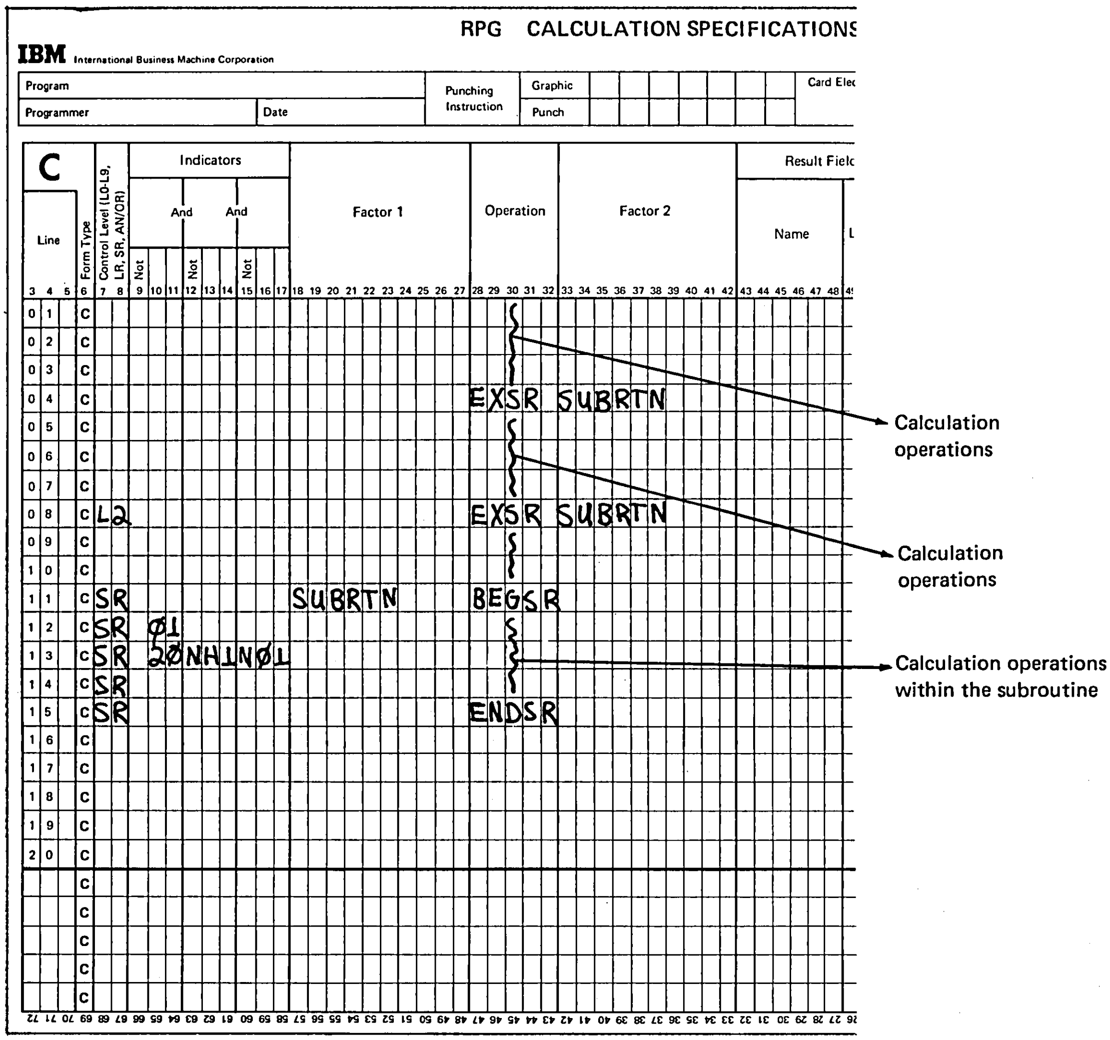


Figure 147. Subroutines (EXSR)

You may use as many subroutines in your main program as you wish. However, you cannot write a subroutine within a subroutine. This means that within one subroutine you cannot have the BEGSR and ENDSR operation codes. One subroutine may call another subroutine, however. In other words, within a subroutine you may have an EXSR operation (Figure 148). A subroutine cannot call itself and cannot call the subroutine which called it.

Subroutines need not be defined in the order in which they are used. However, you must make certain that each one has a different name and a BEGSR and ENDSR operation code.

When you use a GOTO statement in a subroutine, you may only branch to another statement in that same subroutine. Branching (GOTO) to a statement in another subroutine or outside of a subroutine causes an error condition. You cannot use a GOTO from outside the subroutine to a statement within the subroutine either. Figure 149 shows the correct use of GOTO and TAG within a subroutine.

Use of One Subroutine in Many Different Programs

When you wish to do the same operations in many different programs, you may use a subroutine to eliminate duplicate coding in each program. Merely code these operations once and use this subroutine along with your main program deck.

Whenever you code a subroutine to be used in several different programs, remember:

1. When you call the subroutine in your main program (EXSR operation code), you must use the correct name of the subroutine in Factor 2.

2. All fields that will be used both by the subroutine and the main routine must be named the same in each routine. For example, if both the main routine and the subroutine used data from the field called COST on the input card, that field must be named COST in both routines. Keep in mind that the COST field also has the same characteristics (length, decimal positions) in both the main routine and the subroutine.

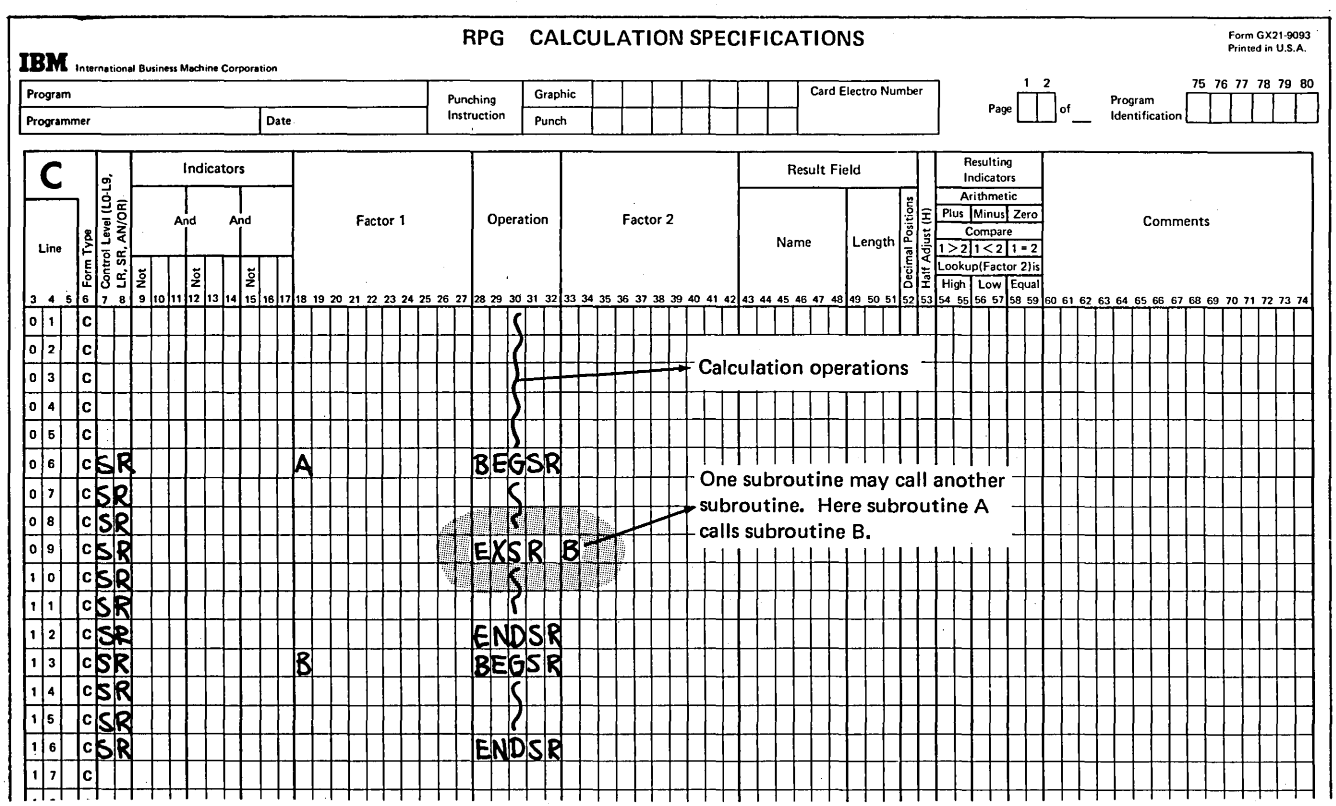


Figure 148. Subroutines: Calling Another Subroutine

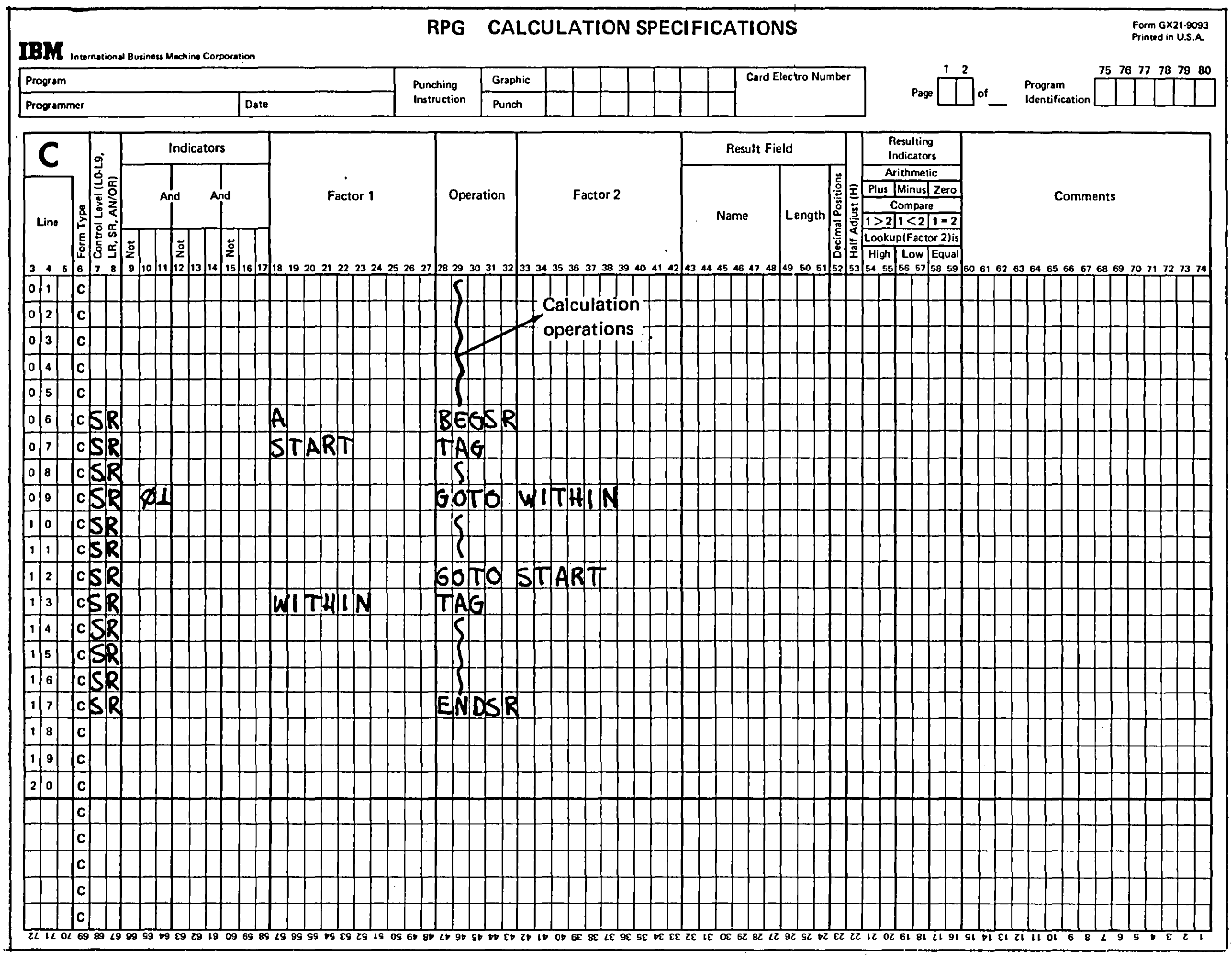


Figure 149. GOTO and TAG Within a Subroutine

PROGRAMMED CONTROL OF INPUT AND OUTPUT

The normal RPG II processing cycle is as follows:

1. A record is read.
2. Calculations are performed.
3. Records are written.

(See *General RPG II Object Program Logic* in Chapter 1 for a brief description of the program cycle.) The normal program cycle can be altered to allow input and output operations during calculations. The following operations provide this capability:

- Exception (EXCPT)
- Force (FORCE)
- Display (DSPLY)
- Read (READ)
- Chain (CHAIN)
- Set lower limit (SETLL)

Exception (EXCPT)

This operation allows records to be written at the time calculations are being done. Use this primarily when you wish to have a variable number of similar or identical records (either detail or total) written in one program cycle. (Remember that normally, only the exact number of records specified in the output specifications are written or punched on a file in one program cycle.) For example, you might use EXCPT to produce a variable number of identical mailing labels, to write out contents of a table, or to produce a number of records having the same information punched in them.

When the EXCPT operation is used, EXCPT is entered in columns 28-32, and columns 7-17 may have entries. All other columns must be blank. The line or lines which are to be written out during calculation time are indicated by an *E* in column 15 of the Output Sheet. Exception lines may not be used in a combined file.

Figure 150 shows the use of the EXCPT operation to produce a variable number of records having the same information punched in them. Records in the input file have two fields, NAME and COUNT. The NAME field is to be entered into a certain number of records. That number is indicated in the COUNT field.

Every time the operation code EXCPT is performed, the exception record indicated by the *E* in column 15 of the Output Sheet is punched. The field CONSEC is used to keep track of the number of records punched. Each time an exception record is written, 1 is added to CONSEC. CONSEC is then compared with COUNT, the field that tells how many records should be punched. If they are not equal (indicator 20 is not on), a branch is taken back to DOAGIN. Another record is punched out. One is added to CONSEC and CONSEC is compared to COUNT. If these fields are now equal, another input record is read. If not, the same operations are done again. Whenever CONSEC equals COUNT, enough records have been punched. CONSEC is then subtracted from itself, making it zero. This last operation is necessary so that an accurate count can be kept for the next record.

Force (FORCE)

FORCE statements enable you to select the file from which the next record is to be taken for processing. They apply to primary or secondary; input, update, or combined files.

Factor 2 in a FORCE statement identifies the file from which the next record is to be selected. If the statement is executed, the record is selected at the start of the next program cycle. If more than one FORCE statement is executed during the same program cycle, all but the last is ignored. FORCE should not be specified at total time.

FORCE statements override the multifile processing method by which the program normally selects records. However, the first record to be processed is always selected by the normal method. The remaining records can be selected by FORCE statements. When end of file is encountered on a forced file, a record will not be retrieved from the file; normal record selection will determine which record is to be processed.

RPG INPUT SPECIFICATIONS

Line	Form Type	Filename	Sequence	Record Identifying Indicator	From	To	Decimal Positions	Field Name
01	I	INPUT	AB	01				
02	I				1	20		NAME
03	I				21	23	0	COUNT
04	I							

RPG CALCULATION SPECIFICATIONS

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Resulting Indicators (Equal)	Comments
01	C		DOAGIN	TAG						
02	C			EXCPT						
03	C		1	ADD	CONSEC	CONSEC	3	0		
04	C		COUNT	COMP	CONSEC				20	EQUAL?
05	C	N20		GOTO	DOAGIN					NO-DOAGIN
06	C	20	CONSEC	SUB	CONSEC	CONSEC				YES-SET INDEX=0
07	C									

RPG OUTPUT SPECIFICATIONS

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators	Field Name	End Position in Output Record
01	O	OUTPUT	E	01		
02	O				NAME	20
03	O					

Figure 150. EXCPT Operation (Producing a Variable Number of Identical Records)

Example

Figure 151 shows part of a job which uses FORCE operation codes and look ahead fields to simulate normal record selection. Normal record selection is not used because records in the two secondary files have two match fields, CUST and ITEM, and those in the primary file have only one, CUST. Normal record selection requires all three to have the same number of match fields.

Indicators 20-23 and 26-28 are used to determine which file the next record is to be read from. The conditions under which the files are chosen follow. Record 1 means the record from the primary file; record 2 the first secondary file; and record 3, the second secondary file.

Condition	*Indicators Set On*	*File Selected*
None of the records match. Record 1 has the lowest CUST field value.	20 and 22	Primary (FIRST)
Record 1 matches record 2. Record 3 has a higher CUST field value.	21 and 22	Primary (FIRST)
Record 1 matches record 3. Record 2 has a higher CUST field value.	20 and 23	Primary (FIRST)
Records 1, 2, and 3 match (CUST field values).	21 and 23	Primary (FIRST)
Record 2 has lower CUST field value than record 1. Record 2 has lower CUST and ITEM fields (together) value than record 3.	26	First secondary (SECOND)
Record 2 matches record 3 (both CUST and ITEM fields). Record 1 has greater CUST field value.	27	First secondary (SECOND)
Record 3 has lower CUST field value than record 1. Record 3 has lower CUST and ITEM field (together) value than record 2.	28	Second secondary (THIRD)

File Description Specification

Line	Form Type	Filename	File Type	File Designation	File Format	Block Length	Record Length	Length of Key Field	Record Address Type	Type of File Organization	Key Field Starting Location	Device	Number of Extents
02	F	FIRST	I	P	F	120	120	6	A	I	10	DISK	01
03	F	SECOND	I	SE	F	96	96					MFCU1	
04	F	THIRD	I	SE	F	96	96					MFCU2	
05	F												
06	F												
07	F												
08	F												
09	F												

Ⓐ

Figure 151 (Part 1 of 2). FORCE Operation Code

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page of | Program Identification

Line	Form Type	Filename	Sequence	Record Identifying Indicator or **	Field Location From	Field Location To	Field Name	
01	I	FIRST	AA	Ø1				
02	I				1	5	CUST	
03	I		AB	**				
04	I				1	5	CUST1	Look ahead field--match field from next record in primary file.
05	I	SECOND	AC	Ø2				
06	I				1	5	CUST	
07	I				6	8	ITEM	
08	I		AD	**				
09	I				1	5	CUST2	Look ahead fields-match fields from next record in first secondary file.
10	I				1	8	MATCH2	
11	I	THIRD	AE	Ø3				
12	I				1	5	CUST	
13	I				6	8	ITEM	
14	I		AF	**				
15	I				1	5	CUST3	Look ahead fields-match fields from next record in second secondary file.
16	I				1	8	MATCH3	
17	I							

Ⓑ

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction | Graphic | Punch | Card Electro Number | Page of | Program Identification

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Resulting Indicators High (1>2)	Low (1<2)	Equal (1=2)
01	C			SETOF		22	23	24
02	C			SETOF		25	26	27
03	C			SETOF		28	29	
04	C		CUST1	COMP	CUST2		20	21
05	C	N20N21		GOTO	TEST2			
06	C		CUST1	COMP	CUST3		22	23
07	C	N22N23		GOTO	TEST2			
08	C			FORCE	FIRST			
09	C	21 23	MATCH2	COMP	MATCH3	25	25	24
10	C			GOTO	ENDTST			
11	C		TEST2	TAG				
12	C		MATCH2	COMP	MATCH3	28	26	27
13	C	N28		FORCE	SECOND			
14	C	28		FORCE	THIRD			
15	C	N27	CUST2	COMP	CUST3			29
16	C		ENDTST	TAG				
17	C							

Ⓒ

Figure 151 (Part 2 of 2). FORCE Operation Code

In addition, indicators 24, 25, and 29 are set to condition calculations which process the record selected.

Condition	*Indicator Set On*
Records 1, 2, and 3 match (CUST fields). Records 2 and 3 match (CUST and ITEM fields).	24
Records 1, 2, and 3 match (CUST fields). ITEM fields in records 2 and 3 do not match.	25
CUST field values in records 2 and 3 match; ITEM fields do not. Record 1 has higher CUST field value.	29

All the calculations shown in Figure 151, insert C are needed to determine which record is to be processed next. The operations which are performed upon the data from the input records are not shown. They do, however, precede the calculations shown in Figure 151, insert C and are conditioned by the indicators set during the previous cycle by the calculations shown.

Display (DSPLY)

The display operation allows either or both of the following:

Models 10 and 12:

1. A field, table element, array element, or literal up to 125 characters long is printed on the printer-keyboard during program execution without a program halt.

2. A field, table element, literal, or array element up to 125 characters long is printed on the printer-keyboard and the program halts, allowing that field to be changed.

Model 15:

1. A field, table element, array element, or literal up to 35 characters long is displayed on the CRT during program execution without a program halt.

2. A field, table element, literal, or array element up to 35 characters long is displayed on the CRT, and the program halts, allowing that field to be changed.

See Figure 152 for coding possibilities and results. Also see Figure 154 under CHAIN operation in this chapter for an example using the DSPLY operation. A literal may not be changed with DSPLY.

There are several points to remember if you wish to enter data during program execution:

1. Numeric data need not be entered with leading zeros; numeric data is right-justified after all characters are keyed.

2. To key a negative field, the field is keyed followed by a minus sign. The length of the field does not need to accommodate the minus sign.

3. Alphameric fields are left-justified after all characters are keyed.

4. Alphameric fields are blanked out and numeric fields are zeroed out.

5. If no characters are entered or the space bar is not pressed, the result field will not be changed.

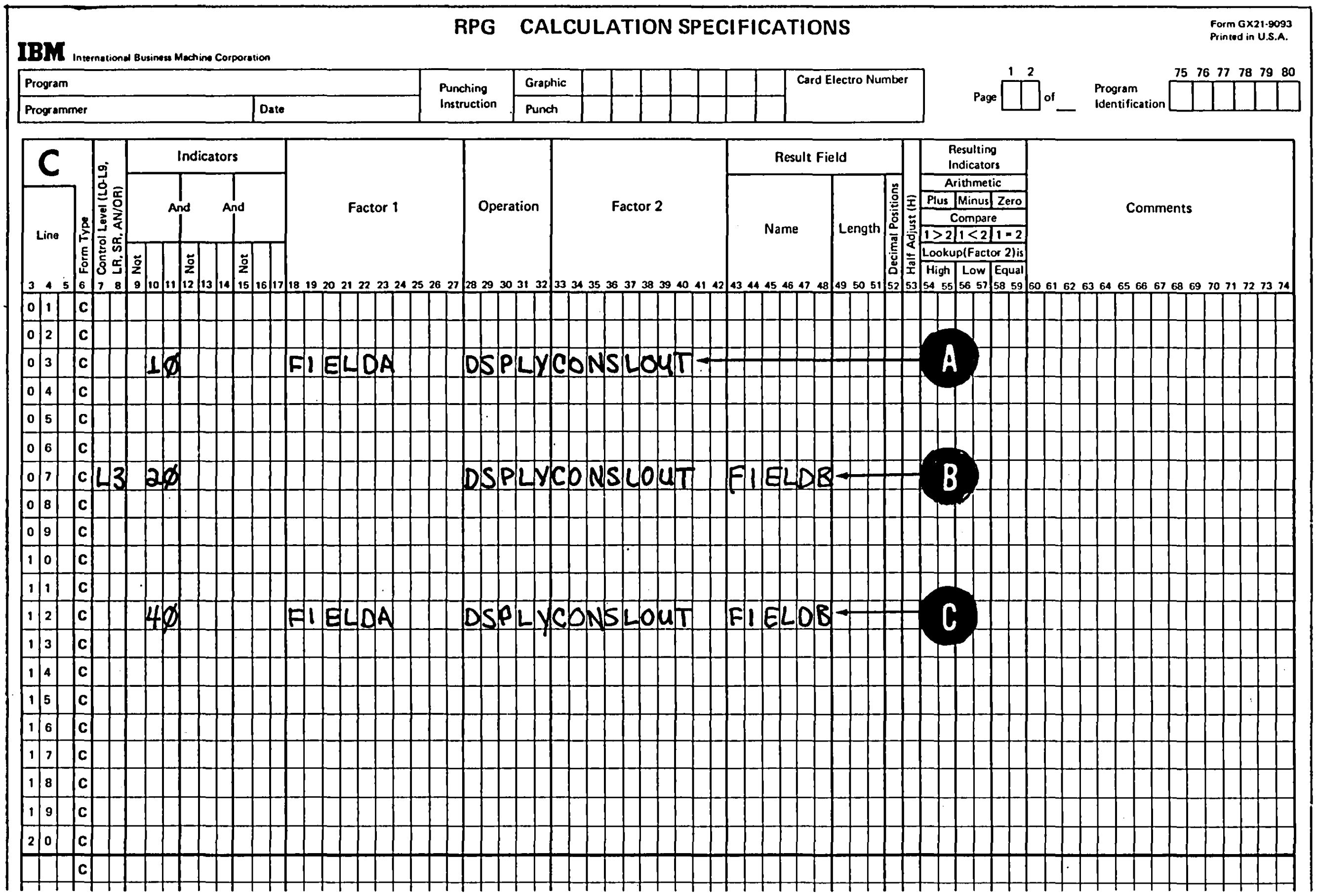
6. (Models 10 and 12 only) The data entered must be followed by pressing END if the data is correct, or CANCEL if you want to re-enter data.

   (Model 15 only) In order to enter data, the operator presses the PF12 key on the 3277 keyboard. This positions the cursor to the first byte of the response line. The operator then enters the desired data via the 3277 keyboard and presses ENTER to continue.

   If the data as displayed is correct, and no correcting entry is required, it is only necessary to press the PF12 key followed by ENTER to continue. If you enter data on the response line and then determine that no entry was required, your recovery is as follows:

   a. If the result field is numeric, enter any non-numeric character and press ENTER.
   b. If the result field is alphameric, fill the response line with any character or combination of characters and press ENTER.

   This causes a halt, which should be responded to with option 0. The original display will reappear and you may proceed by pressing the PF12 key followed by ENTER.



Results:

Model 10 Output

Position 1

A

1. FIELDA is printed as shown. → DSPLY
   Contents of FIELDA (Factor 1)
2. FIELDA does not change.
3. Program does not halt.

B

1. FIELDB is printed as shown. → DSPLY
   Contents of FIELDB (Result Field)
2. Program halts.
3. FIELDB is blanked out if data is entered or the space bar is pressed.
4. Data can be entered in FIELDB.

C

1. FIELDA and FIELDB are printed as shown. → DSPLY
   Contents of FIELDA (Factor 1)
   Contents of FIELDB (Result Field)
2. FIELDA does not change.
3. Program halts.
4. FIELDB is blanked out if data is entered or the space bar is pressed.
5. Data can be entered in FIELDB.

Notes:

- Factor 1 cannot be the name of a whole array.
- Fields A and B can be up to 125 characters long on the Models 10 and 12 or up to 35 characters long on the Model 15.

Figure 152. Methods of Coding the Display Operation

Read (READ)

The READ operation is used to call for immediate input from a demand file during the calculations in the program cycle. This operation differs from the FORCE operation because FORCE specifies input on the next program cycle, not the present one. The READ operation is similar to the CHAIN operation, except that the READ file is processed sequentially and the CHAIN file is processed randomly.

The operation code READ must appear in columns 28-32. Factor 2 contains the name of the file from which a record will be read immediately. An indicator should be used in columns 58-59. An indicator specified in these columns will turn on after each READ operation if an end of file condition is reached. If columns 58-59 are blank, a halt will occur on an end-of-file condition and on subsequent READ operations after the end-of-file condition is reached. The indicator used in columns 58-59 is not reset by subsequent READ operations. Indicators may be specified in columns 7-17.

Note: When a program is doing multiple reads from one or several demand files during the same RPG II cycle, the record identifying indicators assigned to the file(s) remain on throughout the cycle if the previous READ operations were executed successfully.

The following files can appear as Factor 2 in a READ operation (all must be designated demand files with a D in column 16 of the File Description Sheet):

- Sequential or direct disk files processed consecutively and specified as input or update files.
- Indexed disk files processed sequentially by key and specified as input or update files.
- Indexed disk files processed sequentially within limits and specified as input or update files.
- Console files specified as input (Models 10 and 12 only).
- Tape files specified as input.
- MFCU files specified as input or combined files.
- 1442 files specified as input or combined files.
- MFCM files specified as input or combined files (Model 15 only).
- 2501 files specified as input files (Model 15 only).
- CRT77 files specified as input or update files (Model 15 only).
- DISKET files specified as input files.

When using the READ operation for demand files remember these points:

1. Demand files can only be processed by the READ operation.
2. Control levels, matching fields, and look-ahead fields are not allowed with demand files.
3. Numeric sequence testing on the Input Sheet is not allowed for demand files.
4. The MR indicator may not be entered in columns 63-64 (Field Record Relation) on the Input Sheet.
5. When a demand file is conditioned by a U1-U8 indicator which is not on, no records will be read from that file and the end-of-file indicator (in columns 58-59 of the Calculation Sheet) will not turn on.
6. If the end of a file on a diskette is not reached when the program ends, the diskette will be positioned to the second record after the last record read. For more information, see the *IBM System/3 3741 Reference Manual,* GC21-5113.

Example: Assigning Employee Numbers to New Employees

Figure 153 shows the coding necessary to process a demand file with the READ operation code. The combined input and output file NEWNAME, consisting of a deck of cards with a name field in columns 8-96, is read from the primary MFCU hopper. The disk file NUMBRFLE, specified as an update demand file, consists of records containing a seven-digit number and a flag mark. For each record read from NEWNAME, a record is also read from NUMBRFLE during the calculation phase by means of the READ operation code. If the record from the demand file contains a flag (field indicator 88 is off), another record is immediately read. This loop is repeated until a record without a flag has been read from NUMBRFLE; a flag of 'X' is then moved into the FLAG field. When end of file has been reached on the demand file and each time READ is encountered thereafter, resulting indicator H1 is turned on.

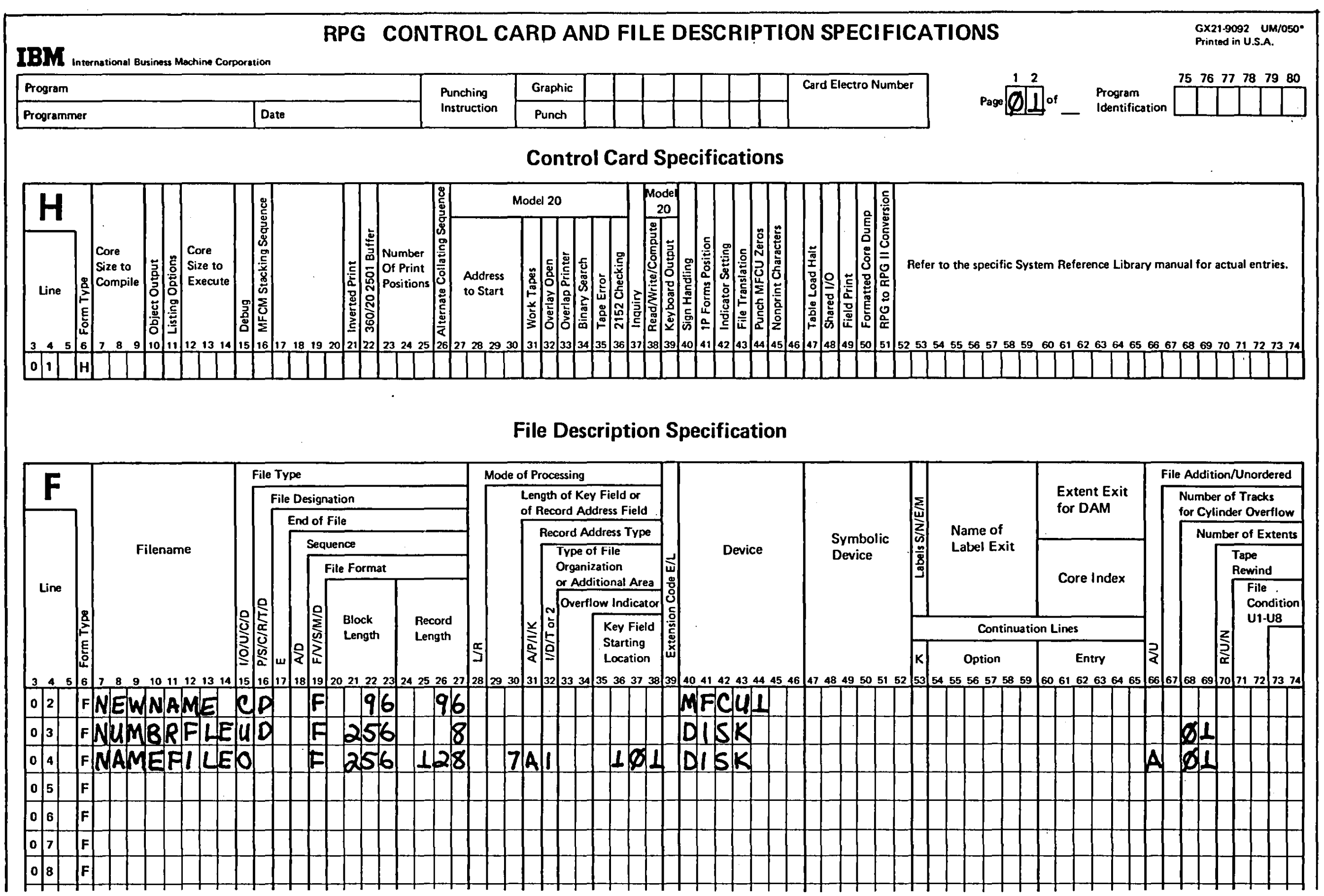


Figure 153 (Part 1 of 2). READ Operation Code

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 U/M 050* Printed in U.S.A. — Page Ø2

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Field Location From	Field Location To	Field Name	Field Indicators Zero or Blank
01	I	NEWNAME	NS	Ø1				
02	I				8	96	NAME	
03	I	NUMBRFLE	NS	Ø2				
04	I				1	7	NUMBER	
05	I				8	8	FLAG	88

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093 Printed in U.S.A. — Page Ø3

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Resulting Indicators
01	C		LOOP	TAG			
02	C			READ	NUMBRFLE		H1
03	C	N88NH1		GOTO	LOOP		
04	C			MOVE	'X'	FLAG	

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050* Printed in U.S.A. — Page Ø4

Line	Form Type	Filename	Type (H/D/T/E)	AND/ADD	Output Indicators	Field Name	End Position in Output Record
01	O	NEWNAME	D		Ø1		
02	O					NUMBER	7
03	O					*PRINT	
04	O	NAMEFILE	D	ADD	Ø1		
05	O					NAME	96
06	O					NUMBER	1Ø7
07	O	NUMBRFLE	D		Ø1		
08	O					FLAG	8

Figure 153 (Part 2 of 2). READ Operation Code

At detail output time, the flagged number from the record in NUMBRFLE is punched and printed on the card from NEWNAME. The record from NUMBRFLE, which now contains a flag, is returned to its original location on the disk. The disk file, NAMEFILE, is then written containing the name from the NEWNAME card file and the number from the demand file, NUMBRFLE.

Chain (CHAIN)

The chain operation causes a record to be read from a disk file during calculations. This operation allows one record to be read in when the operation code CHAIN appears in columns 28-32 of the Calculation Sheet.

The chain operation is used for two purposes:

1. Random processing of an indexed, sequential, or direct file.

2. Loading a direct file.

Note: When chaining to one or more files during the same RPG II cycle, record identifying indicators assigned to the chained file(s) remain on throughout the cycle if any of the previous CHAIN operations were executed successfully. When chaining to the same file more than once during an RPG II cycle, only the last record processed will be updated during output time unless an exception output is associated with each chain operation.

Note: If the same physical file is processed as UPDATE and INPUT, successive chains to the same record may yield the old data after update, because the contents of the input buffers are used instead of rereading the disk.

Indicators in columns 7-17 may be used, but Result Field, Field Length, Decimal Position, and Half-Adjust (columns 43-53) must be blank. File conditioning indicators (U1-U8) can be used to condition a chained file.

Columns 54-55 should contain an entry. If the record is not found, the indicator specified in these columns will turn on. No update is permitted to a chained update file when the specified record is not found. However, addition to a file is allowed when the specified record is not found. Columns 56-59 must always be blank for chain operations.

If an indicator is not specified in columns 54-55, and the record is not found, the program will halt. The options given are to end the job or to bypass the remainder of the current cycle and begin a new cycle. If LR processing has already been initiated, the bypass-and-begin-new-cycle option is not allowed. If the controlled cancel option is taken, files are closed, but the rest of the LR processing does not occur.

When the program is chaining to a file with packed record keys, the entry in Factor 1 of the CHAIN operation must have a packed length which is the same as the length of the key field in the chained file. Packed key fields can be a maximum of 8 bytes. The following chart shows the packed equivalents for unpacked fields from one to 15 bytes in length:

Unpacked Length	*Packed Length*
15, 14	8
13, 12	7
11, 10	6
9, 8	5
7, 6	4
5, 4	3
3, 2	2
1	1

Random Processing

In order to read a record from a sequential or direct file, the record must be identified by relative record number. To read a record from an indexed file, a record key is used for identification. The relative record number or key can be contained in a field specified for that purpose.

The chain operation requires the operation code CHAIN in columns 28-32 of the Calculation Sheet. Factor 1 entries must be a relative record number or key. Relative record numbers must be numeric. Factor 2 must contain the name of the file from which the record will be read. This file is called the file that is chained to, or the chained file (see *Examples, Example 1*).

Direct File Load

To create (load) a direct file, define it as a chained output file on the File Description Sheet. In the calculation specifications, Factor 1 must contain a relative record number, columns 28-32 must contain the operation code CHAIN, and Factor 2 must contain the name of the direct disk file to be loaded.

Relative record numbers define the record position for each record in the direct disk file. The relative number can be all or part of a field in input records or can be generated by the RPG II program. Relative record numbers are used for record identification of the disk records after the disk file is loaded.

When a direct file is loaded as a chained output file, the system clears the disk space required for the direct file with blanks before it is loaded. The relative record number is used to chain to the corresponding relative record position in the disk file. The information is then written on disk, replacing the blanks with data. If a record is not loaded, the space reserved for that record in the disk file remains blank (until the proper record is loaded later).

Once the direct file is loaded, records are inserted or changed in the file by defining the direct file as an update file processed consecutively or by the chain operation (see *Note*).

You may have to allow for *synonyms* when you load a direct file. Synonyms are two or more records with the same relative record number. If you will have synonyms, you can load the file in one of two ways, using multiple passes:

1. Define the disk file as a direct file and clear it to blanks in your first job (by defining it as a chained output file). Once the file has been cleared, one or more subsequent jobs can be run using the update function to read record locations and check for synonyms while loading the file.

2. Load the direct file with records without synonyms, then run another job using the update function to identify synonyms and load them into the file.

Note: The insertion of records in direct disk files is very different from record addition to sequential or indexed files. For sequential disk files, the new record is added in at the first available position at the end of the file. The same process occurs for an indexed file, except that the record key and disk address are added to the file index. Any new records inserted in a direct disk file already have a space reserved for them. Hence, the record is inserted in its proper place, not merely added to the physical end of the file.

Examples

Example 1: Figure 154 shows the coding necessary to chain to and update an indexed file, MASTINV. The CARDIN file consists of cards sorted by item number, each card representing some quantity ordered. Item number is used as a control field. When all the quantities for one item number are added, a control break will occur. At this point in calculations, the master record for that item number must be found and updated. ITEMNO is a field containing the item number of the cards presently being worked on. The chain operation uses ITEMNO to find the master record for that item number. If it is not found, a display operation prints out the item number of the cards. Note that indicator 20 turns *on* when the records are *not* found.

If the master record is found (20 not on) the total quantity for the item number is subtracted from the quantity on hand. After the total calculations, the QOH field in the master record is updated.

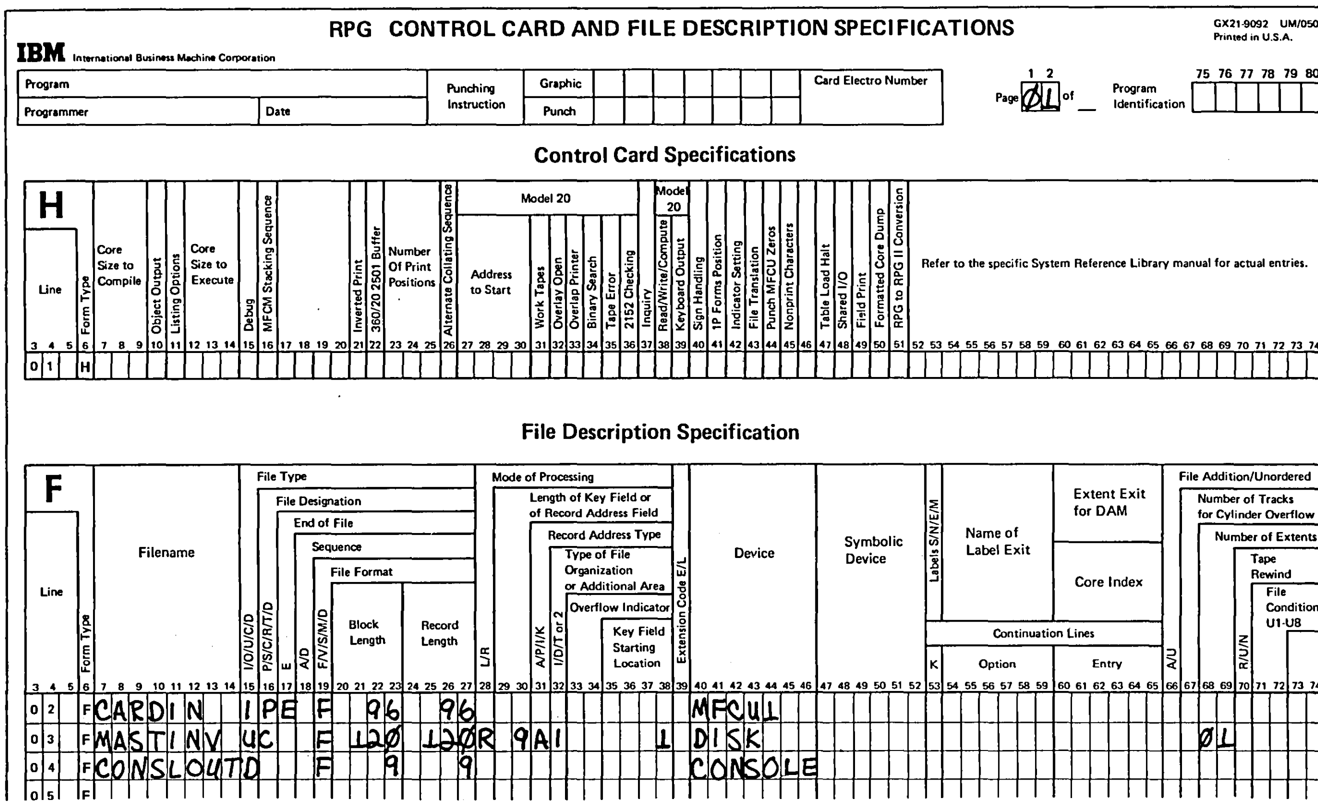


Figure 154 (Part 1 of 2). Chain Operation

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page ØŁ of | Program Identification

Line	Form Type	Filename	Sequence	Number (1-N)	Option (O)	Record Identifying Indicator or **	Record Identification Codes 1: Position	Not (N)	C/Z/D	Character	Field Location From	To	Decimal Positions	Field Name	Control Level (L1-L9)
01	I	CARDIN	AA			Ø1	96		C	X					
02	I										21	29	Ø	ITEMNO	L1
03	I										3Ø	36	Ø	QTY	
04	I	MASTINV	BB			Ø2	12Ø		C	1					
05	I										1	9	Ø	ITMNO	
06	I										1Ø	18	Ø	QOH	

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093 Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page Ø3 of | Program Identification

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Resulting Indicators	Comments
01	C		Ø1	QTY	ADD	TOTQTY	TOTQTY	7	Ø		DETAIL CALCS
02	C	L1		ITEMNO	CHAIN	MASTINV				2Ø	FIND MASTER REC
03	C	L1	2Ø	ITEMNO	DSPLY	CONSLOUT					FOUND?NO-DSPLY
04	C	L1	N2Ø	QOH	SUB	TOTQTY	QOH				YES-TOTAL CALCS
05	C										
06	C										
07	C										
08	C										
09	C										
10	C										

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic / Punch | Card Electro Number | Page Ø4 of | Program Identification

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators	Field Name	End Position in Output Record
01	O	MASTINV	T	L1 Ø2N2Ø		
02	O				QOH	18

Figure 154 (Part 2 of 2). Chain Operation

Example 2: Figure 155 shows the loading of a direct disk file. NAMEFILE, described as a chained output file on the File Description Sheet, is to be loaded with records read from CARDS, a card file read from the primary MFCU hopper.

Prior to loading, NAMEFILE is cleared to blanks. As each record is read from CARDS, the man number (MANNUM) is used as the relative record number to chain to NAMEFILE during calculations. The entire input record, RECORD, is written out on NAMEFILE in the relative record location corresponding to MANNUM. When end of file (*E* in column 17 of the File Description Sheet) is reached on CARDS, any relative record locations on NAMEFILE which have not been loaded with data from CARDS will contain blanks.

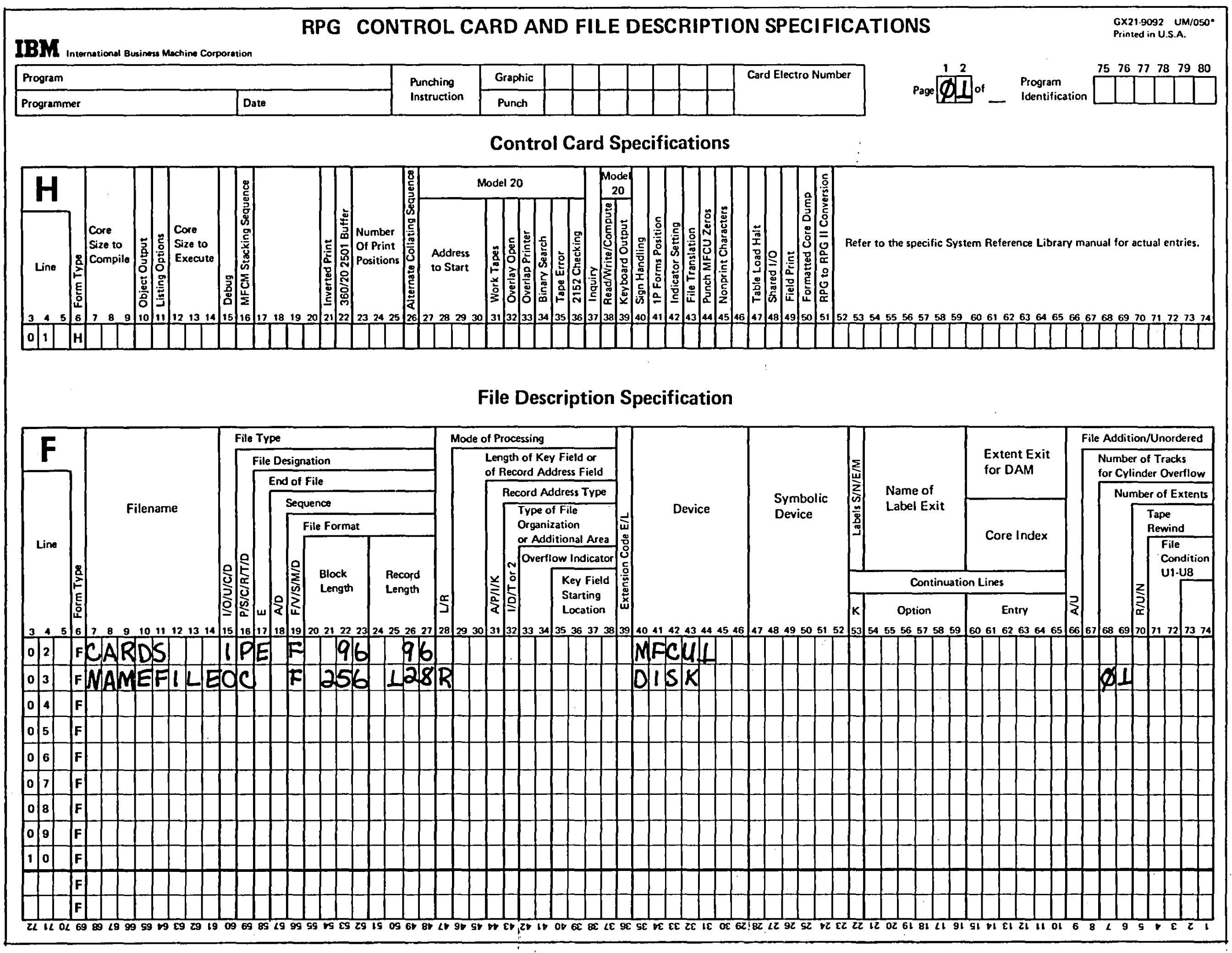


Figure 155 (Part 1 of 2). Direct File Load (Random Load)

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 U/M 050* Printed in U.S.A.

Page 02

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Field Location From	To	Decimal Positions	Field Name
01	I	CARDS	NS	Ø1				
02	I				1	5	Ø	MANNUM
03	I				1	96		RECORD
04	I							
05	I							

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093 Printed in U.S.A.

Page 03

Line	Form Type	Factor 1	Operation	Factor 2
01	C	MANNUM	CHAIN	NAMEFILE
02	C			

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050* Printed in U.S.A.

Page 04

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators	Field Name	End Position in Output Record
01	O	NAMEFILE	D	Ø1		
02	O				RECORD	96
03	O					

Figure 155 (Part 2 of 2). Direct File Load (Random Load)

Set Lower Limits Operation (SETLL)

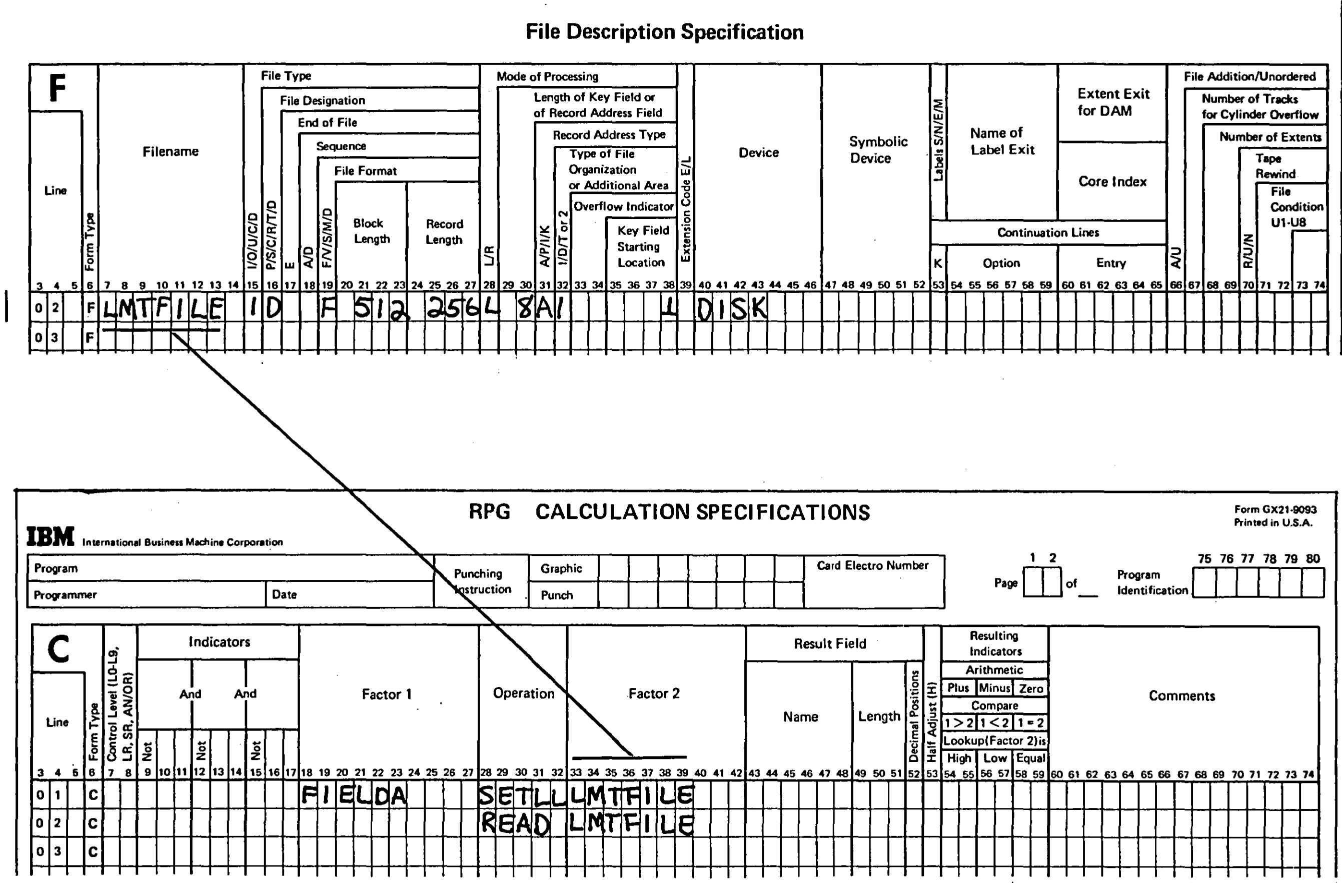
This operation allows the lower limits, for Index Demand files being processed within limits, to be set during calculations.

Factor 1 must contain a field name or literal representing the value of the lower limit being set. The length of the field or literal must be equal to the length of the key for the file named in Factor 2.

Factor 2 must contain the name of the file for which the lower limit is to be set. If a read is performed to the file prior to the first SETLL instruction the record with the lowest key in the file is read. Figure 156 shows a SETLL coding example.

When end-of-file is reached on a file being processed by SETLL, another SETLL can be issued and processing of the file may continue. The SETLL operation can be used whenever a new lower limit is desired.

Note: When a lower limit is specified by SETLL, the end-of-file indicator used on the read operation (READ) to the file being processed is not set off.



Note: FIELDA would be defined as an 8 position alphameric field.

Figure 156. SETLL Operation

DEBUG OPERATION

The debug operation is an RPG II function that you may use to help you find errors in a program which is not working properly. This code causes one or more records to be written containing information helpful for finding programming errors.

Debug (DEBUG)

The DEBUG operation code may be placed at any point or at several points in the calculation operations. Whenever it is encountered, one or more records are written depending upon the specifications entered. One record contains a list of all indicators which are on at the time the DEBUG code was encountered. The other shows the contents of any one field.

Note: The decimal point is not printed.

Specifications

Factor 1 is optional. It may contain a literal or field name which identifies the particular debug operation. The literal or the value of the field named here is written on record 1. Factor 2 must contain the name of the output file on which the records are written. The same output filename must appear in Factor 2 for all DEBUG statements in a program. The file used for a DEBUG operation must be an output file for the statements to work correctly (printer, punch, disk with no add). The result field may be a field, table element, array element, or whole array whose contents you want to write on record 2. Any valid indicator may be used in columns 7-17. Columns 49-59 must be blank.

Because of additional processing considerations, care must be exercised when writing debug records to a direct or indexed file.

The operation code produces results only if the proper entry (1 in column 15) has been made in the control card specifications. If the control card entry has not been made, the operation code DEBUG is treated as a comment if its output filename is not unique for DEBUG. See *Column 15* in Chapter 3 for more information.

Records Written for DEBUG

Record 1 is required. It is written in the following format:

Record Positions	*Information*
2-7	DEBUG–
8	Blank.
9-16	Constant entered in Factor 1 or the statement number of the DEBUG operation code in the program.
17	Blank.
18-31	The words INDICATORS ON–
32–any position (depending on number of indicators on)	The names of all indicators which are on, each separated by a blank. The word NONE if no indicators are on. More than one record may be needed.

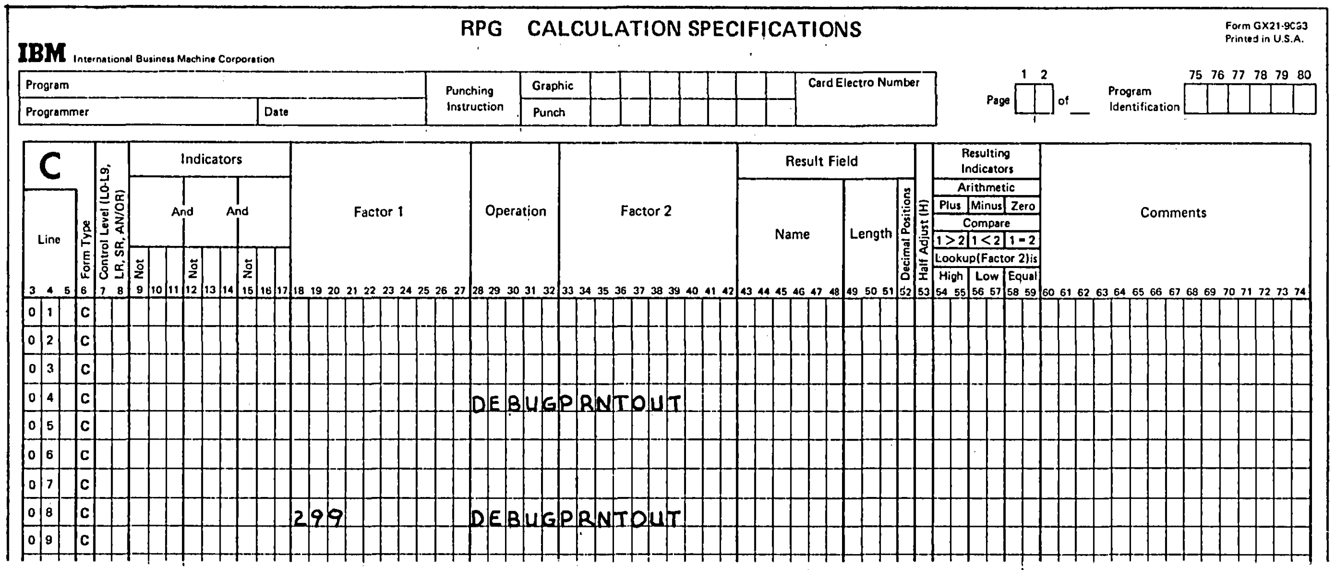
See Figure 157 for an example of coding and output for record 1.

Record 2 is optional and is written only when there is a result field. The record is written in the following format:

Record Positions	*Information*
2-12	The words FIELD VALUE or TABLE VALUE or ARRAY VALUE.
13-14	Blank.
15–any position (depending on length of field)	The contents of the result field or table or array (up to 256 characters per element). More than one record may be needed.

The field is written in record 2 according to the following rules:

1. A blank is used to separate each array element.
2. When applicable, a negative sign is written following an array element, table element, or field.
3. When the result field cannot be contained in a record, a continuation begins in position two of the following record.
4. When one or more elements of an array can be written on a single record, but the next element cannot be entirely contained on the record, then that next element will be written in position two of the next record.



IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Line	Form Type	Factor 1	Operation
01	C		
02	C		
03	C		
04	C		DEBUGPRNTOUT
05	C		
06	C		
07	C		
08	C	299	DEBUGPRNTOUT
09	C		

Output

```
DEBUG-          1 INDICATORS ON- L0 01 05 06 28
DEBUG-        299 INDICATORS ON- L0 01 05 07 10 28 99
```

Figure 157. Sample Coding and Output for Record 1

TIME (Time of Day) OPERATION (Model 15 only)

This calculation operation code allows you to access the system time of day as well as the system date. To use this operation code, columns 28-32 must contain the operation code TIME and the result field must specify the name of a numeric field. To contain the time of day, the field specified by the result field must be six-digit numeric (with no decimals). To contain both the time of day and the date, the result field must specify the name of a 12-digit numeric field (with no decimals). Whole arrays cannot be specified in the result field.

Columns 49-52 may be used to define the time of day field. Factor 1 and Factor 2 are not used and must be blank. "Half Adjust" and "Resulting Indicators" (columns 53-59) are not used and must also be blank.

If the field specified in the result field is 6-digit numeric, time will be returned in the following format:

hhmmss

Where hh is hours, mm is minutes, and ss is seconds. (See Figure 158 for an example of the time of day operation.)

If the field specified in the result field is 12-digit numeric, both time and date will be returned in one of the following formats, depending on how the system date was defined during system generation:

hhmmss	mmddyy
hhmmss	ddmmyy

Where hh is hours, mm is minutes, ss is seconds, dd is day, mm is month, and yy is year.

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line (3-5)	Form Type (6)	Control Level (LO-L9, LR, SR, AN/OR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators Plus/High 1>2 (54-55)	Minus/Low 1<2 (56-57)	Zero/Equal 1=2 (58-59)	Comments (60-74)
01	C		Ø1		TIME		TIMFLD	6	Ø					TIME ONLY
02	C													
03	C		Ø2		TIME		TIMFLD	12	Ø					TIME AND DATE
04	C													

Figure 158. Time of Day Operation

Output specifications describe your output records. These specifications may be divided into two general categories:

1. Record description entries (columns 7-31) which describe the output file records to be written or punched.

2. Field description entries (columns 23-74) which indicate the position and the format of data on the output record.

Write the specifications on the Output Sheet (Figure 159). The field description entries start one liner lower than record description entries.

COLUMNS 1-2 (PAGE)

See Chapter 2.

COLUMNS 3-5 (LINE)

See Chapter 2.

COLUMN 6 (FORM TYPE)

An *O* must appear in column 6.

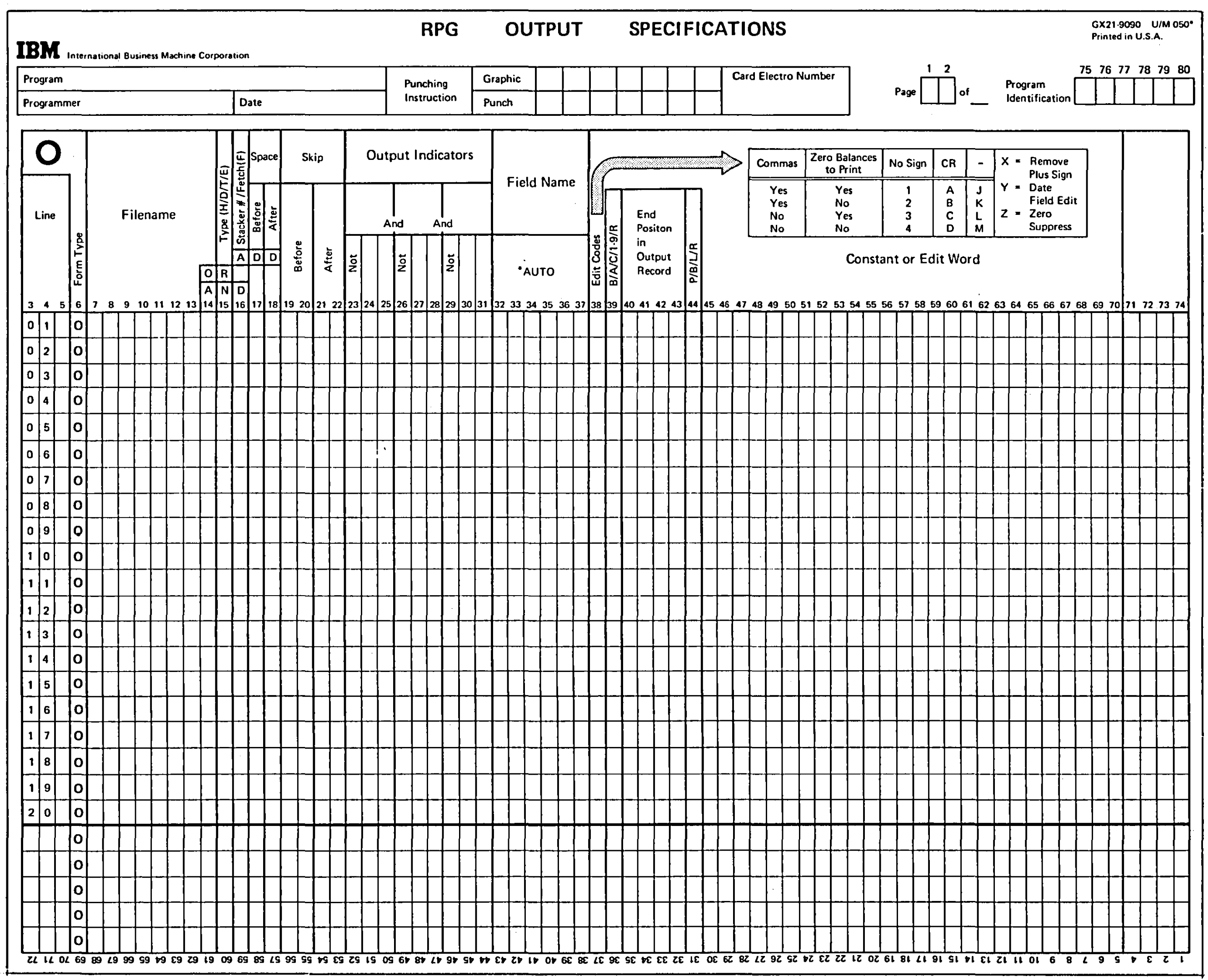


Figure 159. Output-Format Sheet

COLUMNS 7-14 (FILENAME)

Use columns 7-14 to identify the file to which records are to be written. The filename must begin in column 7. Use the same filename given in the file description specifications. You need to specify the output filename only once. That name, however, must be on the first line that identifies the file.

COLUMN 15 (TYPE)

Entry	*Explanation*
H	Heading records.
D	Detail records.
T	Total records.
E	Exception Records (records to be written during calculation time).

Use column 15 to indicate the type of record that is to be written. This record may be printed, written on disk, or punched or printed on a card. Perhaps the clearest method of describing output files is to enter the records for each file in this order: heading, detail, total, and exception (Figure 160, insert A).

Another method is to enter all headings records for all output files, then, all detail records for all output files, etc., as shown in Figure 160, insert B.

Use of heading and detail specifications together with control level and overflow indicators specifying when output records are to be written is described under *Columns 23-31, (Output Indicators)* in this chapter.

Heading records usually contain unchanging identifying information such as column headings, as well as page numbers and date.

Detail records are closely connected with input data. Most data in a detail record comes directly from the input record or is the result of calculations performed on data from the input record.

Total records usually contain data that is the end result of specific calculations on several detail records. Exception output conditioned by level indicators (L0-L9) or total output should not be specified for primary or secondary update files, as the results of the update will be unpredictable.

Exception records are written or punched during calculation time. This is an unusual case and can be indicated only when the operation code EXCPT is used. *E* may not be specified for a combined file. See *Operation Codes* in Chapter 8 for further information on the EXCPT operation.

COLUMNS 16-18 (ADD A RECORD)

Entry	*Explanation*
ADD	Add a record.

Columns 16-18 may be used to specify that a record is to be added to an input, output, or update file. The output device for these files *must* be a disk. An *A* must also be coded in column 66 of the File Description Specification Sheet for the file to which the record will be added.

ADD must appear in columns 16-18 of the first line for each record identified which is to be added.

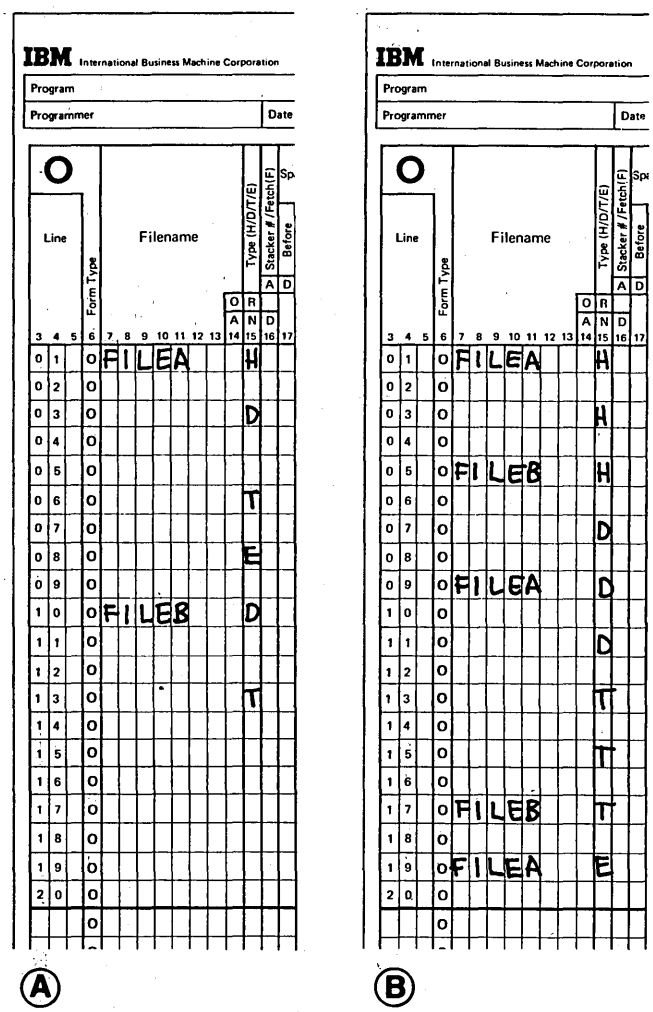


Figure 160. Order of Output Record Types

COLUMN 16 (STACKER SELECT/FETCH OVERFLOW)

Entry	*Explanation*
Blank	Cards automatically fall into predetermined stacker (primary hopper – stacker 1, secondary hopper – MFCU or MFCM (Model A2) stacker 4, MFCM (Model A1) stacker 5).
1-2	Indicates stacker you want to select for 1442.
1-4	Indicates stacker you want to select for MFCU or MFCM (Model A2).
1-5	Indicates stacker you want to select for MFCM (Model A1).
F	Fetch overflow.

Column 16 may be used for two different purposes:

1. To select a special stacker into which certain cards are to go.
2. To indicate that the overflow routine can be used at this point for a printer file.

Stacker Select

Use column 16 to indicate that certain cards are to be stacked in a specific stacker. If you make no entry, cards go into a predetermined stacker as follows:

	Primary hopper	*Secondary hopper*
MFCU	1	4
MFCM (Model A1)	1	5
MFCM (Model A2)	1	4
1442	1	N/A

Only combined or output card files may be stacker selected in the output specifications. If any output operations are to be performed on cards from a combined file that are also to be stacker selected, stacker selection should be done by the output specifications not by the input specifications. Stacker selection in output specifications overrides stacker selection in input specifications.

If stacker selection is done on the basis of matching records, it should only be done for detail output (D in column 15). It is only at this time the MR indicator signals the matching status of the card that is ready to be stacker selected.

OR lines may have different entries in column 16; AND lines may not. An OR line containing a blank in column 16 causes cards to fall into the normal stacker associated with the hopper used. The stacker select entry on the previous line is not assumed.

Fetch Overflow

When the fetch overflow routine is not used, the following usually occurs when the overflow line is sensed:

1. All remaining detail lines in that program cycle are printed (if a printer operation spaced or skipped to the overflow area).
2. All remaining total lines in that program cycle are printed.
3. All lines conditioned by an overflow indicator are printed.
4. Forms advance to a new page if a skip to a new page has been specified.

If you do not want all of the remaining detail and total lines printed on the page before overflow lines are printed and forms advance to the new page, you may cause overflow lines to be printed ahead of the usual time. This is known as fetching the overflow routine and is indicated by the entry in column 16. Overflow is fetched only if all the conditions specified by the indicators in columns 23-31 are met and an overflow has occurred. See *Columns 33-34* (Overflow Indicators) Chapter 4 for detailed information and examples of a fetched overflow routine.

The fetched overflow routine does not automatically cause forms to advance. A skip to line 01 (new page) must also be specified on a line conditioned by the overflow indicator in order to advance the forms.

F must be entered in column 16 of each OR line if you want to fetch the overflow routine for each record in the OR relationship.

COLUMNS 17-22 (SPACE/SKIP)

Columns 17-22 are used to specify spacing and line skipping for a printer file. If these columns are blank, single spacing occurs automatically after each line is printed.

Line spacing and skipping may be specified both before and after printing of a line. There may be as many as six spaces (three before, three after) between lines of printing. Only space before and space after can be specified on output for the printer-keyboard.

You may specify different spacing and skipping on OR lines. If no spacing or skipping entries are in the OR line, spacing and skipping is done according to the entries in the line preceding the OR line.

If both spacing and skipping are specified on the same line, they are done in this order:

1. Skip before.
2. Space before.
3. Skip after.
4. Space after.

Note: Because of hardware limitations on the 3284 Printer, it is necessary to space after at least one line. If a space after of zero is specified or implied (space and/or skip before specified with no space or skip after) the compiler will diagnose this and assume a space after of one in addition to any space and/or skip before specification.

COLUMNS 17-18 (SPACE)

Entry	*Explanation*
0	No spacing.
1	Single spacing.
2	Double spacing.
3	Triple spacing.

Spacing is used in reference to the lines on one page. You may indicate that spacing should be done before (column 17) or after (column 18) a line is printed. If the destination of a space operation is a line beyond the overflow line (but not on a new page), the overflow indicator turns on and remains on until all overflow lines are printed.

The console will always space before printing, due to the carriage return mechanism. Therefore, a space before entry blank, zero, or one will result in a single space before printing (Models 10 and 12 only).

Note: The 3284 Printer requires a space after of at least one line. If a space after of zero is either specified or implied (space and/or skip before specified with no space or skip after), a space after of one is assumed, in addition to any space and/or skip before specified.

COLUMNS 19-22 (SKIP)

Entry	*Explanation*
01-99	Lines 1-99.
A0-A9	Lines 100-109.
B0-B2	Lines 110-112.

Skipping refers to jumping from one printing line to another without stopping at lines in between. This is usually done when a new page is needed. A skip to a lower line number means advance to a new page. Skipping may also be used, however, when a great deal of space is needed between lines.

The entry must be the two-digit number which indicates the number of the next line to be printed. You may indicate that skipping should be done before (columns 19-20) or after (columns 21-22) a line is printed. If you specify a skip to the same line number as the forms are positioned on, no movement of the paper occurs. If the destination of a skip operation is a line beyond the overflow line (but not on a new page), the overflow indicator is turned on and remains on until all overflow lines are printed. The destination line of a skip operation must not be beyond the form length defined on the Line Counter Sheet.

COLUMNS 23-31 (OUTPUT INDICATORS)

Entry	*Explanation*
01-99	Any resulting indicator, field indicator, or record identifying indicator previously specified.
L1-L9	Any control level indicators previously specified.
H1-H9	Any halt indicators previously specified.
U1-U8	Any external indicator set prior to program execution.
OA-OG, OV	Any overflow indicator previously assigned to this file. (See also *overflow indicators*, columns 33-34 of file description specifications.)
MR	Matching record indicator.
LR	Last record indicator.
1P	First page indicator.
L0	Level zero indicator.

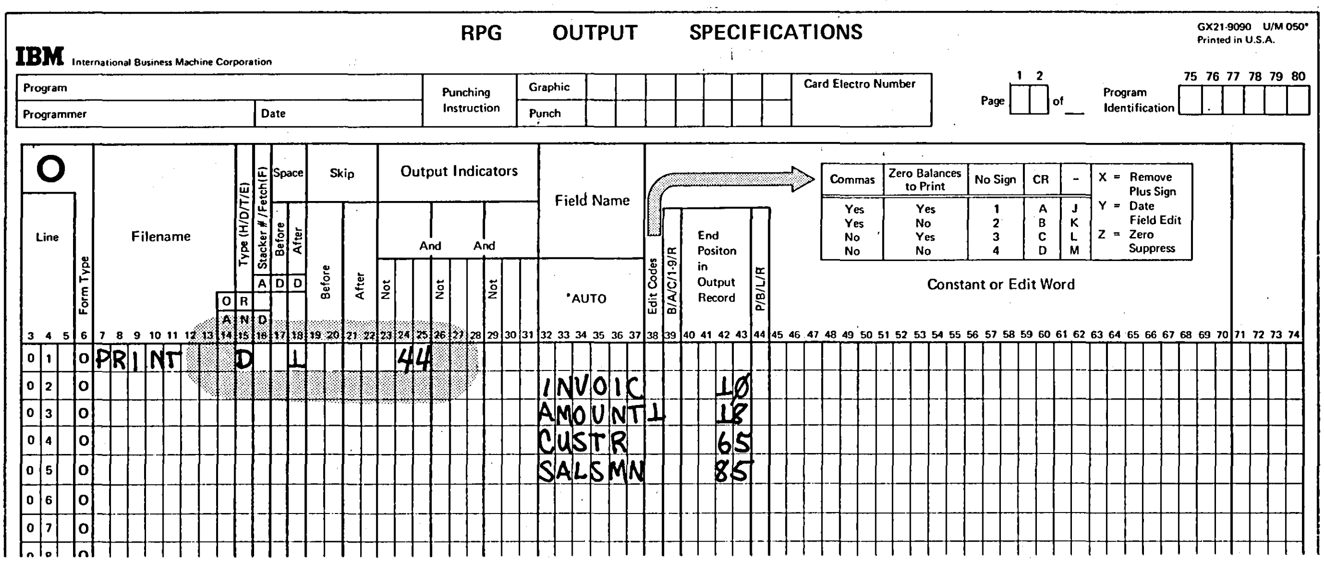
Use output indicators to give the conditions under which output operations are to be done. More specifically, use them to tell:

1. When you want to output a line (see *Examples, Example 1*).

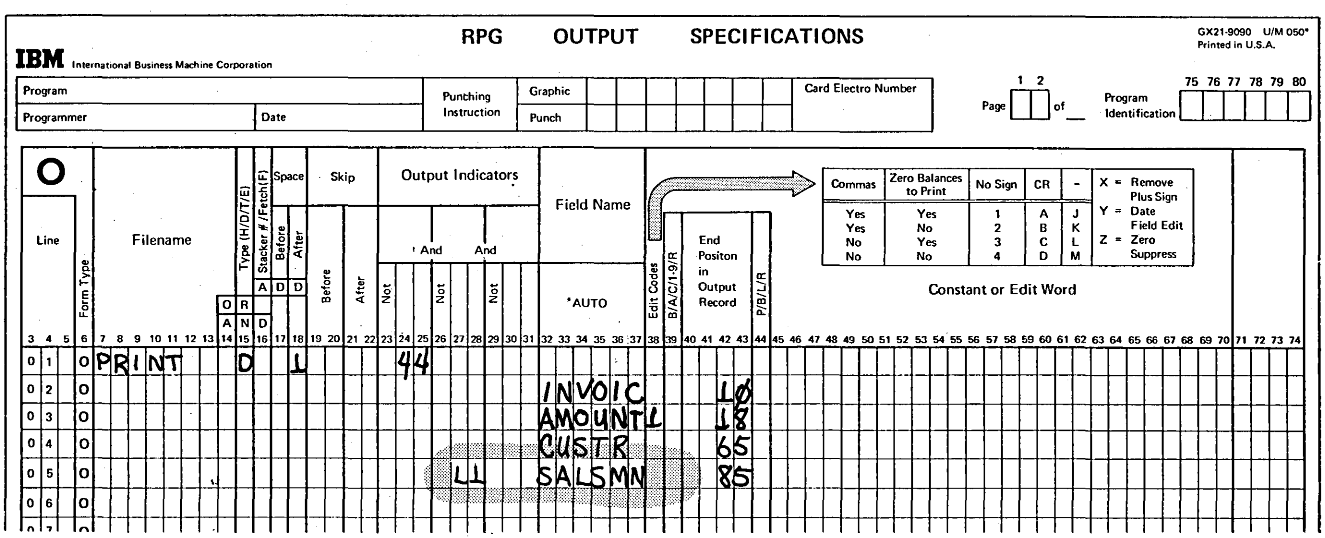
2. When you want to output a field (see *Examples, Example 2*).

When you use an indicator to condition an entire line of print, place it on the line which specified the type of record (Figure 161, insert A). Place an indicator which conditions when a field is to be printed on the same line as the field name (Figure 161, insert B).

There are three separate output indicator fields (columns 23-25, 26-28, and 29-31). One indicator may be entered in each field. If these indicators are on, the output operation will be done. An *N* in the column (23, 26, or 29) preceding each indicator means that the output operation will be done only if the indicator is not on. No output line should be conditioned by all negative indicators (at least one of the indicators used should be positive). If all negative indicators condition a heading or detail operation, the operation is performed at the beginning of the program cycle when 1P lines are written. The overflow indicators may not be specified on an *E* (exception output) line.



(A)



(B)

Figure 161. Output Indicator

Warning: When defining records of combined or update files, avoid writing or punching multiple records on one cycle. In Figure 162, for example, if indicators 02 and 03 are both on, two records qualify for output on the same cycle. Results are unpredictable. Writing or punching to a combined file or update file can only occur once for each cycle.

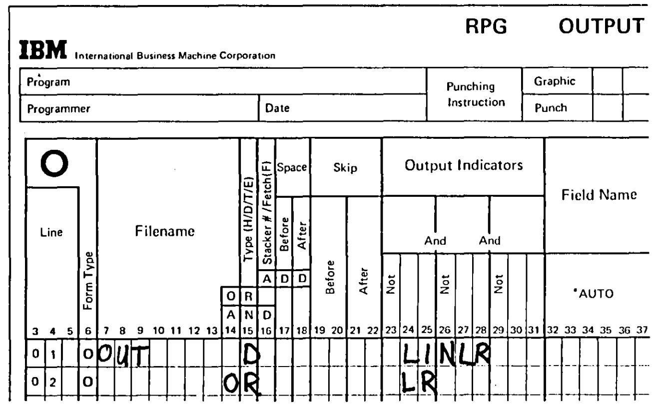
In System/3 Disk RPG II, all total lines conditioned by LR will be performed last.

AND and OR Lines

If you need to use more than three indicators to condition an output operation, you may use an AND line. Enter the word AND in columns 14-16 and as many indicators as needed. The condition for all indicators in an AND relationship must be satisfied before the output operation is done.

Output indicators may also be in an OR relationship. If either or both of the OR conditions are met, the output operation will be done. OR lines are indicated by the word OR in columns 14-15. Both AND or OR lines may be used together to condition an entire output line. A maximum of 20 AND, OR, or mixed AND and OR lines are allowed in an output operation. AND and OR lines cannot be used to condition a field (see *Examples, Example 3*).

The use of an L0-L9 indicator in an OR relationship with an LR indicator can result in the specified operation being done twice when LR is on. One operation is performed during LR processing and the other at detail or total time. The following example shows how to eliminate duplicate output during the LR cycle (LR may have been set on during the previous cycle).



External Indicators

A file named in the output-format specifications may be conditioned by an external indicator in the file description specifications. External indicators can also be used to condition a record or field.

No output can occur to a file if it is conditioned by an external indicator and that indicator is off. Therefore, if a file is conditioned by an external indicator, all output records handled by the file must also be conditioned by the same indicator.

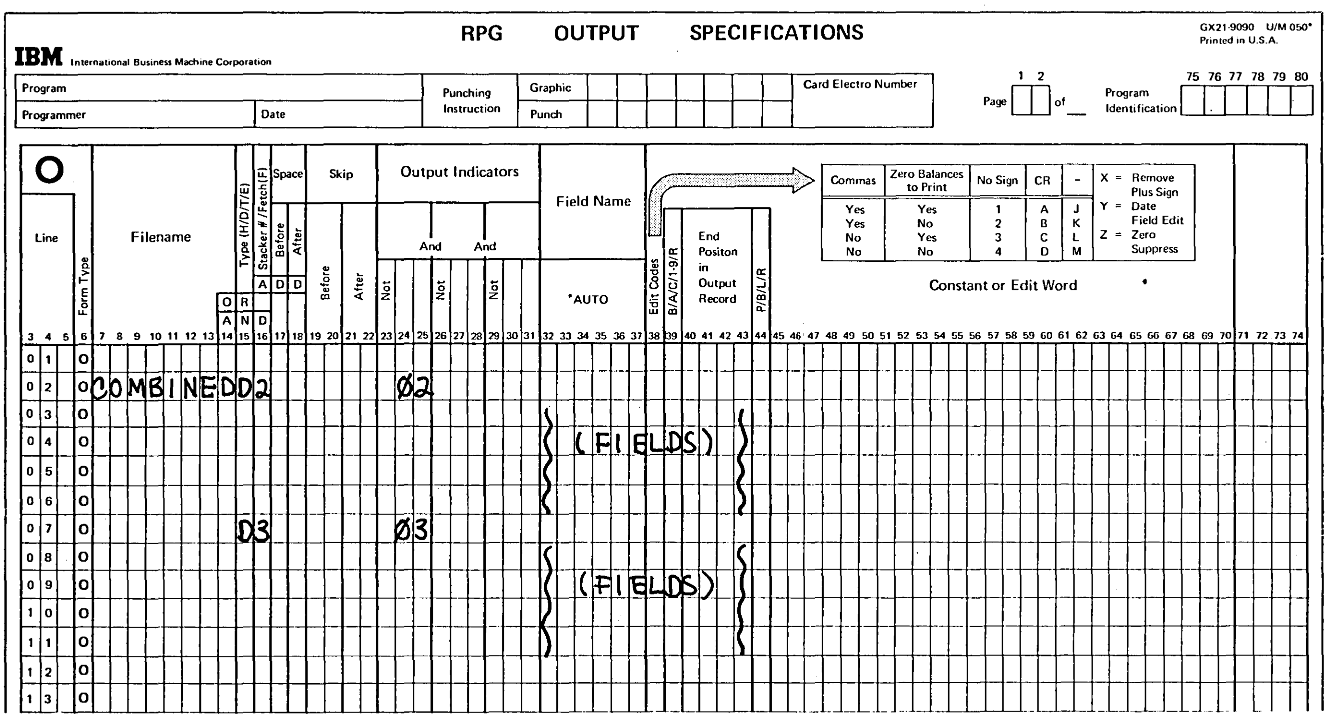


Figure 162. Two Records from a File Qualifying for Output on the Same Cycle

Control Level Indicators

Control level indicators entered in columns 23-31 of this sheet specify when output records or fields are to be written:

1. If the control level indicator is entered along with a *T* in column 15 and no overflow indicator is used, the record is written only after the last record of a control group has been processed.

2. If the indicator is entered along with a *D* in column 15 and no overflow indicator is used, the record is written only after the first record of the new control group has been processed.

3. If the control level indicator is entered along with an overflow indicator, the record is written after the overflow line has been sensed (provided a control break has also occurred).

Overflow Indicators

Overflow indicators are used to condition output operations on the printer. The operations conditioned by the overflow indicator are done only after the overflow line has been passed.

If you have not assigned an overflow indicator to the printer file in the file description specifications, you may not use an overflow indicator in the output specifications. In this case, advancing the forms to a new page is handled automatically, even though no overflow indicator has been assigned. If any specification line not conditioned by an overflow indicator specifies a skip to a line on a new page, overflow indicators turn off before forms advance to a new page.

An overflow indicator may appear on either AND or OR lines. However, only one overflow indicator may be associated with one group of output indicators. That overflow indicator must also be the same indicator associated with the file on the File Description Sheet.

When the overflow indicator is used in an AND relationship with a record identifying indicator, unusual results are often obtained. This is because the record type might not be the one read when overflow has occurred. Thus, the record type indicator is not on and all lines conditioned by both overflow and record type indicators do not print.

If at all possible, use overflow indicators and record type indicators in an OR relationship when conditioning output lines.

An overflow indicator cannot condition an exception line (*E* in column 15), but may condition fields within the exception record.

If an overflow indicator is not specified, the RPG II compiler assigns *OA*.

First Page Indicator

The first page (1P) indicator is usually used to allow printing on the first page. It may also be used in connection with the overflow indicator to allow printing on every page (see *Examples, Example 4*). The information printed out on the line conditioned by the 1P indicator is usually constant information used as headings. The constant information is specified on the Output Sheet, columns 45-70.

The 1P indicator is used only with heading or detail output lines. It cannot be used to condition total or exception output lines. Use this indicator only when other indicators (control level or resulting indicators) cannot be used to control printing on every page.

All lines conditioned by the 1P indicator are written out even before the first record from any input file is processed. Therefore, do not condition output fields (except PAGE and UDATE) which are based upon data from input records by the 1P indicator. Calculation operations cannot be conditioned by the 1P indicator.

When printed output is being spooled on Model 15, the 1P halt allowing forms alignment is not issued. To perform alignment of printed output when using print spooling, you may use the ALIGN-YES parameter on the PRINTER OCL statement. For more information, see the *IBM System/3 Model 15 System Control Programming Reference Manual*, GC21-5077.

Error Conditions

On certain error conditions, you may not want output performed. Indicators can be used to prevent the data that caused the error from being used (see *Examples, Example 5*).

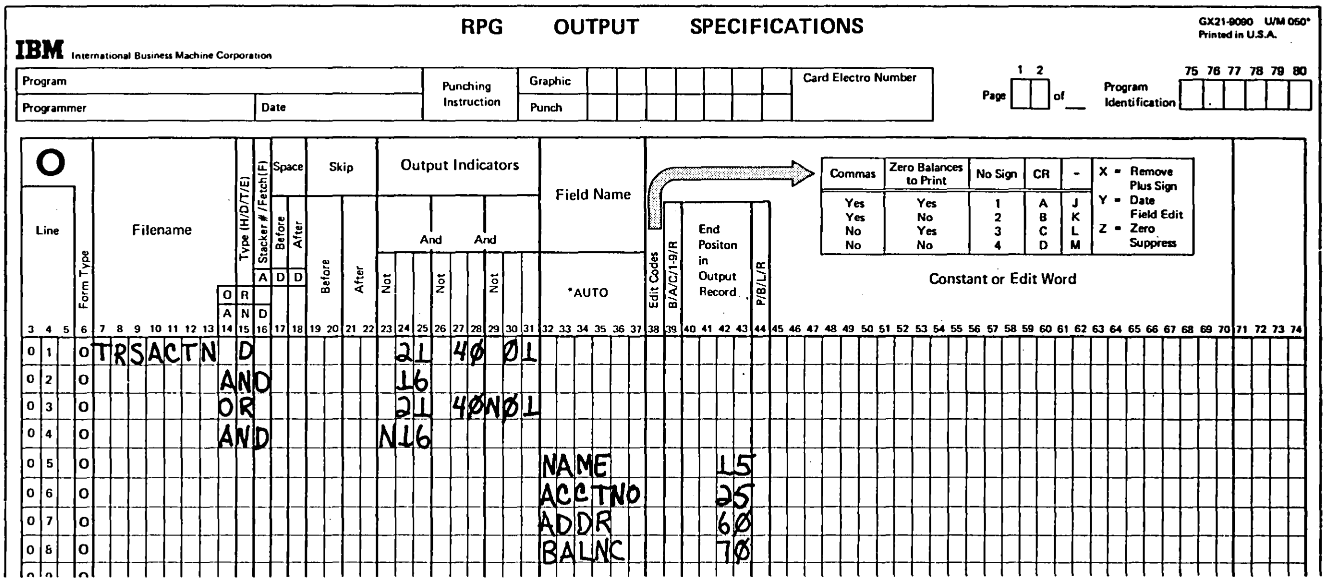
Examples

Example 1: Figure 161, insert A shows the use of one indicator to condition an entire line of printing. When 44 is on, the fields named INVOIC, AMOUNT, CUSTR, and SALSMN are all printed.

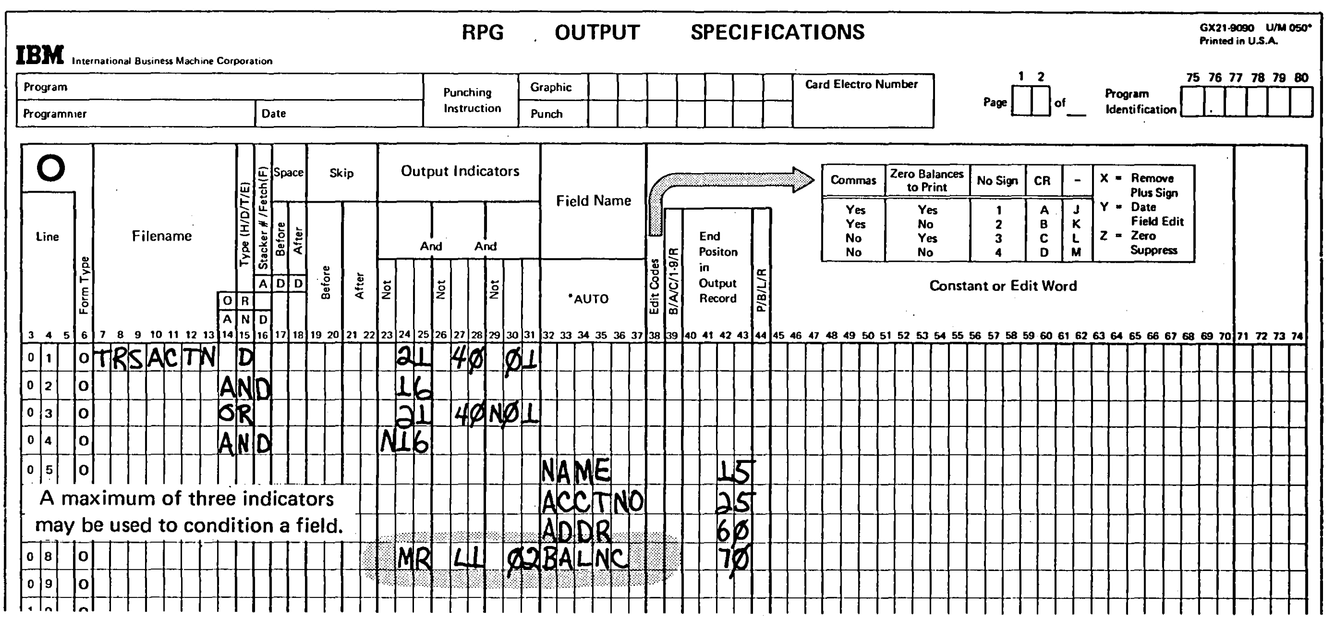
Example 2: Figure 161, insert B shows the use of a control level indicator to condition when one field should be printed. When indicator 44 is on, fields INVOIC, AMOUNT, and CUSTR are always printed. However, SALSMN is printed only if 44 and L1 are on.

Example 3: The use of indicators in both AND and OR lines to condition an output line is shown by Figure 163, insert A. The specifications in lines 01-04 say that the detail line is written if either one of two sets of conditions is met. If indicators 21, 40, 01, and 16 are all on, the line is written, or if 21 and 40 are on and 01 and 16 are off, the line is also written.

A maximum of three indicators may be used on the Output Sheet to condition a field since AND and OR lines may not be used to condition an output field (Figure 163, insert B).



Ⓐ

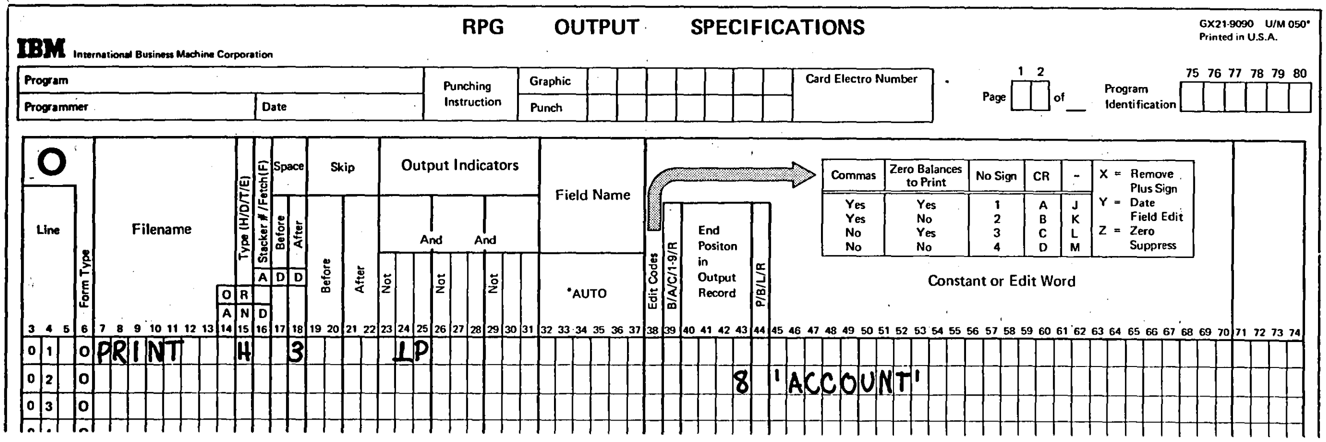


Ⓑ

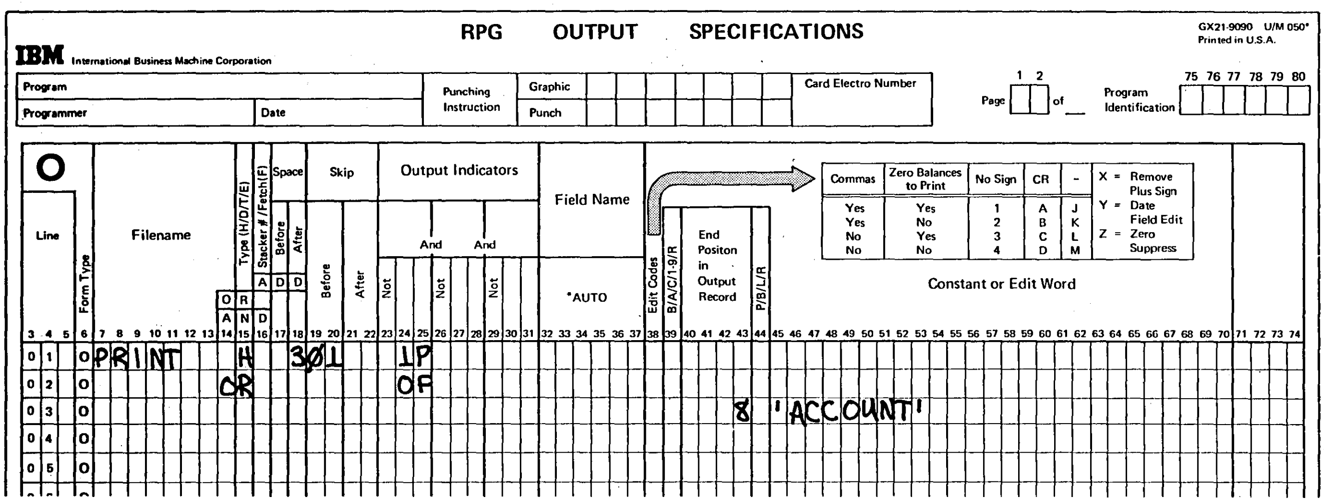
Figure 163. Output Indicators

However, you can condition an output field with more than three indicators by using the SETON operation in calculations. For instance, indicators 10, 12, 14, 16, and 18 are to condition an output field named PAY. In calculation specifications, you can SETON indicator 20 if indicators 10, 12, and 14 are on. Then condition the output field PAY on indicators 20, 16, and 18 on the Output Sheet.

Example 4: Figure 164, insert A shows how the 1P indicator is used when headings are to be printed on the first page only. Figure 164, insert B shows the use of the 1P indicator and overflow indicator to print headings on every page.



Ⓐ



Ⓑ

Figure 164. 1P Indicator

Example 5: Figure 165 shows coding necessary to check for an error condition and to stop processing on and writing from the record in error. If FIELDB contains all zeros, halt indicator H1 turns on (see line 03 of Figure 165, insert A). In the calculation specifications, if H1 is on, resulting indicator 02 turns off (see line 01 of Figure 163, insert B). On the Output Sheet, FIELDA and FIELDB are printed only if 01 is on (see lines 03 and 05 of Figure 165, insert C). Therefore, if indicator 01 is off, fields *A* and *B* are not printed. Use this general format when you do not want information that is in error to be printed.

COLUMNS 32-37 (FIELD NAME)

In columns 32-37, use one of the following to name every field that is to be written out.

- Any field name previously used in this program.
- The special words PAGE, PAGE1, PAGE2, *PLACE, *PRINT, UDATE, UDAY, UMONTH, and UYEAR.
- A table name, array name, or array element.

The field names used are the same as the field names on the Input Sheet (columns 53-58) or the Calculation Sheet (columns 43-48). Do not use these columns if a constant is used (see *Columns 45-70* in this chapter). If a field name is entered in columns 32-37, columns 7-22 must be blank.

Fields may be listed on the sheet in any order since the sequence in which they appear on the printed form is determined by the entry in columns 40-43. However, they are usually listed sequentially. If later fields overlap the first fields specified, the data which is overlayed is lost.

The sign (+ or -) of a numeric field is in the units position (rightmost digit). A minus sign in the units position prints as a letter unless the field is edited (see *Column 38* in this chapter).

PAGE

PAGE is a special word which causes automatic numbering of your pages. Enter the word PAGE, PAGE1, or PAGE2 in these columns if you wish pages (or an individual record) to be numbered. When a PAGE field is named in these columns without being defined elsewhere, it is assumed to be a four-position numeric field with no decimal positions. However, a PAGE field can be defined in input or calculation specifications and may be up to 15 positions long. A PAGE field defined elsewhere must be defined with zero decimal positions. Leading zeros are suppressed, and the sign is not printed in the rightmost position unless an edit word or edit code is specified. The page number starts with 1 unless otherwise specified, and one is automatically added each time the PAGE field is written. See *Columns 53-58* in Chapter 7 for information concerning page numbering starting at a number other than 1.

It is possible at any point in your job to restart the page numbering sequence. To do this, set the PAGE field to zero before it is printed. One method of setting PAGE field to zero is to use Blank After (see *Column 39* in this chapter). Another way is to use an output indicator. A PAGE field will always be printed even though the field is conditioned by an indicator. If the indicator is on, the PAGE field is set to zero, and one is added before it is written. Remember that one is always added to the PAGE field before it is written (see *Examples, Example 1*).

The three possible PAGE entries, PAGE, PAGE1, and PAGE2, may be used for different output files. Do not use the same name for two different output files.

*PLACE

*PLACE is a special RPG II word which makes it possible to write or punch the same field in several locations on one record without having to name the field and give its end position each time the field is written or punched. The fields are written or punched in the same relative positions ending in the column specified by *PLACE. For example, if you wish fields A, B, and C to appear twice on one line, you can specify this in two ways:

1. Define each field and its corresponding end position each time it is to be printed (Figure 166, insert A).
2. Use the special word *PLACE (Figure 166, insert B).

Both coding methods produce a line which looks like this:

Print positions

1 — 10	11 — 20	21 — 30	31 — 40	41 — 50	51 — 60
FIELDA	FIELDB	FIELDC	FIELDA	FIELDB	FIELDC

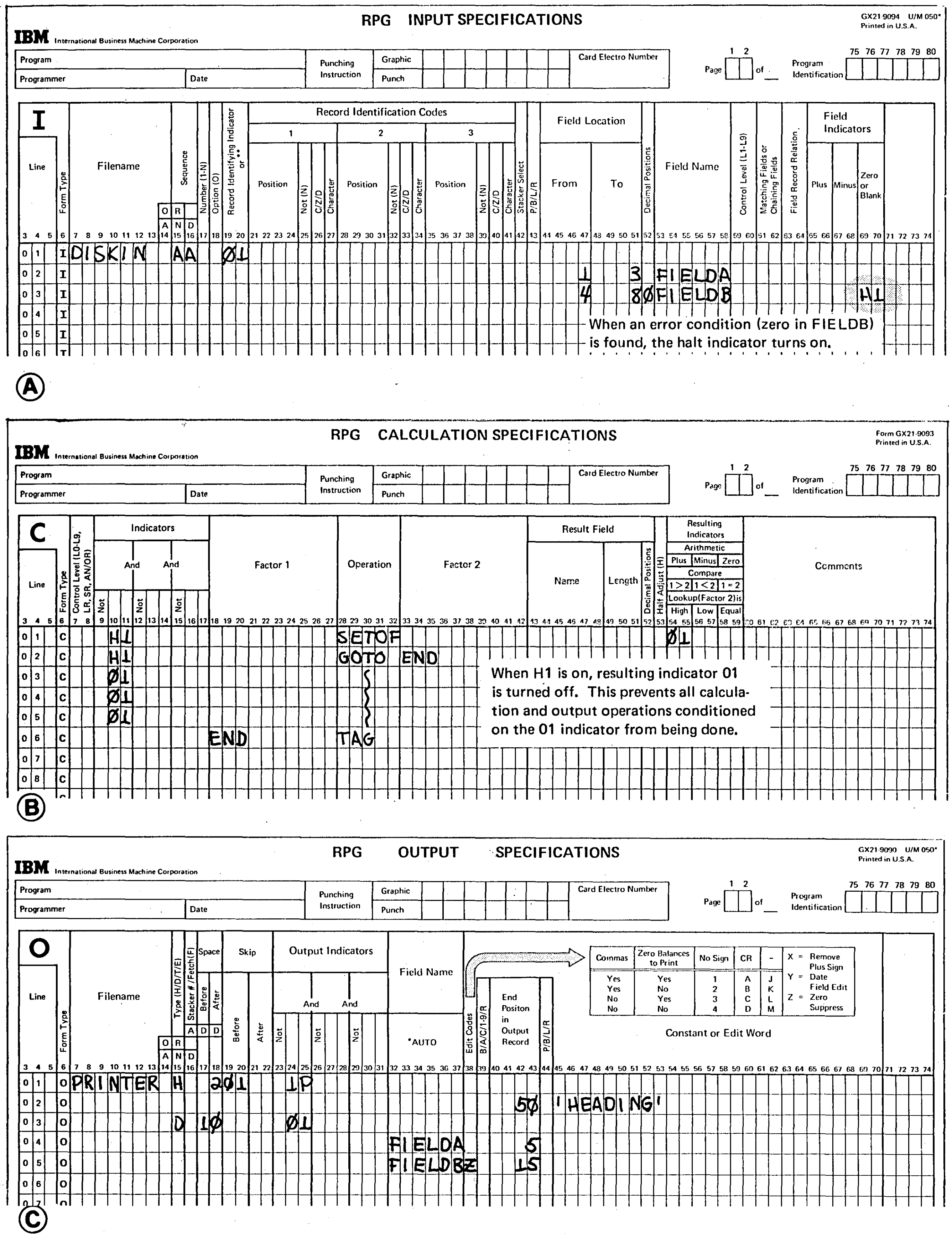


Figure 165. Preventing Fields From Printing

RPG OUTPUT SPECIFICATIONS

Line	Form Type	Filename	Type (H/D/T/E)	Space After	Field Name	End Position in Output Record
01	O	PRINT	D	1		
02	O				FIELDA	1Ø
03	O				FIELDB	2Ø
04	O				FIELDC	3Ø
05	O				FIELDA	4Ø
06	O				FIELDB	5Ø
07	O				FIELDC	6Ø
08	O					
09	O					

Ⓐ

RPG OUTPUT SPECIFICATIONS

Line	Form Type	Filename	Type (H/D/T/E)	Space After	Field Name	End Position in Output Record
01	O	PRINT	D	1		
02	O				FIELDA	1Ø
03	O				FIELDB	2Ø
04	O				FIELDC	3Ø
05	O				*PLACE	6Ø
06	O					
07	O					
08	O					
09	O					

Ⓑ

Figure 166. Printing Fields Twice on the Same Line

When using *PLACE, all fields named for each record type (H/D/T/E) are written or punched as usual in the locations specified. The entry *PLACE then causes all of these same fields to be written or punched ending at the position specified in the *PLACE statements.

When using *PLACE, remember:

1. *PLACE must be specified after the field names which are to be placed in different positions in one line (see *Examples, Example 2*).

2. *PLACE causes *all* fields (in a record type) above the *PLACE entry to be written or punched.

3. *PLACE must appear on a separate specification line for every additional time you want the field or group of fields written or punched.

4. The end position specified for *PLACE must be at least twice the highest previously specified field end position, but not greater than 256.

5. An end position must be specified for every *PLACE line. If you do not allow enough space for all fields and constants prior to the *PLACE to be printed again, overlapping occurs, with the *PLACE field overlapping prior characters. The end position must not be lower than the preceding end position specification.

6. The leftmost position of the fields to be moved by the *PLACE specification is always assumed to be position 1.

7. The high-end position to be used by *PLACE cannot be defined by a whole array. If a whole array does have the highest end position of all fields preceding the *PLACE, a field must be defined which has an end position greater than the end position of the whole array. This field can be a one-position blank constant.

8. When *PLACE is specified for card output, the fields and constants named above *PLACE will be repunched. Any printed output on the cards will not be reprinted unless an * is entered in column 40 (MFCU only) of the same line as *PLACE.

9. A *PLACE specification must not be conditioned by indicators in columns 23-31. *PLACE is automatically conditioned by the same indicators which condition the field or fields to be repeated.

Note: Attempts to use the *PLACE function for other than its defined purpose may produce unpredictable results.

*PRINT

*PRINT is a special RPG II word which causes fields and constants that were punched in the card to be printed on the card. This enables you to more easily determine what information is found on the card. For the MFCU, *PRINT prints the field in the positions which correspond one-for-one to the columns in which the field is punched (see *Examples, Example 3*). For the MFCM, *PRINT prints the field in the following manner.

Punched	*Printed*
Columns 1-64	Positions 1-64 by print head 1.
Columns 65-80	Positions 49-64 by print head 2.

When using *PRINT, remember:

1. *PRINT may be used only once for each record.

2. *PRINT must be specified after all punch fields which are to be printed on the card are named.

3. The *PRINT specification may be conditioned by indicators in columns 23-31. Columns 7-22 and 38-74 may not be used.

4. *PRINT may be used on a card file only.

If you want to print the fields in positions other than those which correspond to the punch positions of the fields, you must use the card printing option (see *Columns 40-43* in this chapter).

Date Field

Often you want the date to appear on your printed report, punched card, or output record. Use special words UDATE, UMONTH, UDAY, and UYEAR to get the date field you desire. The date is entered by using a DATE OCL statement (Models 10, 12, and 15) or a DATE OCC (Model 15 only). See the appropriate SCP reference manual listed under *Related Publications* in the Preface.

1 UDATE gives a six-character numeric date field in one of two formats (d, m, and y are the day, month and year positions in the UDATE field):

   a. United States (mmddyy).
   b. United Kingdom/World Trade (ddmmyy).

   The format is specified by an entry in *Column 21* of the control card. The edited date field is eight characters long, in one of three formats:

   a. United States (MM/DD/YY).
   b. United Kingdom (DD/MM/YY)
   c. World Trade (DD.MM.YY).

2. UDAY may be used for days only, UMONTH for months only, and UYEAR for years only.

3. These fields may not be changed by any operations specified in the program.

Examples

Example 1: Figure 167 shows how an output indicator can be used to reset a PAGE field to zero. When indicator 15 is on, the PAGE field is reset to zero and one is added before the field is printed. When 15 is off, one is added to the contents of the PAGE field before it is printed.

Example 2: Figure 168 shows the use of the special word *PLACE to print the same fields several times on the same line. Fields A, B, and C are to be printed four times on one line (Figure 168, insert A). In Figure 168, insert B *PLACE is specified after the fields which are to be printed several times on the same line. All fields to which *PLACE applies appear on the same record. The second *PLACE causes the original three fields to be repeated on the printed line. Field D, which appears on the total record, is not affected by *PLACE.

Notice that an end position (columns 40-43) is given for every *PLACE. Fields A, B, and C have a total length of 15 characters; thus the end position for each *PLACE allows room for printing 15 additional characters on the output line. The resulting printed line is 60 characters long. There is no overlapping of output fields.

Note: If the end position given for the *PLACE field does not allow room for all characters to be repeated, previous characters in the output line are overlaid by the *PLACE field.

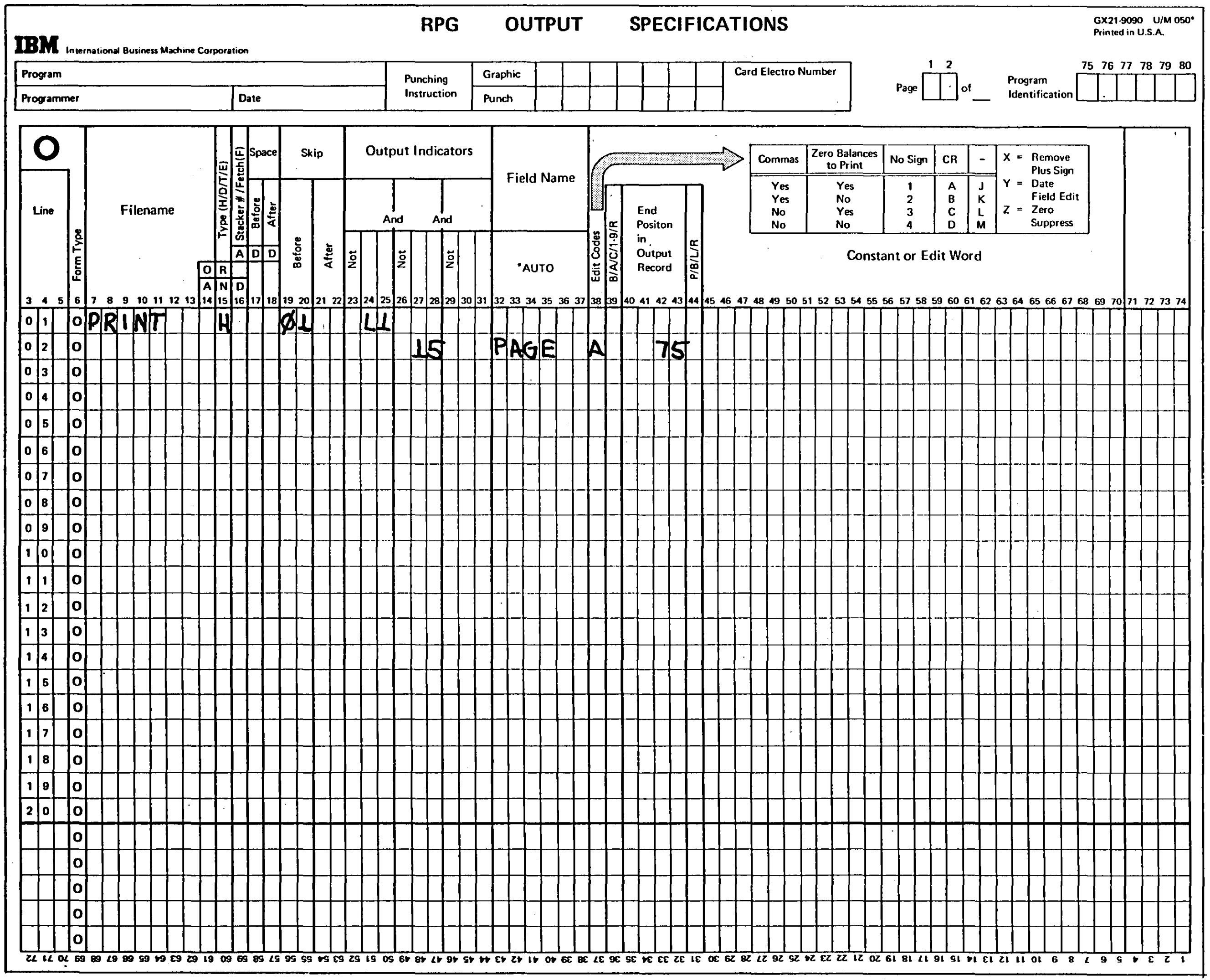
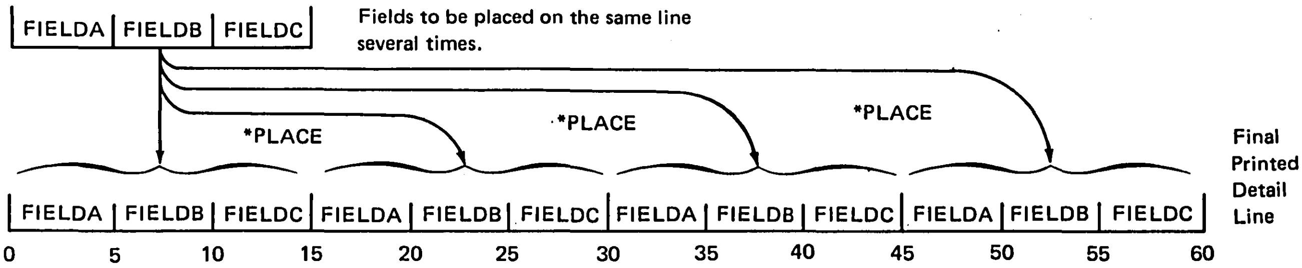
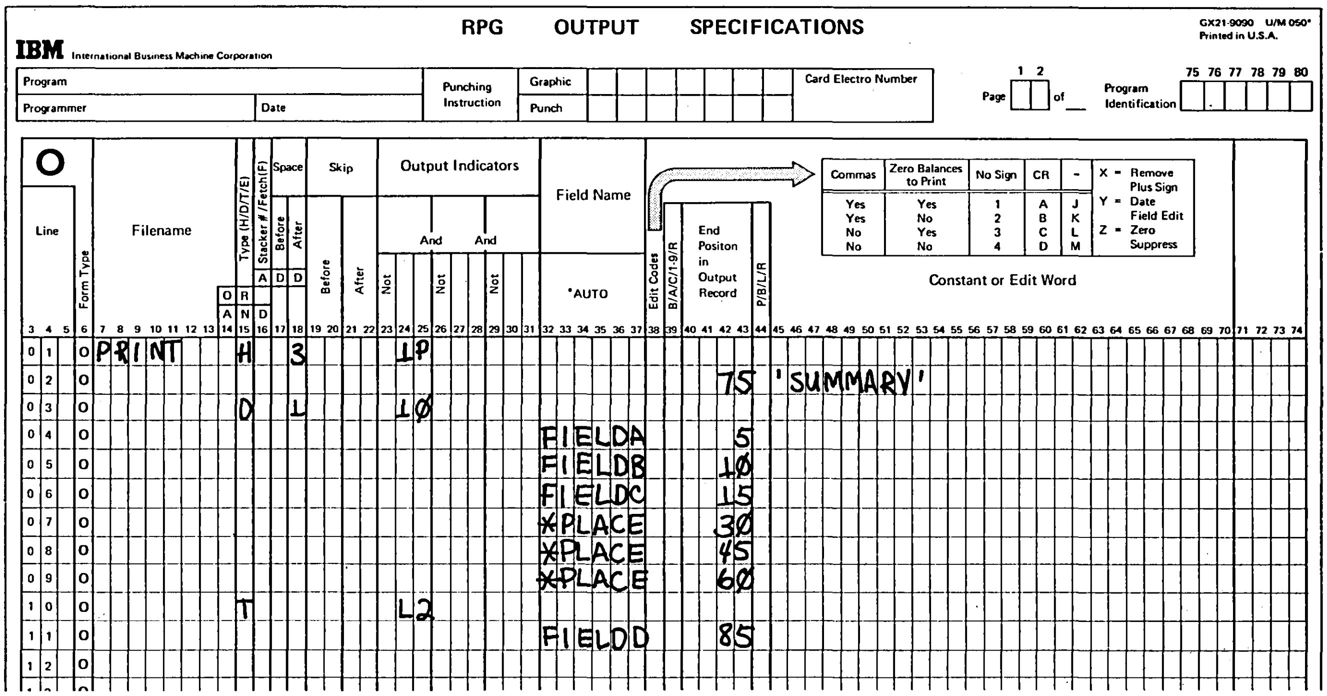


Figure 167. Resetting the PAGE Field to Zero



Ⓐ



IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 U/M 050* Printed in U.S.A.

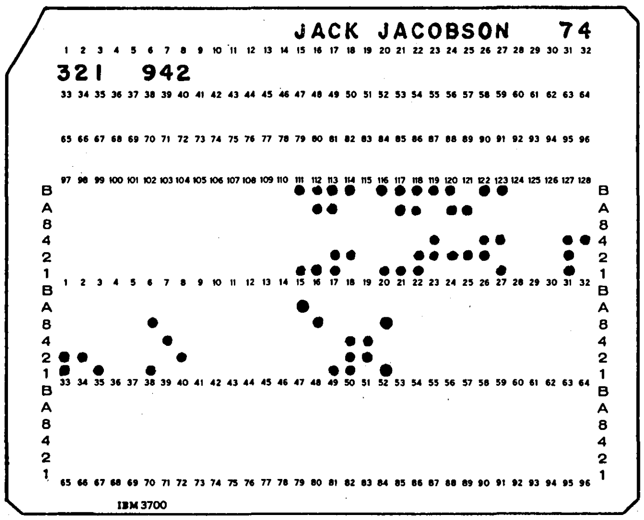
Line	Form Type	Filename	Type (H/D/T/E)	Space After	Output Indicators	Field Name	End Position in Output Record	Constant or Edit Word
01	O	PRINT	H	3	1P			
02	O						75	'SUMMARY'
03	O		D	1	10			
04	O					FIELDA	5	
05	O					FIELDB	10	
06	O					FIELDC	15	
07	O					*PLACE	30	
08	O					*PLACE	45	
09	O					*PLACE	60	
10	O		T		L2			
11	O					FIELDD	85	
12	O							

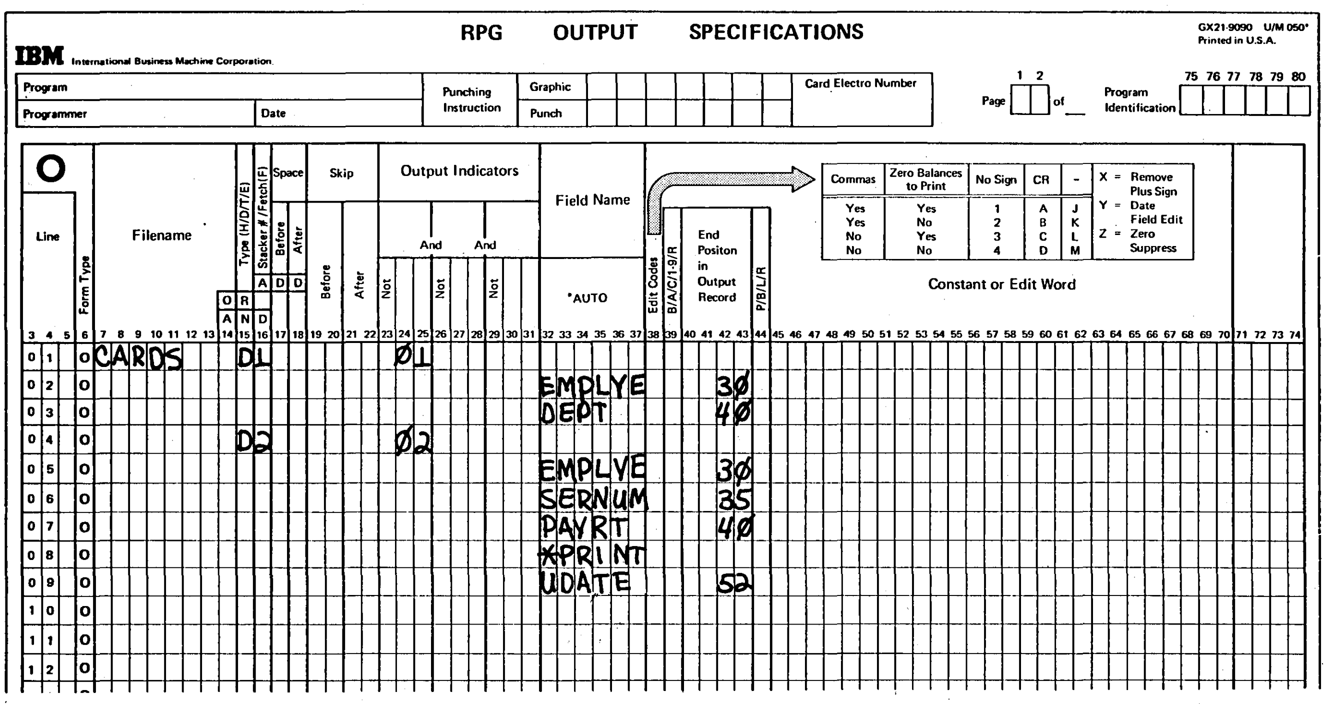
Ⓑ

Figure 168. *PLACE

Example 3: Figure 169 shows how the special word *PRINT may be used to cause printing of the output fields on the punched cards. The fields EMPLYE, SERNUM, and PAYRT are to be punched on the card (specification lines 05-07). The *PRINT entry in line 08 causes the three fields written above the *PRINT entry (EMPLYE, SERNUM, and PAYRT) to print on the card in positions corresponding one-for-one to the punch positions (see Figure 169). The UDATE field (line 09) is punched but not printed because it is written after the *PRINT entry.

Notice in Figure 169 that *PRINT is specified after the fields which are to be printed. All fields to which *PRINT apply appear on the same record. Therefore, the *PRINT entry applies only to fields specified in lines 05-07, not to fields specified in lines 02 and 03.





Line	Form Type	Filename	Type	Output Indicators	Field Name	End Position in Output Record
01	O	CARDS	D1	01		
02	O				EMPLYE	30
03	O				DEPT	40
04	O		D2	02		
05	O				EMPLYE	30
06	O				SERNUM	35
07	O				PAYRT	40
08	O				*PRINT	
09	O				UDATE	52
10	O					
11	O					
12	O					

Figure 169. *PRINT

COLUMN 38 (EDIT CODES)

Use column 38 when you want to:

1. Suppress leading zeros for a numeric field.

2. Omit a sign from the low order position of a numeric field.

3. Punctuate a numeric field without setting up your own edit word.

A table summarizing the edit codes that can be used is printed above columns 45-70 on the Output Sheet.

Each edit code punctuates differently. If you use an edit code in column 38, columns 45-70 must be blank unless asterisk fill or a floating dollar sign is required ('*' or '$' entered in columns 45-47). If an edit code is used to punctuate an array, two spaces are left between elements of the array to the left of each element. Only unpacked numeric data can be edited.

Figure 169 shows the edit codes and how data looks when it is edited. Each code punctuates the field a little differently. All codes suppress leading zeros, except the J World Trade format for output (J-entry in column 21 of the control card specifications). For this J-entry, all zero balances and balances with zero values to the left of the decimal

Edit Codes	Positive Number - Two Decimal Positions	Positive Number - No Decimal Positions	Negative Number - * Three Decimal Positions	Negative Number - * No Decimal Positions	Zero Balance - United States, United Kingdom	Zero Balance - World Trade ** /	Zero Balance - World Trade ** J	Zero Balance - No Decimal Positions
Unedited	1234567	1234567	00012}	00012}	000000	000000	000000	000000
1	12,345.67	1,234,567	.120	120	.00	,00	0,00	0
2	12,345.67	1,234,567	.120	120				
3	12345.67	1234567	.120	120	.00	,00	0,00	0
4	12345.67	1234567	.120	120				
A	12,345.67ƀƀ	1,234,567ƀƀ	.120CR	120CR	.00	,00	0,00	0
B	12,345.67ƀƀ	1,234,567ƀƀ	.120CR	120CR				
C	12345.67ƀƀ	1234567ƀƀ	.120CR	120CR	.00	,00	0,00	0
D	12345.67ƀƀ	1234567ƀƀ	.120CR	120CR				
J	12,345.67ƀ	1,234,567ƀ	.120–	120–	.00	,00	0,00	0
K	12,345.67ƀ	1,234,567ƀ	.120–	120–				
L	12345.67ƀ	1234567ƀ	.120–	120–	.00	,00	0,00	0
M	12345.67ƀ	1234567ƀ	.120–	120–				
X	1234567	1234567	00012}	00012}	000000	000000	000000	000000
Y			0/01/20	0/01/20	0/00/00	0.00.00	0.00.00	0/00/00
Ƶ	1234567	1234567	120	120				

* The character } is a negative zero. It is printed for the 64 character set, but not for the 48 character set.

** Zero balances for the World Trade format are printed or punched in two ways, depending on the entry made in column 21 of the control card specifications. Two decimal positions are used for illustration.

Figure 170. Examples of Edit Code Usage

comma are written or punched with one leading zero (0,00 or 0,04). If an edit code is specified on the Output Sheet, and the edit code is to print zero balances, a zero balance field will always have a zero to the left of the decimal comma. The edit code cannot suppress it.

Normally, when you use an edit code in column 38, you cannot define an edit word in columns 45-70; however, there are two exceptions:

1. If you want leading zeros replaced by asterisks, enter '*' in columns 45-47 of the line containing the edit code.

2. If you want a dollar sign to appear before the first digit in the field (floating dollar sign), enter '$' in columns 45-47 of the line containing the edit code.

Asterisk fill and floating dollar sign are not allowed with *X, Y,* and *Z* edit codes.

It is also possible to have a dollar sign appear before the asterisk fill (fixed dollar sign). This is done in the following way:

1. Place a dollar sign constant one space before the beginning of the edited field.

2. Place '*' in column 45-47 of the line containing the edit code.

Figure 171 shows the effect different edit codes have on the same field with a specified end position for output.

COLUMN 39 (BLANK AFTER)

Entry	*Explanation*
Blank	Field is not to be reset (blanked or zeroed) after writing.
B	Field is to be reset (blanked or zeroed) after writing.

Use column 39 to reset a field to zeros or blanks. Numeric fields are set to zero and alphameric fields are set to blanks. This column must be blank for Look-Ahead fields, Udate fields (UDATE, UDAY, UMONTH, UYEAR), and constants.

Edit Codes	Negative Number —Two Decimal Positions— End Position Specified as 10.								
	Output Print Positions								
	3	4	5	6	7	8	9	10	11
Unedited				0	0	4	1	K*	
1					4	.	1	2	
2					4	.	1	2	
3					4	.	1	2	
4					4	.	1	2	
A			4	.	1	2	C	R	
B			4	.	1	2	C	R	
C			4	.	1	2	C	R	
D			4	.	1	2	C	R	
J				4	.	1	2	-	
K				4	.	1	2	-	
L				4	.	1	2	-	
M				4	.	1	2	-	
X				0	0	4	1	K*	
Y			0	/	4	1	/	2	
Z						4	1	2	

* K represents a negative 2

Figure 171. Effect of Edit Codes on End Position

Resetting fields to zeros is useful when you are accumulating and printing totals for each control group. After finding the total for one group and printing it, you want to start accumulating totals for the next group. Before you do this, however, you want your total field to start with zeros, not with the total it had for the previous group. Blank After will reset the total field to zero after it is printed.

If the field is to be used for output more than once (punching and printing), be sure the *B* is entered on the last output line for that field. Otherwise, the field is blanked out before all required output is finished.

If a field name specified with Blank After is a table name, the element of the table looked up last will be blanked or zeroed.

COLUMNS 40-43 (END POSITION IN OUTPUT RECORD)

Disk, Punched Cards and Printed Reports

Use columns 40-43 to indicate the location on the output record of the field or constant that is to be written. You enter only the number of the punching or printing position of the rightmost character in the field or constant.

The largest number to be used to indicate end position for disk output is 9999. The largest number for printer output depends upon the number of print positions on the printer.

When *PLACE is specified for the printer (see *Columns 33-37* in this chapter), end position indicates the end position of the last field of the group that is to be printed. Thus you must be sure you have indicated an end position that allows enough room for all specified fields to be printed.

Be sure to allow enough space (as indicated by end position entries) on your output record to hold edited fields. If you overlap edited fields the resulting output may be unpredictable.

Printing on Cards (MFCM)

The MFCM prints and punches fields and constants on a card by using *PRINT in columns 32-37. Data punched in columns 1-64 is printed in positions 1-64 by print head 1. Data punched in columns 65-80 is printed in positions 49-64 by print head 2.

If you wish to print in other positions than those provided by *PRINT you must specify the following:

1. Name the field in columns 32-37.
2. Specify a print head number (1-6) in column 41.
3. Specify a print end position (01-64) in columns 42 and 43. (The leading zero in column 42 is mandatory.)

Printing on Cards (MFCU)

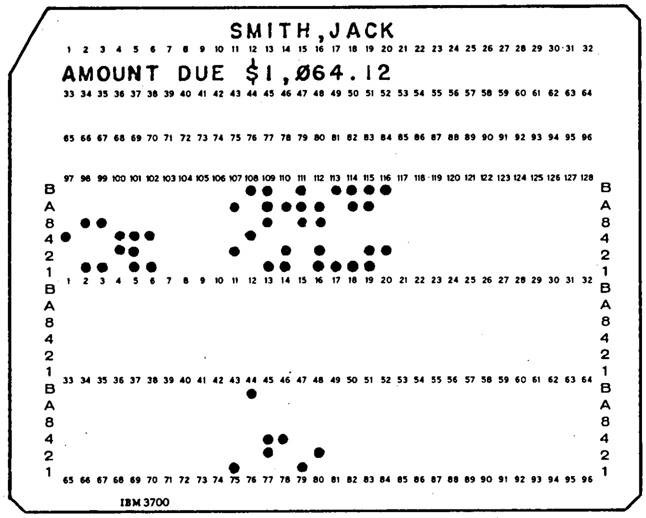
The MFCU prints and punches fields and constants in the same positions on a card by using *PRINT in columns 32-37. If you want to print fields in positions *other* than those which correspond to the punch positions of the fields, you must:

1. Name the field in columns 32-37.
2. Place an * in column 40.
3. Specify an end position for that field in columns 41-43. The maximum entry for an end position is 128.

The field will be printed in the upper portion of the card in the position you have specified.

All lines with an * in column 40 should follow all lines specifying punching only and all *PRINT lines for that record (see *Example*). All the punching for a card is done before the printing.

Note: If Blank After (column 39) is specified for a field to be punched and printed, the *B* entry must be entered on the last line specifying printing for that field. All the printing is done for a card after all the punching, so be careful not to blank out a punch field and then try to print it later. If *PRINT is the last line specifying printing for a field, the *B* entry is made in the last punching specification line for that field. If an * is used in column 40 to print a field after it is punched, the *B* entry is made in the last print specification line for that field. A Blank After entry is correctly entered for a punch and print field in Figure 172.



RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 U/M 050* Printed in U.S.A.

Program | Programmer | Date | Punching Instruction: Graphic, Punch | Card Electro Number | Page 1 2 of | Program Identification 75 76 77 78 79 80

Line	Form Type	Filename	Type (H/D/T/E)	Output Indicators		Field Name	Edit Codes / B/A/C/1-9/R	End Position in Output Record	Constant or Edit Word
01	O	CARDOUT	D	Ø1	Ø3				
02	O					NAME		2Ø	
03	O					*PRINT			
04	O					ACCTNO		6	
05	O					AMTDUE		8Ø	
06	O					AMTDUE	B*	52	' , $0. '
07	O						*	42	'AMOUNT DUE'
08	O								
09	O								

Edit code table: Commas / Zero Balances to Print / No Sign / CR / – : Yes Yes 1 A J; Yes No 2 B K; No Yes 3 C L; No No 4 D M. X = Remove Plus Sign; Y = Date Field Edit; Z = Zero Suppress.

Figure 172. Printing on the MFCU

Example

Figure 172 shows several examples of printing on a card. The coding shows that the name field will be punched and printed in the same card columns. The account number field is punched only. The amount due field is punched in columns 75-80, but for ease of reading it is printed with an edit word in columns 44-52. For the same reason, a constant is printed to identify the amount due field.

In line 06, the field AMTDUE is blanked out after it is printed by a B entry in column 39. If the B entry appeared in column 39 of line 05, the field would be blanked out after punching and would not be available for printing.

COLUMN 44 (PACKED OR BINARY FIELD)

Entry	*Explanation*
Blank	Output field in unpacked numeric or alphameric format
P	Output field in packed decimal format
B	Output field in binary format.

Column 44 must have an entry if a numeric field (decimal number) is to be written in packed decimal or binary format. Packed decimal and binary fields should not be printed.

After decimal fields have been processed, they may be left in the unpacked format. However, for more efficient use of disk, tape, or 80-column card space, decimal fields can be converted into packed decimal or binary format. Fields of four or less bytes are converted to two bytes of binary data for output; fields from five to nine bytes are converted to four bytes of binary data for output. The output device for packed decimal or binary fields can be disk, tape, 1442, MFCM or 3741 directly attached. See Column 43 in Chapter 7 for related information pertaining to input packed and binary fields.

You cannot specify packed or binary output to the following files:

- MFCU files
- CRT/keyboard files
- CONSOLE files
- BSCA files

COLUMNS 45-70 (CONSTANT OR EDIT WORD)

Use columns 45-70 to specify a constant or an edit word.

Constant

A constant is any unchanging information that is entered by a specification. Constants are usually words used for report headings, column headings or card identification. To print a constant on a card, an * must be entered in column 40 (see *Columns 40-43* in this chapter for printing on cards).

The following rules apply to constants (refer to Figure 173 for examples):

1. Field name (columns 32-37) must be blank.
2. A constant must be enclosed in apostrophes. Enter the leading apostrophe in column 45.
3. An apostrophe in a constant must be represented by two apostrophes. For example, if *George's* appears as a constant it must be coded GEORGE''S.

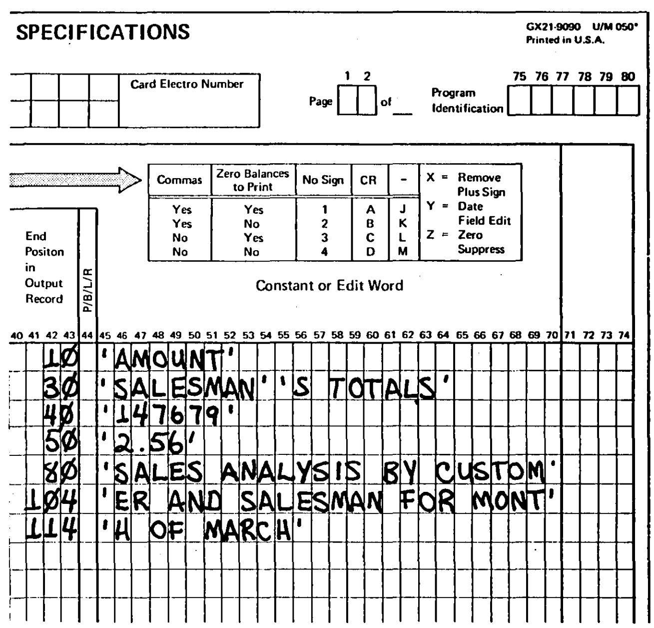


Figure 173. Examples of Output Constants

4. Up to 24 characters of constant information can be placed in one line. Additional lines may be used, but each line must be treated as a separate line of constants. The end position of each line must appear in columns 40-43.

Edit Word

An edit word gives you more flexibility in punctuating a numeric field than an edit code. You directly specify whether commas, decimal points, and zero suppression are needed, whether the negative sign should print, whether the output is dollars and cents, and whether you want a dollar sign and leading asterisks. Constants can be used within edit words (see *Examples of Edit Words* in the following test).

The following rules apply to edit words:

1. Column 38 (Edit Codes) must not be used.

2. Columns 32-37 (Field Name) must contain the name of a numeric field.

3. Columns 40-43 (End Position in Output Record) must contain an entry.

4. An edit word must be enclosed in apostrophes. Enter leading apostrophe in column 45. The edit word itself must begin in column 46.

5. Any printable character is valid, but certain characters in certain positions have special uses (see *Editing Considerations* in the following text).

6. An edit word cannot be longer than 24 characters.

7. The number of replaceable characters in the edit word must be equal to the length of the field to be edited. See *Editing Considerations* in the following text for a discussion of replaceable characters.

8. All leading zeros are suppressed unless a zero or asterisk is specified in the edit word. The zero or asterisk indicates the last leading zero in the field to be replaced by a blank or asterisk.

9. Any zeros or asterisks following the leftmost zero or asterisk are treated as constants (they are not replaceable characters).

10. Any constant to the left of the zero suppression stop character (except $) will be suppressed unless a significant digit precedes the constant.

Editing Considerations

Always leave exactly enough room on the output file for the edited field. If the field to be edited is seven characters long on the input record, make sure seven positions allows enough space for it to be written on the output file. By the time the field is edited, it may contain many more characters than seven.

When computing the length of an edited output field, determine how many of the editing characters are replaceable. The number of replaceable characters in the edit word must be equal to the length of the field to be edited (see following *Note*). The replaceable characters are:

Character	*Use*
b	Blank.
*	Asterisk fill.
0	Terminate zero suppression.

A blank is always a replaceable character, but only the first * or 0 may be a replaceable character (see rule 9 above). Fixed or floating dollar signs, decimal points, commas, ampersands (representing blanks), negative signs (CR or -), and constant information are not replaceable characters.

Note: There is an exception to the rule that the number of replaceable characters in a word must be equal to the number of characters to be edited in the unedited field. If the first character of the edit word is a zero, it is possible to have an additional replaceable character in the edit word. In this case, the field will not be zero suppressed, but all other specified editing will be performed. For example:

	Unedited Field	Edit Word	Edited Field	Unedited Field Length	Replaceable Characters in Edit Word	Edited Field Length
A	012345	0ƀƀƀƀƀ	ƀ12345	6	6	6
B	012345	0ƀƀƀƀƀƀ	ƀ012345	6	7	7
C	012345	00ƀƀƀƀƀƀ	ƀ0012345	6	7	8

In case A, the edit word contains six replaceable characters, including the leading zero. The edited field is six positions long.

In case B, the edit word contains seven replaceable characters. The leading zero is retained but the word occupies a space in the edited field which is seven positions long.

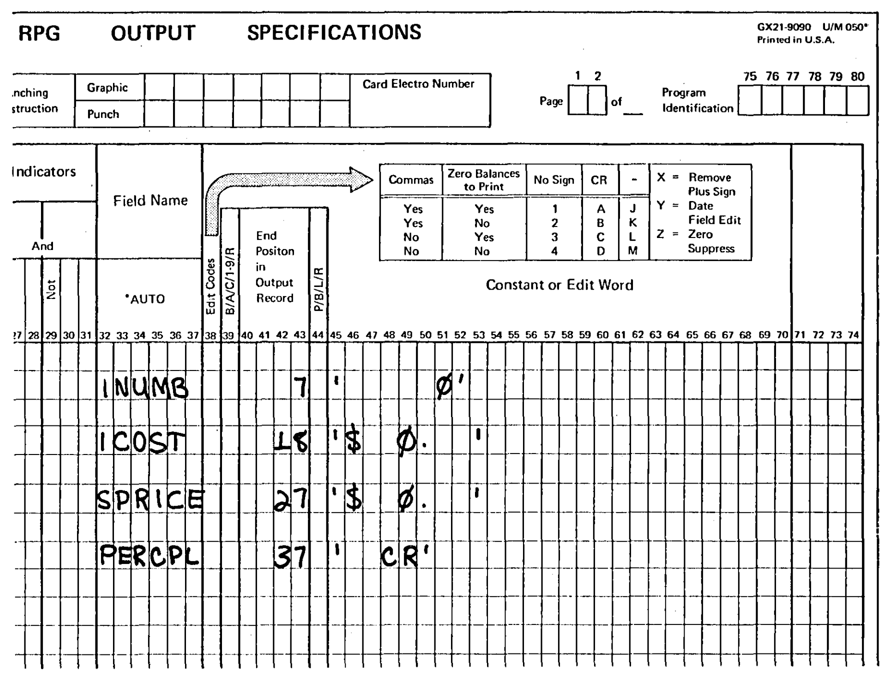
In case C, the edit word contains seven replaceable characters. The second zero is a constant that will always appear in the edited field as a leading zero. The edited field is eight positions long.

If it is necessary to show a negative number, a sign must be included in the edit word. You may use either the minus sign (–) or the letters CR. These print only for a negative number; however, the character positions they require must be taken into consideration when entering the end position of the field on the Output Sheet. Figure 174 shows that for the field PERCPL, CR is to be printed for a negative balance. Assume the field PERCPL contains the negative data 2N(-25%). The printed output would be 25CR.

Unedited Data

Item number - 000241
Item cost - 02000
Selling price - 02200
% profit or loss - 25

Output Sheet



Edited Output

241 $ 20.00 $ 22.00 25

Figure 174. Using the Output-Format Sheet to Format Data

If PERCPL was positive, CR would not print and the same field would appear as 25.

You may also use a minus sign to indicate a negative balance. If you want to leave a space between the number and the negative sign, place an ampersand (&) in the edit word before the minus sign. PERCPL would then print as 25ƀ–.

If you wish to have a dollar sign printed, you also indicate this in your edit word. To print a dollar sign at the left of the field called SPRICE, put the dollar sign ($) next to the first quote mark, then put in the necessary blanks and punctuation. A dollar sign in this position is called a fixed dollar sign. The SPRICE field in Figure 175, line A can look like any of the following (N stands for any number):

$NNN.NN

$ NN.NN

$ N.NN

$ .NN

Suppose, however, you do not want a lot of empty space between the dollar sign and the first digit when zero suppression occurs. (This is commonly the case when writing checks.) You may fill in this empty space with asterisks (*). Instead of using 0 to indicate zero suppression, you use the asterisk to indicate that all extra spaces should be filled with asterisks. The SPRICE field in Figure 175, line B can look like any of the following (N stands for any number):

$NNN.NN

$*NN.NN

$* *N.NN

$* * *.NN

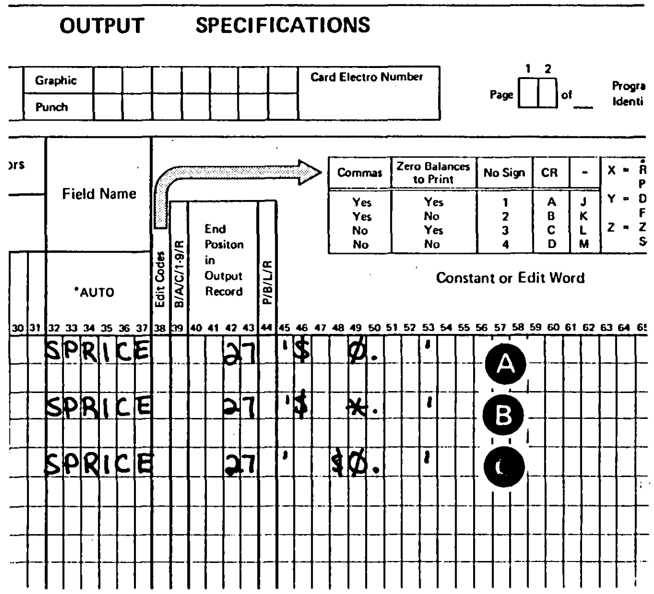


Figure 175. Different Edit Words Used on the Same Field

You may always want the dollar sign to be next to the leftmost digit instead of filling in the space with asterisks or leaving extra blanks. This is indicated in the edit word by placing the $ next to the zero suppress 0. A dollar sign which changes positions depending upon the number of positions zero suppressed is known as a floating dollar sign. When printed, the SPRICE field in Figure 175, line C can look like any of the following:

$NNN.NN

$NN.NN

$N.NN

$.NN

Note that an extra space must be left in the edit word for the floating dollar sign. This ensures a print position for the dollar sign if the output field is full.

Examples of Edit Words

Figure 176 shows examples of edit words. All examples assume that column 38 is blank. In an attempt to avoid confusion about the number of blank positions in an edited data field, the symbol ƀ is used to indicate where blank spaces appear. Zeros have not been slashed where no confusion with the letter *O* is likely to result.

Examples labeled A-H are sample edit words for some of the most frequently desired output formats. The numbered examples (1-53) that follow this first group are intended to show possible ways of handling many of the editing situations with which you might be faced.

The letters and numbers under the heading *Example Number* in Figure 176 refer to the letters and numbers in the following text:

A. Normal method of editing an amount field. Decimal point appears between dollars and cents; commas offset every three positions in the dollar portion of the field. The symbol CR appears in the edited data field when the data is negative; otherwise, it is replaced by blanks.

Since zero suppression occurs through the unit-dollar position (zero in the edit word just left of the decimal point), blanks replace leading zeros and constants until a significant digit is encountered or through the specified zero. Thus, the decimal point and data to its right always appear in the edited data. Notice that, since zero suppression occurs through the position of the zero in the edit word, zero is replaced by a blank when no significant digit appears in the data field.

B. Normal method of punctuating a quantity field. Leading zeros and constants are replaced by blanks through the position of the zero suppression zero (the next-to-last position in the edit word). Thus, if the entire data field is zero, a zero appears only in the low-order position of the edited data. A minus sign appears in the edited data if the field is negative; if not, the minus sign is replaced by a blank. The constant ON HAND always appears in the edited data as it is specified in the edit word regardless of whether the minus sign appears as specified or as a blank.

C. Normal editing of an amount field. Because the zero suppression zero appears in the ten-dollar position of the edit word, leading zeros and constants are retained starting with the unit-dollars position. Because the dollar sign is placed just left of the zero suppression zero, it becomes a floating dollar sign. In an edited data field, the floating dollar sign always appears to the immediate left of the first digit. Notice that an extra position is allowed in the high-order portion of the edit word to accommodate the floating dollar sign. The minus sign appears as a constant since a zero is specified to the left of it.

D. Similar to example C, except that zero suppression is allowed up to the decimal point, CR is used to indicate a negative value, and two asterisks are printed at the end of the edited data. In the edited data shown, the dollar sign has floated to the left to precede the first significant digit. If the unedited data were all zeros, it would appear in the output record as \$.00ƀƀ**. Note, again, the extra position in the leftmost portion of the edit word to allow for the dollar sign.

E. Similar to example D, except that no symbol is used to indicate a negative value and the edit word includes a fixed dollar sign. Because the dollar sign is placed in the extreme left position of the edit word, it is a fixed dollar sign. The fixed dollar sign always appears in the leftmost position of the edited data field.

F. This example shows that a space can be left in the edited data field between a fixed dollar sign and the first digit, even when the entire field contains significant digits. An ampersand (&) in an edit word becomes a blank in the edited field. The minus sign appears in the edited data if the field is negative. The constant GROSS always appears in the edited data.

G. By not specifying a zero or asterisk, zero suppression can occur throughout the field; thus, edited data begins with the first significant digit.

H. This example shows the use of asterisk fill. Asterisks replace all positions in the edit word to the left of the first significant digit. If the asterisk were in the rightmost position of the edit word, the entire edited field would contain asterisks when the data was all zero.

EDIT WORD (columns 45–70)	EXAMPLE NUMBER	SOURCE DATA	APPEARS IN OUTPUT RECORD AS:
`'  ,   ,  Ø.  &CR'`	A	0000000005 –	ƀƀƀƀƀƀƀƀƀƀ.05 ƀCR
`'  ,   , Ø -&ON&HAND'`	B	00000000	ƀƀƀƀƀƀƀƀƀ0 ƀ ƀONƀHAND
`'   ,   ,$Ø .  -*'`	C	0000000005 +	ƀƀƀƀƀƀƀƀƀ$0.05 ƀ *
`'   ,   , $Ø.  CR**'`	D	0034567890 –	ƀƀƀ$345,678.90 CR **
`'$  ,   ,  Ø.  '`	E	0000000000	$ƀƀƀƀƀƀƀƀƀƀ.00
`'$&  ,   , Ø  .  &-&GROSS'`	F	1234567890 –	$ƀ12,345,678,90 ƀ – ƀGROSS
`'   ,   ,   .  -'`	G	00000000123 –	ƀƀƀƀƀƀƀƀƀƀ1.23 –
`'  ,   , * .  &-'`	H	0000135792	*****1,357.92 ƀƀ
	1	0000135678	0000135678
	2	0000135678 +	0000135678
	3	0000135678 –	0000135670
`'          '`	4	0000000000	ƀƀƀƀƀƀƀƀƀƀ
`'          '`	5	0000135678 +	ƀƀƀƀ135678
`'          '`	6	0000135678 –	ƀƀƀƀ135678
`'         Ø'`	7	0000135678 –	ƀƀƀƀ135678
`'Ø         '`	8	0000135678 +	ƀ000135678
`'          &CR&NET'`	9	0000135678 +	ƀƀƀƀ135678 ƀƀƀ ƀNET
`'          &CR&NET'`	10	0000135678 –	ƀƀƀƀ135678 ƀCR ƀNET
`'          &-&&NET'`	11	0000135678 –	ƀƀƀƀ135678 ƀ – ƀƀNET
`'          &NET&CR'`	12	0000135678	ƀƀƀƀ135678ƀNETƀƀƀ
`'          &NET&CR'`	13	0000135678 –	ƀƀƀƀ135678 ƀNETƀCR
`'          &&PROFIT'`	14	0000135678	ƀƀƀƀ135678 ƀƀPROFIT
`'$          &-&NET'`	15	0000135678 +	$ƀƀƀƀ135678 ƀƀ ƀNET
`'$          &-&NET'`	16	0000135678 –	$ƀƀƀƀ135678 ƀ – ƀNET
`'$Ø         -&NET'`	17	0000135678	$0000135678 ƀ ƀNET
`'       $Ø  &CR'`	18	0000135678 –	ƀƀƀƀ$135678 ƀCR
`'       $Ø  &CR'`	19	1234567809 –	$1234567809 ƀCR
`'$       Ø  &CR&GROSS'`	20	0000000000	$ƀƀƀƀƀƀƀƀ00 ƀƀƀ ƀGROSS
`'*         '`	21	0000135678 –	*000135678
`'*         '`	22	1234567890 +	1234567890
`'         *'`	23	0000135678 –	****135678
`'  ,   ,   .  &CR&&NET'`	24	0000135678 –	ƀƀƀƀƀ1,356,78 ƀCR ƀƀNET
`'  ,   ,   .  &CR&-NET'`	25	0000135678	ƀƀƀƀƀ1,356,78 ƀƀƀ ƀ–NET
`'   ,   ,$Ø .  &NET'`	26	0000000005	ƀƀƀƀƀƀƀƀƀ$0,05 ƀNET
`'   ,   , $Ø.  '`	27	0000000005	ƀƀƀƀƀƀƀƀƀƀ$.05
`'   ,   , $Ø.  -'`	28	1234567890 –	$12,345,678.90 –
`'   ,   , $Ø.  CR'`	29	0001356789 –	ƀƀƀƀ$13,567.89 CR
`'  ,   ,  *.  *CR**'`	30	0000135678 +	*****1,356.78 ƀƀƀ **
`'   , $Ø,   .  -SALES'`	31	0000001234	ƀƀƀƀƀƀ$,012.34 ƀ SALES
`'$&  ,   ,  Ø.  CR'`	32	1234567890 –	$ƀ12,345,678.90 CR
`'  ,   ,   ,   -OLD&BALNCE'`	33	1234567890 –	1,234,567,890 – OLDƀBALNCE
`'  ,   ,   , Ø -OLD&BALNCE'`	34	0000000000	ƀƀƀƀƀƀƀƀƀƀƀƀƀ0 ƀ OLDƀBALNCE
`'  ,   ,   DOLLARS  CENTS'`	35	0000135678	ƀƀƀƀƀ1,356DOLLARS78 CENTS
`'  ,   DOLLARS  CENTS'`	36	000000	ƀƀƀƀƀƀƀƀƀƀƀƀƀƀ CENTS
`'  , Ø DOLLARS  CENTS&CR'`	37	000000	ƀƀƀƀ0DOLLARS00 ƀƀƀƀƀƀƀƀ
`'  Ø LBS.&  OZ.TARE&-'`	38	000002 +	ƀƀƀ0LBS.ƀ02 ƀƀƀƀƀƀƀƀƀ
`'   ØLBS.  OZ.TARE&-'`	39	000002 –	ƀƀƀƀLBS.02 OZ.TAREƀ–
`'Ø  -  -    '`	40	095140036	ƀ95–14–0036
`'Ø HRS.  MINS.&O''CLOCK'`	41	0042	ƀ0HRS.42 MINS.ƀO'CLOCK

Figure 176 (Part 1 of 2). Examples of Edit Words

Edit Word	Line	Source Data	Output
`'  ,  Ø.  '`	42	000000	b̸b̸b̸b̸b̸.00
`'  ,   .Ø '`	43	000000	b̸b̸b̸b̸b̸b̸b̸0
`'  ,  $,Ø  .  '`	44	00123456	b̸b̸b̸1$,234.56
`'   ,   , Ø * '`	45	0000000000	b̸b̸b̸b̸b̸b̸b̸b̸b̸b̸0*00
`'Ø,   ,Ø  '`	46	001234	b̸,012,034
`'   ,  *,   *   '`	47	0000001234	******,012*34
`'& ,* Ø,   '`	48	013579	***130,579
`'   -  -  &LATER'`	49	093066	b̸9–30–66 b̸LATER
`'   &  &  &LATER'`	50	093066	b̸9b̸30b̸66 b̸LATER
`'   /  /  '`	51	100166	10/01/66
`'  ,   ,   .  - '`	52	000000015 –	b̸b̸b̸b̸b̸b̸b̸b̸b̸b̸15 –
`'  ,   , Ø .  - '`	53	000000005	b̸b̸b̸b̸b̸b̸b̸b̸0.05 b̸

Figure 176 (Part 2 of 2). Examples of Edit Words

1. No edit word. The data in the output record has the same format as the unedited data. Notice that the low-order position of the output field is printed as an alphabetic character (J-R) if the source data field is negative.

2. Same as 1.

3. Same as 1.

4. A blank edit word. All leading zeros are blanked and any sign in the low-order position of the unedited field is removed when the data is edited. Negative values are not identified.

5. Same as 4.

6. Same as 4.

7. The effect is the same as shown in examples 4, 5, and 6.

8. Although the zero suppression zero appears in the high-order position of the edit word, suppression of the first leading zero cannot be avoided. See *Note* in *Editing Considerations* in this section for a discussion of an exception.

9. An ampersand appears as a blank in the edited data. The symbol CR appears in the edited data if the field is negative. It is replaced by blanks if the field is positive. The constant NET always appears in the edited data field.

10. Same as 9.

11. An ampersand appears as a blank in the edited data. A minus sign, instead of CR, indicates negative values.

12. NET CR indicates when the edited data field is negative. Therefore, when the edited field is positive, NET CR appears as blanks.

13. Same as 12.

14. The constant PROFIT appears in the edited data field. Negative values are not identified.

15. Similar to example 11, except that a fixed dollar sign is shown. An extra position is added to the edit word to allow for the dollar sign.

16. Same as 15.

17. Although the dollar sign appears to the immediate left of the zero suppression zero, it is a fixed dollar sign because it appears in the leftmost position of the edit word.

18. The floating dollar sign is shown for different numbers of leading zeros. Note the extra position in the high-order portion of the edit word to allow for the dollar sign.

19. Same as 18.

20. This example shows how some zeros can appear in the edited field when the entire field is zero. Zero suppression occurs through the position of the 0 in the edit word. This leaves two positions in which zeros can appear in the edited field.

21. This example shows asterisk protection and zero suppression for a single position. Note that the asterisk is replaced by a significant digit in the position. Negative values are not identified.

22. Same as 21.

23. Asterisk protection and zero suppression for an entire field. Asterisks are replaced by significant digits.

24. A method of editing an amount field. Punctuation and zeros to the left of the first significant digit are blanked. The decimal point is also lost when there are fewer than three significant digits. The constants NET or -NET always appear in the edited field.

25. Same as 24.

26. Standard method for placing the floating dollar sign so that at least the decimal point is retained regardless of the number of leading zeros. The extra position appears in the leftmost position of the edit word to compensate for the floating dollar sign.

27. Same as 26.

28. Same as 26.

29. Same as 26.

30. Asterisk protection and zero suppression to the decimal point. The decimal point is retained regardless of the number of leading zeros. Note that asterisks replace punctuation when leading zeros are suppressed. The second asterisk appears only when the edited data field is negative; the third and fourth asterisks always appear in the edited field.

31. This example shows that a constant (in this case, a comma) follows the dollar sign in the edited data if the floating dollar sign and the zero suppression zero immediately precede a constant. This applies if there are a number of leading zeros. In the case of a comma, this looks awkward; in the case of a decimal point it is a normal approach (see example 27).

32. This example shows how to insert a space between a fixed dollar sign and the first data digit when all digits in the field are significant. An ampersand in an edit word appears as a space in the edited data field.

33. Normal punctuation of a quantity field. In this example, all leading zeros, including the units position, are suppressed (compare with example 34).

34. Normal method of showing a single zero in the edited data field when the data field contains only zeros.

35. Constants in the edit word are handled the same as punctuation marks; that is, only constants to the right of the first significant digit or the zero suppression zero appear in the edited data. Examples 37-38 show how more edit word constants, other than the CR or minus, can be blanked on a positive field. Examples 37-39 also show the effect that the position of the zero suppression zero has on constants. In example 38, an ampersand placed after the first constant provides a space following that constant in the edited data.

36. See example 35.

37. See example 35.

38. See example 35.

39. See example 35.

40. Possible method for editing a social security number field. A hyphen (-) is used within the edit word. In the example shown, the initial zero is suppressed. However, if you want the initial zero to appear in the edited data, you must leave an extra position in the edit word. See the note under *Editing Considerations* for a discussion of this exception.

41. This example shows the use of constants in the edit word. In this example, the constant contains an apostrophe.

42. This example shows the effect that the position of the zero suppression zero has on the decimal point (or any other constants) and following zeros.

43. Same as 42.

44. This example shows that a dollar sign separated from the zero suppression zero, even if only by a comma, is a constant rather than a floating dollar sign.

45. Any zero or asterisk to the right of the high-order zero or asterisk is a constant, not a zero suppression zero or asterisk-protection symbol. Examples 47 and 48 also show that asterisk protection replaces not only blanks, but also other constants to the left of the first significant digit.

46. Same as 45.

47. Same as 45.

48. Same as 45.

49. An example of editing a date field. Since month numbers have at most one leading zero, it is not necessary to specify a zero suppression zero. Example 50 shows the use of an ampersand to retain a blank space in the edited data.

50. Same as 49.

51. Same as 49.

52. This example shows what happens to the decimal point when no zero suppression zero is specified for a field which has fewer than three significant digits.

53. This example shows how to retain the decimal point in a data field which has fewer than three significant digits.

COLUMNS 71-74

Columns 71-74 are not used.

COLUMNS 75-80 (PROGRAM IDENTIFICATION)

See Chapter 2.

Appendix A. Running an RPG II Program—Halts and OCL

RPG II HALT PROCEDURES

Figure A-1 is a list of error conditions resulting in a halt during execution or compilation of an RPG II program. Options available to the operator following each halt are also given. The options are:

0 – Continue: Control is returned to the program, and processing continues.

1 – Bypass: The remainder of the program cycle is bypassed, and the next record is read.

2 – Controlled Cancel: End-of-job operations (specified by a LR indicator in your program) are done, tables are dumped, and files are closed.

3 – Immediate Cancel: The job is cancelled without returning control to the RPG II program.

For Model 15 severity codes are assigned to halts. A NOHALT OCL statement may be used to allow the user to indicate the level of severity for options. See the *IBM System/3 Model 15 System Control Programming Reference Manual*, GC21-5077, for more information.

Resetting Halts (Models 10 and 12)

In order to select an option, the operator dials its corresponding number on the rightmost address/data switch and presses console START. (He presses HALT/RESET if the system has the Dual Program Feature.) A complete discussion of operator procedures appears in the appropriate operator's guide listed under *Related Publications* in the Preface.

Resetting Halts (Model 15)

In order to select an option, the operator presses the PF12 key on the keyboard. This positions the cursor to the last halt. The operator selects the option desired by entering 0, 1, 2, or 3 via the keyboard, and presses ENTER to continue. A complete discussion of operator procedures appears in the appropriate operator's guide listed under *Related Publications* in the Preface.

Appendix F contains a detailed list of compilation errors.

OPERATION CONTROL LANGUAGE FOR RPG II

In order to compile an RPG II source program, the RPG II compiler program must be loaded into main storage. This can be done by including an IBM-supplied procedure named RPG (located in the Source Library) in the job stream. The OCL statements that include the library procedure are:

```
/&
// CALL RPG, R1
// RUN
```

The OCL statements included in the Source Library procedure named RPG are shown in Figure A-2.

Note: This example assumes the system is on R1.

Library procedures can be modified. OCL statements necessary to modify a library procedure are described in the appropriate SCP reference manual listed under *Related Publications* in the Preface.

Halt Display	Error Description	Operator Options				Model 15 only	
		0-Continue	1-Bypass	2-Controlled Cancel[2]	3-Immediate Cancel	Severity Code	Default Option
H1	Indicator H1 is on	X		X	X	4	none
H2	Indicator H2 is on	X		X	X	4	none
H3	Indicator H3 is on	X		X	X	4	none
H4	Indicator H4 is on	X		X	X	4	none
H5	Indicator H5 is on	X		X	X	4	none
H6	Indicator H6 is on	X		X	X	4	none
H7	Indicator H7 is on	X		X	X	4	none
H8	Indicator H8 is on	X		X	X	4	none
H9	Indicator H9 is on	X		X	X	4	none
H0	All halt indicators have been displayed	X		X	X	4	none
11	Square root of negative number asked	X		X	X	4	2
12	Overflow during divide	X		X	X	4	2
13	Division by zero attempted	X		X	X	4	2
14	Zero, negative, or invalid array index	X		X	X	4	2
15	Table out of sequence	X		X	X	4	2
16	No table data found	X		X	X	4	2
17	Too much data for table	X		X	X	4	2
18	Terminal errors in RPG source				X	2	3
19	Warning errors in RPG source	X			X	2	0

Figure A-1 (Part 1 of 3). RPG II Halts and Operator Options

Halt Display	Error Description	Operator Options				Model 15 only	
		0-Continue	1-Bypass	2-Controlled Cancel[2]	3-Immediate Cancel	Severity Code	Default Option
10	No Primary or Secondary files opened				X	2	3
1A	Exceeded specified object core or insufficient core to compile				X	2	3
1C	Invalid call to RPG Halt routine			X	X	4	2
1E	End-of-file on demand file	X		X	X	4	2
1F	Attempting to access beyond extent			X	X	4	2
1H	Attempting to add duplicate key	X		X	X	4	2
1J	Attempting to add key in wrong order	X		X	X	4	2
1L	Key modified by record update or invalid record update operation	X		X	X	4	2
1P	1P forms alignment	X	X			1	none
1U	Record not found. Key not in index or record number too large.		X	X	X	4	2
1Y	Invalid response to display		X	X	X	4	2
1b̸	Prepare for table output	X			X	4	none
J0-J9[1]	Record out of sequence		X	X	X	4	2
L0-L9[1]	File out of matching record sequence		X	X	X	4	2

Figure A-1 (Part 2 of 3). RPG II Halts and Operator Options

Halt Display	Error Description	Operator Options				Model 15 only	
		0-Continue	1-Bypass	2-Controlled Cancel[2]	3-Immediate Cancel	Severity Code	Default Option
U0-U9[1]	Unidentified Record		X	X	X	4	2
J0 (Model 15)	Multiple output to MFCM combined file	X		X	X	4	2
U0 (Model 15)	RPG compiler error				X	8	3
-b̸	Key too low or high for volume on line		X	X	X	4	2
–'	High key missing for volume on line		X	X	X	4	2
'8	Terminal errors during auto report				X	2	3
Y1	Non-ASCII characters in BSCA record			X	X	4	2
Y3	Invalid request to BSCA			X	X	4	2
Y4	Connection lost on BSCA file			X	X	4	2
Y5	Permanent error on BSCA			X	X	4	2
YA	Two consecutive conversational replies made to BSCA file			X	X	4	2

[1] On Model 15, halts J2-J9, U3-U9, L0 and L2-L9 are not issued. Instead, halts J1, U1, U2, and L1, when issued, include the file number.

| [2] If a halt occurs after the close routine is entered, the 2 option is no longer a valid operator option. The 3 option becomes the default.

Figure A-1 (Part 3 of 3). RPG II Halts and Operator Options

```
// LOAD $RPG,R1

// FILE NAME-$SOURCE,UNIT-R1, RETAIN-S,TRACKS-10,PACK-SYSTEM¹

// FILE NAME-$WORK,UNIT-R1,RETAIN-S,TRACKS-10,PACK-SYSTEM¹

// RUN
```

| [1] For Model 15 Systems, $SOURCE and $WORK are also supported on the 5445 or 3340 disk.

Figure A-2. IBM–Supplied Library Procedure for Compiling an RPG II Source Program

Appendix B. RPG II Sample Programs

This appendix contains two complete RPG II sample programs, SAMPL1 and SAMPL2, which should be run immediately after system generation. If these programs execute successfully, your system has been generated properly. The following information is included for each of these sample programs:

- Detailed description of the RPG II specifications (see *Sample Program 1* and *Sample Program 2*).
- Procedures for loading, compiling, and executing the sample programs on System/3 Models 8, 10, and 12, complete with the compilation listing and execution output (see *System/3 Models 8, 10, and 12 RPG II Sample Programs*).
- Procedures for loading, compiling, and executing the sample programs on System/3 Model 15, complete with the compilation listing and execution output (see *System/3 Model 15 RPG II Sample Programs*).

In addition to the sample programs SAMPL1 and SAMPL2, this appendix contains the RPG II specifications for and explanations of three additional RPG II example programs (see *Example Programs*).

SAMPLE PROGRAM 1

SAMPL1 loads 100 records into an indexed disk file. The records are created in calculations by means of a program loop. SAMPL1 should be followed by SAMPL2, which prints out the indexed file, verifying that it was properly loaded. Figure B-1 shows the completed specifications sheets for SAMPL1 for Models 10 and 12. Figure B-2 shows the specifications for Model 15.

Control Card Specifications

This card must be present in every job. It is the first card in the source deck.

File Description Specifications

These specifications (Figure B-1 or B-2) describe the files used in the program. The indexed output file, DISKOUT, will consist of 128-position records with a 6-position key field starting in the first record position. DISKOUT is a single volume file (01 in columns 68-69). A printer output file with a record length of 96 is also defined on the File Description Sheet.

Input Specifications

The single input file must be further described on the Input Specifications Sheet (Figure B-1 or B-2).

Calculation Specifications

The indexed file is loaded by means of a loop in calculations as follows:

- *line 01:* The result field, COUNT, is set to zero.
- *line 02:* The result field, RECNBR, is set to zero.
- *line 03:* REPEAT serves as a label for the loop in calculations.
- *line 04:* COUNT is incremented by five.
- *line 05:* RECNBR is incremented by one.
- *line 06:* If COUNT compares equal to 505, indicator 02 turns on.
- *line 07:* If COUNT is not equal to 505, the line on the Output Sheet (see the Output Sheet in Figure B-1 or B-2) which is identified by an *E* in column 15 is written on disk. Thus, COUNT becomes the output key field and RECNBR becomes a 3-position output field containing the record number.
- *line 08:* The program loops back to the REPEAT label. The calculations in lines 4-7 are repeated until COUNT compares equal to 505 (100 records have been written on the indexed file).
- *line 09:* When 100 records have been written, the LR indicator turns on.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 Printed in U.S.A.

Program: SAMPLE PROGRAM #1	Date	Punching Instruction	Graphic / Punch	Card Electro Number
Programmer				

Page 01 of __ Program Identification (75–80): SAMPL1

Control Card Specifications

Refer to the specific System Reference Library manual for actual entries.

Line (3–5)	Form Type (6)	Core Size to Compile (7–9)	Object Output (10)	Listing Options (11)	Core Size to Execute (12–14)
01	H				008

File Description Specification

```
Line  Form
02    F ************************************************
03    F *                                              *
04    F * THIS PROGRAM -                               *
05    F *                                              *
06    F *      1. LOADS 100 RECORDS TO AN INDEXED FILE. *
07    F *                                              *
08    F *      2. READS ONE RECORD FROM FILE $SOURCE FOR *
09    F *         INPUT. THE FILE $SOURCE IS BUILT WHEN *
10    F *         SAMPLE PROGRAM SAMPL2 IS COMPILED BY  *
11    F *         GIVING A RETAIN-T PARAMETER TO THE FILE *
12    F *         $SOURCE                              *
13    F *                                              *
14    F *      3. CREATES THE OUTPUT DATA USING A       *
15    F *         LOOP IN THE CALCULATION SPECIFICATIONS. *
16    F *                                              *
17    F *      4. USES KEYS FROM 000005 THROUGH 000500  *
18    F *         IN INCREMENTS OF 5.                   *
19    F *                                              *
20    F *      5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2 *
21    F *         TO VERIFY THAT THE FILE WAS PROPERLY  *
22    F *         LOADED.                              *
23    F *                                              *
24    F ************************************************
25    F$SOURCE IP  F  96  96            DISK
26    FDISKOUT O   F 256 128 06AI     1 DISK                         01
27    FPRINTER O   F  96  96            PRINTER
      F
      F
      F
      F
```

Figure B-1 (Part 1 of 3). Specifications for SAMPL1 (Models 10 and 12)

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9094 Printed in U.S.A.

Program: SAMPLE PROGRAM #1 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page (1-2): Ø2 of ___ | Program Identification (75-80): SAMPL1

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1 (21-27)	Record Identification Codes 2 (28-34)	Record Identification Codes 3 (35-41)	Stacker Select (42)	P/B/L/R (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)	(71-74)
01	I	$SOURCE	NS			Ø1																
02	I											1	1		NODATA							
03	I																					
04	I																					
05	I																					
06	I																					

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation

Form GX21-9093 Printed in U.S.A.

Program: SAMPLE PROGRAM #1 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page (1-2): Ø3 of ___ | Program Identification (75-80): SAMPL1

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR, AN/OR) (7-8)	Indicators (9-11)	And (12-14)	And (15-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators Plus / High / 1>2 (54-55)	Minus / Low / 1<2 (56-57)	Zero / Equal / 1=2 (58-59)	Comments (60-74)
01	C		Ø1				Z-ADD	Ø	COUNT	6	Ø					
02	C		Ø1				Z-ADD	Ø	RECNBR	3	Ø					
03	C					REPEAT	TAG									
04	C		Ø1			COUNT	ADD	5	COUNT							
05	C		Ø1			RECNBR	ADD	1	RECNBR							
06	C		Ø1			COUNT	COMP	5Ø5							Ø2	
07	C		Ø1	NØ2			EXCPT									
08	C		Ø1	NØ2			GOTO	REPEAT								
09	C						SETON						LR			
10	C	LR				RECNBR	SUB	1	RECNBR							
11	C															
12	C															
13	C															
14	C															
15	C															
16	C															
17	C															
18	C															
19	C															
20	C															
	C															
	C															
	C															
	C															
	C															

Figure B-1 (Part 2 of 3). Specifications for SAMPL1 (Models 10 and 12)

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090
Printed in U.S.A.

Program: SAMPLE PROGRAM #1 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number: | Page: Ø4 of ___ | Program Identification (75–80): SAMPL1

Line (3–5)	Form Type (6)	Filename (7–14)	Type (H/D/T/E) (15)	Stacker # / Fetch (F) (16)	Space Before (17)	Space After (18)	Skip Before (19–20)	Skip After (21–22)	Output Indicators (23–31)	Field Name / *AUTO (32–37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40–43)	P/B/L/R (44)	Constant or Edit Word (45–70)
01	O	PRINTER	T			2	Ø4		LR						
02	O												2Ø		'SAMPLE PROGRAM HAS'
03	O												27		'LOADED'
04	O									RECNBR	Z		31		
05	O												39		'RECORDS'
06	O												61		'INTO AN INDEXED FILE.'
07	O		T			2			LR						
08	O												21		'KEYS ARE IN ASCENDING'
09	O												42		'SEQUENCE STARTING AT'
10	O												64		'ØØØØØ5 AND INCREASING'
11	O												84		'IN INCREMENTS OF 5.'
12	O		T					Ø1	LR						
13	O												21		'SAMPLE PROGRAM 2 WILL'
14	O												44		'PRINT FROM THE INDEXED'
15	O												65		'FILE TO SHOW THAT IT'
16	O												86		'WAS PROPERLY LOADED.'
17	O	DISKOUT	E						Ø1 NØ2						
18	O									COUNT			6		
19	O												94		'RECORD NUMBER'
20	O									RECNBR			128		
	O														
	O														
	O														
	O														
	O														

Edit code legend:

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Figure B-1 (Part 3 of 3). Specifications for SAMPL1 (Models 10 and 12)

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

IBM International Business Machine Corporation

GX21-9092
Printed in U.S.A.

Program: SAMPLE PROGRAM #1
Programmer:
Date:
Punching Instruction — Graphic: / Punch:
Card Electro Number
Page (1 2): Ø1 of __
Program Identification (75–80): SAMPL1

Control Card Specifications

Columns: Line (3–5) · Form Type (6) · Core Size to Compile (7–9) · Object Output (10) · Listing Options (11) · Core Size to Execute (12–14) · Debug (15) · MFCM Stacking Sequence (16) · Inverted Print (21) · 360/20 2501 Buffer (22) · Number Of Print Positions (23–25) · Alternate Collating Sequence (26) · Model 20: Address to Start (27–30), Work Tapes (31), Overlay Open (32), Overlap Printer (33), Binary Search (34), Tape Error (35), 2152 Checking (36), Inquiry (37) · Model 20: Read/Write/Compute (38), Keyboard Output (39) · Sign Handling (40) · 1P Forms Position (41) · Indicator Setting (42) · File Translation (43) · Punch MFCU Zeros (44) · Nonprint Characters (45) · Table Load Halt (47) · Shared I/O (48) · Field Print (49) · Formatted Core Dump (50) · RPG to RPG II Conversion (51) · Refer to the specific System Reference Library manual for actual entries (52–74).

```
01 H     ØØ8
```

File Description Specification

Columns: Line (3–5) · Form Type (6) · Filename (7–14) · File Type: I/O/U/C/D (15) · File Designation: P/S/C/R/T/D (16) · End of File: E (17) · Sequence: A/D (18) · File Format: F/V/S/M/D (19) · 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: A/P/I/K (31) · Type of File Organization or Additional Area: I/D/T or 2 (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/E/M (53) · Name of Label Exit (54–59) · Extent Exit for DAM / Core Index (60–65) · Continuation Lines: K (53), Option (54–59), Entry (60–65) · A/U (66) · File Addition/Unordered (66) · Number of Tracks for Cylinder Overflow (67–68) · Number of Extents (68–69) · Tape Rewind: R/U/N (70) · File Condition U1-U8 (71–72)

```
02 F*************************************************
03 F*                                               *
04 F*   THIS PROGRAM -                              *
05 F*                                               *
06 F*   1. LOADS 1ØØ RECORDS TO AN INDEXED FILE.    *
07 F*                                               *
08 F*   2. READS ONE RECORD FROM FILE $SOURCE FOR   *
09 F*      INPUT. THE FILE $SOURCE IS BUILT WHEN    *
10 F*      SAMPLE PROGRAM SAMPL2 IS COMPILED BY     *
11 F*      GIVING A RETAIN-T PARAMETER TO THE FILE  *
12 F*      $SOURCE                                  *
13 F*                                               *
14 F*   3. CREATES THE OUTPUT DATA USING A          *
15 F*      LOOP IN THE CALCULATION SPECIFICATIONS.  *
16 F*                                               *
17 F*   4. USES KEYS FROM ØØØØØ5 THROUGH ØØØ5ØØ     *
18 F*      IN INCREMENTS OF 5.                      *
19 F*                                               *
20 F*   5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2   *
21 F*      TO VERIFY THAT THE FILE WAS PROPERLY     *
22 F*      LOADED.                                  *
23 F*                                               *
24 F*************************************************
25 F$SOURCE IP  F 256 256            DISK
26 FDISKOUT O   F 256 128 Ø6AI     1 DISK                        Ø1
27 FPRINTER O   F  96  96            PRINTER
```

Figure B-2 (Part 1 or 3). Specifications for SAMPL1 (Model 15)

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094
Printed in U.S.A.

Program: SAMPLE PROGRAM #1
Programmer: | Date:
Punching Instruction: Graphic / Punch
Card Electro Number
Page: Ø2 of ___
Program Identification (75–80): SAMPL1

I

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (1-N) (17)	Option (O) (18)	Record Identifying Indicator or ** (19-20)	Record Identification Codes 1: Position / Not (N) / C/Z/D / Character (21-27)	2: Position / Not (N) / C/Z/D / Character (28-34)	3: Position / Not (N) / C/Z/D / Character (35-41)	Stacker Select (42)	P/B/L/R (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)	(71-74)
01	I	$SOURCE	NS			Ø1																
02	I											1	1		NODATA							
03	I																					
04	I																					
05	I																					
06	I																					

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093
Printed in U.S.A.

Program: SAMPLE PROGRAM #1
Programmer: | Date:
Punching Instruction: Graphic / Punch
Card Electro Number
Page: Ø3 of ___
Program Identification (75–80): SAMPL1

C

Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR, AN/OR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators Plus / High / 1>2 (54-55)	Minus / Low / 1<2 (56-57)	Zero / Equal / 1=2 (58-59)	Comments (60-74)
01	C	LØ			Z-ADD	Ø	COUNT	6	Ø					
02	C	LØ			Z-ADD	Ø	RECNBR	3	Ø					
03	C	LØ		REPEAT	TAG									
04	C	LØ		COUNT	ADD	5	COUNT							
05	C	LØ		RECNBR	ADD	1	RECNBR							
06	C	LØ		COUNT	COMP	5Ø5							Ø2	
07	C	LØ	NØ2		EXCPT									
08	C	LØ	NØ2		GOTO	REPEAT								
09	C	LØ			SETON						LR			
10	C	LR		RECNBR	SUB	1	RECNBR							
11	C													
12	C													
13	C													
14	C													
15	C													
16	C													
17	C													
18	C													
19	C													
20	C													
	C													
	C													
	C													
	C													
	C													

Figure B-2 (Part 2 of 3). Specifications for SAMPL1 (Model 15)

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 Printed in U.S.A.

Program: SAMPLE PROGRAM #1 — Page: 04 — Program Identification: SAMPL1

Line	Form Type	Filename	Type (H/D/T/E)	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01	O	PRINTER	T		2	04		LR				
02	O										20	'SAMPLE PROGRAM HAS'
03	O										27	'LOADED'
04	O								RECNBR	Z	31	
05	O										39	'RECORDS'
06	O										61	'INTO AN INDEXED FILE.'
07	O		T		2			LR				
08	O										21	'KEYS ARE IN ASCENDING'
09	O										42	'SEQUENCE STARTING AT'
10	O										64	'ØØØØØ5 AND INCREASING'
11	O										84	'IN INCREMENTS OF 5.'
12	O		T				Ø1	LR				
13	O										21	'SAMPLE PROGRAM 2 WILL'
14	O										44	'PRINT FROM THE INDEXED'
15	O										65	'FILE TO SHOW THAT IT'
16	O										86	'WAS PROPERLY LOADED.'
17	O	DISKOUT	E									
18	O								COUNT		6	
19	O										94	'RECORD NUMBER'
20	O								RECNBR		128	

Figure B-2 (Part 3 of 3). Specifications for SAMPL1 (Model 15)

The output files, PRINTER and DISKOUT, are described in detail on the Output Sheet. Three total output lines are printed after end-of-file has occurred on $SOURCE. The printer skips to line 04 before printing the first line and double-spaces after printing each of the first two lines. The RECNBR field, which now contains a value of 100, is inserted into the first output line in positions 29-31. After printing the last output line, the printer skips to line 01 of the following page.

The disk record to be written by exception output in calculations is also described on the Output Sheet.

SAMPLE PROGRAM 2

Sample Program 1 (SAMPL1) must be executed before Sample Program 2 (SAMPL2). SAMPL2 reads the indexed file created by SAMPL1 and prints out fields from each record read. Thus, SAMPL2 verifies that SAMPL1 loaded the indexed file properly. The program specifications for SAMPL2 for all models are shown in Figure B-3.

Control Card Specifications

This card must be present in every job. It is the first card in the source deck.

File Description Specifications

These specifications describe the files used by SAMPL2 for input and output of data.

The indexed file created by SAMPL1 is named DISKIN in this program. It is defined with an *E* in column 17 so that the program will not end until end-of-file of the disk input file. Note that a different block length is given than was specified when the file was created.

A printer file, PRINTER, is described for the printed output of SAMPL2. Since an overflow indicator is specified for the file, later operations can be conditioned on overflow (see the Output Sheet in Figure B-3).

Input Specifications

The fields of interest in DISKIN are described in detail on the Input Sheet. A character zero in position one of the input records will turn on record identifying indicator 01.

Calculation Specifications

The field named COUNT is incremented by one on each program cycle to keep a running total of the records which have been read from DISKIN and printed out on PRINTER.

Output Specifications

Three different output lines are described for the printer file, PRINTER.

The first printer line is a heading line which will be printed on line 4 of the first output page (conditioned by 1P) and each succeeding page (conditioned by OF in an OR relationship). The printer will double-space (2 in column 18) after the heading line is printed. Thus, each output page will have a heading consisting of three constant fields and a page field. Because the PAGE reserved word has been used, pages will automatically be numbered sequentially.

For each record read from DISKIN (indicator 01 is on), a detail line consisting of three fields from each input record is written. These fields are reformatted so that the output line ends in position 25.

The printer triple-spaces (3 in column 17) before the total line is printed. The total line is printed when end-of-file (LR is on) has occurred on DISKIN. The 3-position COUNT field which was incremented in calculations is followed by a statement in the total line indicating how many records were read and printed from DISKIN. If COUNT is equal to 100, SAMPL1 and SAMPL2 have executed successfully.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 Printed in U.S.A.

Program: SAMPLE PROGRAM #2
Programmer:
Date:

Page 01 of __ Program Identification (75–80): SAMPL2

Control Card Specifications

```
  01 H     008
```

File Description Specification

```
  02 F*************************************************
  03 F*                                               *
  04 F* THIS PROGRAM -                                *
  05 F*                                               *
  06 F*    1. MUST BE PRECEDED BY SAMPLE PROGRAM 1    *
  07 F*       WHICH LOADS AN INDEXED FILE.            *
  08 F*                                               *
  09 F*    2. READS AN INDEXED FILE SEQUENTIALLY.     *
  10 F*                                               *
  11 F*    3. USES A BLOCK LENGTH FOR DISK WHICH      *
  12 F*       IS DIFFERENT FROM THAT USED FOR         *
  13 F*       LOADING THE FILE IN SAMPLE PROGRAM 1.   *
  14 F*                                               *
  15 F*    4. COUNTS THE NUMBER OF RECORDS READ SO    *
  16 F*       THAT THE USER CAN QUICKLY VERIFY THAT   *
  17 F*       100 RECORDS WERE LOADED.                *
  18 F*                                               *
  19 F*************************************************
  20 FDISKIN  IPE F 512 128 06AI     1 DISK                        01
  21 FPRINTER O   F  96  96     OF     PRINTER
  22 F
```

Figure B-3 (Part 1 of 2). Specifications for SAMPL2

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 Printed in U.S.A.

Program: SAMPLE PROGRAM #2 — Page: 02 — Program Identification: SAMPL2

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Position	C/Z/D	Character	Field Location From	Field Location To	Decimal Positions	Field Name
01	I	DISKIN	NS	01	1	C	0				
02	I							1	6		KEY
03	I							82	94		DESC
04	I							126	128	0	RECNBR
05	I										
06	I										

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093 Printed in U.S.A.

Program: SAMPLE PROGRAM #2 — Page: 03 — Program Identification: SAMPL2

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions
01	C	01	COUNT	ADD	1	COUNT	3	0
02	C							
03	C							

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 Printed in U.S.A.

Program: SAMPLE PROGRAM #2 — Page: 04 — Program Identification: SAMPL2

Line	Form Type	Filename	Type (H/D/T/E)	Space	Skip	Output Indicators	Field Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01	O	PRINTER	H	2	04	LP				
02	O	OR				OF				
03	O								5	'KEY'
04	O								22	'DESCRIPTION'
05	O								30	'PAGE'
06	O						PAGE		35	
07	O		D	1		01				
08	O						KEY		6	
09	O						DESC		21	
10	O						RECNBR	Z	25	
11	O		T	3	01	LR				
12	O						COUNT	Z	3	
13	O								26	'RECORDS WERE READ FROM'
14	O								44	'THE INDEXED FILE.'
15	O									

Edit codes: Commas / Zero Balances to Print / No Sign / CR / -: Yes Yes 1 A J; Yes No 2 B K; No Yes 3 C L; No No 4 D M. X = Remove Plus Sign; Y = Date Field Edit; Z = Zero Suppress.

Figure B-3 (Part 2 of 2). Specifications for SAMPL2

Instructions on how to load, compile, and execute the RPG II sample programs for Models 8, 10, and 12 are shown in Figure B-4. Listings for SAMPL1 and SAMPL2 are shown in Figures B-5 through B-8. For a description of the programs, see *Sample Program 1* and *Sample Program 2* earlier in this appendix.

Preparing to Run System/3 Model 8, 10, or 12 RPG II Sample Programs

Mount the distribution disk cartridge on R1 and ready the disks.

Set ADDRESS/DATA switches as required.

Set the program load selector at FIXED DISK.

Press PROGRAM LOAD.

Does your system have DPF?

Yes —— No

Yes: EJ is displayed in the message display unit when initial program loading is complete. Press program 1 HALT/ RESET.

Enter these statements using the system input device (MFCU, 1442 Card Read Punch, 5471 Printer-Keyboard, or 3741):

```
// DATE 00/00/00
// CALL $RPSPL,R1
// RUN
```

The system begins reading OCL statements from the system input device.

The system copies the sample programs and all procedures needed to compile and execute the sample programs to F1.

A Figure B-4, part 2

Figure B-4 (Part 1 of 4). Preparing to Run System/3 Model 8, 10, or 12 RPG II Sample Programs

A Figure B-4, part 1

Compiling the SAMPL1 Program (Models 8, 10, or 12)

Note: You must have 5 tracks available on R1 for the files you are using. If you do not have 5 tracks available on the unit specified in the FILE statements of the called procedure, you can change these FILE statements in the procedure, leaving the TRACKS and RETAIN parameters as they are. (See the appropriate SCP reference manual listed under *Related Publications* in the Preface.)

Remove the distribution disk cartridge.

Mount the tailored system disk cartridge on R1 and ready the disks.

Enter these OCL statements using the system input device:

```
// DATE 00/00/00
// CALL $RPSP1,F1
// RUN
//CALL $RPSP2,F1
// RUN
// CALL $RPSP3,F1
// RUN
// CALL $RPSP4,F1
// RUN
```

Ready the printer.

Set ADDRESS/DATA switches to appropriate device if needed.

Set the program load selector at REMOVABLE DISK.

Press PROGRAM LOAD.

Does your system have DPF?

Yes — No

Yes: EJ is displayed in the message display unit when initial program loading is complete. Press program 1 HALT/RESET.

The system begins reading OCL statements from the system input device.

The SAMPL1 program is compiled. EJ is displayed in the message display unit when the SAMPL1 program is compiled. The SAMPL1 object program is on R1.

B Figure B-4, part 3

Figure B-4 (Part 2 of 4). Preparing to Run System/3 Model 8, 10, or 12 RPG II Sample Programs

B Figure B-4, part 2

Note: See Figure B-5 for a printout of SAMPL1 source program.

Compiling the SAMPL2 Program (Model 8, 10, or 12)

Does your system have DPF?

Yes — No

Press program 1 HALT/RESET. | Press console START.

EJ is displayed in the message display unit when the SAMPL2 program is compiled. The SAMPL2 object program is on R1.

Note: See Figure B-6 for a printout of SAMPL2 source program.

Executing the SAMPL1 Program (Model 8, 10, or 12)

Does your system have DPF?

Yes — No

Press program 1 HALT/RESET. | Press console START.

EJ is displayed in the message display unit when the SAMPL1 program has been executed and the output has been printed.

Note: See Figure B-7 for a printout of SAMPL1 program execution.

C Figure B-4, part 4

Figure B-4 (Part 3 of 4). Preparing to Run System/3 Model 8, 10, or 12 RPG II Sample Programs

C Figure B-4, part 3

Executing the SAMPL2 Program (Model 8, 10, or 12)

Does your system have DPF?

Yes — No

Yes: Press program 1 HALT/RESET.

No: Press console START.

EJ is displayed in the message display unit when the SAMPL2 program has been executed and the output has been printed. The program is complete.

Note: See Figure B-8 for a printout of SAMPL2 program execution.

Figure B-4 (Part 4 of 4). Preparing to Run System/3 Model 8, 10, or 12 RPG II Sample Programs

```
// CALL $RPSP1,F1
XX LOAD $RPG,R1
XX FILE NAME-$WORK,UNIT-R1,RETAIN-S,TRACKS-05,PACK-SYSTEM
XX FILE NAME-$SOURCE,UNIT-R1,RETAIN-S,TRACKS-05,PACK-SYSTEM
XX COMPILE UNIT-F1,SOURCE-$SAMP1
XX RUN
// RUN
```

```
     0101 H      008                                                                  SAMPL1
     0102 F***************************************************                        SAMPL1
     0103 F*                                                  *                       SAMPL1
     0104 F* THIS PROGRAM -                                   *                       SAMPL1
     0105 F*                                                  *                       SAMPL1
     0106 F*    1. LOADS 100 RECORDS TO AN INDEXED FILE.      *                       SAMPL1
     0107 F*                                                  *                       SAMPL1
     0108 F*    2. READS ONE RECORD FROM FILE $SOURCE FOR     *                       SAMPL1
     0109 F*       INPUT. THE FILE $SOURCE IS BUILT WHEN      *                       SAMPL1
     01091F*       SAMPLE PROGRAM SAMPL2 IS COMPILED BY       *                       SAMPL1
     01092F*       GIVING A RETAIN-T PARAMETER TO THE         *                       SAMPL1
     01093F*       FILE $SOURCE.                              *                       SAMPL1
     01094F*                                                  *                       SAMPL1
     0110 F*    3. CREATES THE OUTPUT DATA USING A            *                       SAMPL1
     0111 F*       LOOP IN THE CALCULATION SPECIFICATIONS.    *                       SAMPL1
     0112 F*                                                  *                       SAMPL1
     0113 F*    4. USES KEYS FROM 000005 THROUGH 000500       *                       SAMPL1
     0114 F*       IN INCREMENTS OF 5.                        *                       SAMPL1
     0115 F*                                                  *                       SAMPL1
     0116 F*    5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2     *                       SAMPL1
     0117 F*       TO VERIFY THAT THE FILE WAS PROPERLY       *                       SAMPL1
     0118 F*       LOADED.                                    *                       SAMPL1
     0119 F*                                                  *                       SAMPL1
     0120 F***************************************************                        SAMPL1
0001 0121 F$SOURCE IP  F  96  96             DISK                                     SAMPL1
0002 0122 FDISKOUT O   F 256 128 06AI      1 DISK                         01          SAMPL1
0003 0123 FPRINTER O   F  96  96             PRINTER                                  SAMPL1
0004 0201 I$SOURCE NS  01                                                             SAMPL1
0005 0202 I                                    1   1 NODATA                           SAMPL1
0006 0301 C   01                 Z-ADDO         COUNT   60                            SAMPL1
0007 0302 C   01                 Z-ADDO         RECNBR  30                            SAMPL1
0008 0303 C           REPEAT     TAG                                                  SAMPL1
0009 0304 C   01      COUNT      ADD  5         COUNT                                 SAMPL1
0010 0305 C   01      RECNBR     ADD  1         RECNBR                                SAMPL1
0011 0306 C   01      COUNT      COMP 505                        02                   SAMPL1
0012 0307 C   01NC2              EXCPT                                                SAMPL1
0013 0308 C   01NC2              GOTO REPEAT                                          SAMPL1
0014 03081C                      SETON                   LR                           SAMPL1
0015 0309 CLR         RECNBR     SUB  1         RECNBR                                SAMPL1
0016 0401 OPRINTER T  204  LR                                                         SAMPL1
0017 0402 O                                   20 'SAMPLE PROGRAM 1 HAS'               SAMPL1
0018 0403 O                                   27 'LOADED'                             SAMPL1
0019 0404 O                          RECNBRZ  31                                      SAMPL1
0020 0405 O                                   39 'RECORDS'                            SAMPL1
0021 0406 O                                   61 'INTO AN INDEXED FILE.'              SAMPL1
0022 0408 O           T  2     LR                                                     SAMPL1
0023 0409 O                                   21 'KEYS ARE IN ASCENDING'              SAMPL1
0024 0410 O                                   42 'SEQUENCE STARTING AT'               SAMPL1
0025 0411 O                                   64 '000005 AND INCREASING'              SAMPL1
0026 0412 O                                   84 'IN INCREMENTS OF 5.'                SAMPL1
0027 0413 O           T     01 LR                                                     SAMPL1
0028 0414 O                                   21 'SAMPLE PROGRAM 2 WILL'              SAMPL1
0029 0415 O                                   44 'PRINT FROM THE INDEXED'             SAMPL1
0030 0416 O                                   65 'FILE TO SHOW THAT IT'               SAMPL1
0031 0417 O                                   86 'WAS PROPERLY LOADED.'               SAMPL1
0032 0501 ODISKOUT E        01NC2                                                     SAMPL1
0033 0502 O                          COUNT     6                                      SAMPL1
0034 0503 O                                   94 'RECORD NUMBER'                      SAMPL1
0035 0504 O                          RECNBR  128                                      SAMPL1
```

Figure B-5 (Part 1 of 2). SAMPL1 Source Program Printout (Models 8, 10, and 12)

```
 INDICATORS USEC
        LR 01 02

RG 314 UNREFERENCEC FIELD NAMES
 STMT#  NAME
 0005 NOCATA

  FIELD NAMES USEC
 STMT#  NAME  DEC  LGTH   DISP
 0006 COUNT   C    CC6    C1C5
 0007 RECNBR  C    003    01C8

  LABELS USEC
 STMT#  NAME    TYPE
 0008 REPEAT   TAG

  ERROR SEVERITY                                      TEXT
RG  314 W          FIELD, TABLE CR ARRAY NAME CEFINED BUT NEVER USED.

                                         CORE LSAGE CF RPGII CCCE
     START  NAME IF   CODE     NAME      TITLE
     ADDR   OVERLAY   LENGTH
     1100             C642     RGROOT    RCOT
     179A             0091     RGMAIN    INPLT MAINLINE
     1828             0046     RGSUBS    RECCRD IC
     1871             CC26     RGSUBS    CONTROL FIELDS
     1742             005C     RGSUBS    INPLT CTRL RTN
     1792             0008     RGSUBS    SUBSEG
     1897             0C27     $$CSIP    5444 CONSEC INPLT
     188E             C079     $$SRBR    SYSTEM SUBR
     1937             C026     $$SRUA    SYSTEM SUBR
     195D             C01C     $$SRTC    SYSTEM SUBR
     1979             0C81     $$SRMO    SYSTEM SUBR
     19FA             0043     $$SRSB    SYSTEM SUBR
     1A3D             0038     $$SRCI    SYSTEM SUBR
     1A75             002F     $$SRBP    SYSTEM SUBR
     1AB2             0008     RGMAIN    TOTAL CALCS
     1AA4             000E     RGSUBS    CONSTANTS
     1ABD             0C1D     RGMAIN    INPLT FIELDS
     1B88             004B     RGMAIN    DETAIL CALCS
     1B83             0005     RGSUBS    CONSTANTS
     1ACA             009D     RGSUBS    OUTPUT CTRL RTN
     1C03             0043     $$PGRI    RESET RESULTING INDR
     1BD3             C030     RGSUBS    EXCEPTICN
     1B77             CCCC     RGSUBS    SUBSEG
     1C46             0059     $$IOUT    5444 INDEXED CUTPUT
     1C37             C01C     $$SRCF    SYSTEM SUBR
     1C9F             C098     $$SRBI    SYSTEM SUBR
     1D53             0008     RGMAIN    TOTAL CUTPUT
     1D87             0024     RGMAIN    LR & CVERFLCW PROCESSING
     1D6A             C01D     RGSUBS    OVERFLCW SUBSEGMENT
     1D5E             000C     RGSUBS    SUBSEG
     1CAB             00FB     $$LPRT    52C3 PRINT
     1FA3             002D     RGMAIN    CLOSE
     1EA6             00E9     RGSUBS    CONSTANTS
     1FC0             C076     RGSUBS    LR PRCCESSING
     1F8F             0014     RGSUBS    LR CALCS
     2046             CC71     RGMAIN    OPEN

                     04023     SAMPL1    TOTAL CCRE USAGE REQUIRED TC EXECUTE


TOTAL NUMBER OF LIBRARY SECTORS REQUIRED     18
```

Figure B-5 (Part 2 of 2). SAMPL1 Source Program Printout (Models 8, 10, and 12)

```
// CALL $RPSP2,F1
XX LOAD $RPG,R1
XX FILE NAME-$WORK,UNIT-R1,RETAIN-S,TRACKS-05,PACK-SYSTEM
XX FILE NAME-$SOURCE,UNIT-R1,RETAIN-T,TRACKS-05,PACK-SYSTEM
XX COMPILE UNIT-F1,SOURCE-$SAMP2
XX RUN
// RUN
```

```
     0101 H      CC8                                                                                 SAMPL2
     0102 F***********************************************                                           SAMPL2
     0103 F*                                             *                                           SAMPL2
     0104 F* THIS PROGRAM -                              *                                           SAMPL2
     0105 F*                                             *                                           SAMPL2
     0106 F*    1. MUST BE PRECEDED BY SAMPLE PROGRAM 1  *                                           SAMPL2
     0107 F*       WHICH LOADS AN INDEXED FILE.          *                                           SAMPL2
     0108 F*                                             *                                           SAMPL2
     0109 F*    2. READS AN INDEXED FILE SEQUENTIALLY.   *                                           SAMPL2
     0110 F*                                             *                                           SAMPL2
     0111 F*    3. USES A BLOCK LENGTH FOR DISK WHICH    *                                           SAMPL2
     0112 F*       IS DIFFERENT FROM THAT USED FOR       *                                           SAMPL2
     0113 F*       LOADING THE FILE IN SAMPLE PROGRAM 1. *                                           SAMPL2
     0114 F*                                             *                                           SAMPL2
     0115 F*    4. COUNTS THE NUMBER OF RECORDS READ SO  *                                           SAMPL2
     0116 F*       THAT THE USER CAN QUICKLY VERIFY THAT *                                           SAMPL2
     0117 F*       100 RECORDS WERE LOADED.              *                                           SAMPL2
     0118 F*                                             *                                           SAMPL2
     0119 F***********************************************                                           SAMPL2
0001 0120 FDISKIN  IPE F 512 128 06AI      1 DISK                                  01                SAMPL2
0002 0121 FPRINTER O   F  96  96      OF      PRINTER                                                SAMPL2
0003 0201 IDISKIN  NS  01   1 CC                                                                     SAMPL2
0004 0202 I                                        1   6 KEY                                         SAMPL2
0005 0203 I                                       82  94 DESC                                        SAMPL2
0006 0204 I                                      126 1280RECNBR                                      SAMPL2
0007 0301 C   01       COUNT      ADD  1         COUNT   30                                          SAMPL2
0008 0401 OPRINTER H   204     1P                                                                    SAMPL2
0009 0402 O        OR          OF                                                                    SAMPL2
0010 0403 O                                        5 'KEY'                                           SAMPL2
0011 0404 O                                       22 'DESCRIPTION'                                   SAMPL2
0012 0405 O                                       30 'PAGE'                                          SAMPL2
0013 0406 O                          PAGE  Z      35                                                 SAMPL2
0014 0407 O        D  1       01                                                                     SAMPL2
0015 0408 O                          KEY           6                                                 SAMPL2
0016 0409 O                          DESC         21                                                 SAMPL2
0017 0410 O                          RECNBRZ      25                                                 SAMPL2
0018 0411 O        T 3    01 LR                                                                      SAMPL2
0019 0412 O                          COUNT Z       3                                                 SAMPL2
0020 0413 O                                       26 'RECORDS WERE READ FROM'                        SAMPL2
0021 0414 O                                       44 'THE INDEXED FILE.'                             SAMPL2
```

Figure B-6 (Part 1 of 2). SAMPL2 Source Program Printout (Models 8, 10, and 12)

```
INCICATORS USEC
      LR OF IP 01

 FIELD NAMES USEC
STMT#  NAME   CEC   LGTH   DISP
 0013 PAGE     0    0C4    C11C
 0004 KEY           006    C105
 0005 CESC          013    0112
 0006 RECNBR   0    0C3    0115
 0007 COUNT    C    C03    C118

                                            CORE LSAGE CF RPGII CCCE
  START   NAME IF   CODE       NAME       TITLE
  ADCR    OVERLAY   LENGTH
  1100              06AC       RGROOT     ROOT
  1804              00A0       RGMAIN     INPUT MAINLINE
  18A4              C04C       RGSUBS     RECORD IC
  18F0              0026       RGSUBS     CCNTROL FIELDS
  17AC              0050       RGSUBS     INPUT CTRL RTN
  17FC              C008       RGSUBS     SUBSEG
  1916              003B       $$ISIP     5444 IDX SEC INPUT
  1951              0079       $$SRBR     SYSTEM SUBR
  19CA              0038       $$SRDI     SYSTEM SUBR
  1A02              006D       $$SRIC     SYSTEM SUBR
  1A6F              007B       $$SRRC     SYSTEM SUBR
  1AEA              0029       $$SRRI     SYSTEM SUBR
  1B13              CC1C       $$SRTC     SYSTEM SUBR
  1B2F              0081       $$SRMO     SYSTEM SUBR
  1BB0              0043       $$SRSB     SYSTEM SUBR
  1BF3              C02F       $$SRBP     SYSTEM SUBR
  1C22              CC2C       RGMAIN     INPUT FIELDS
  1C4F              0010       RGMAIN     DETAIL CALCS
  1C4E              0001       RGSUBS     CONSTANTS
  1D0D              0032       RGMAIN     DETAIL CUTPUT
  1D08              0005       RGSUBS     CONSTANTS
  1C5F              C05D       RGSUBS     OUTPUT CTRL RTN
  1CFC              000C       RGSUBS     SUBSEG
  1D3F              CCFB       $$LPRT     5203 PRINT
  1E3A              000B       RGMAIN     TOTAL CUTPUT
  1E80              0024       RGMAIN     LR & CVERFLCW PRCCESSING
  1E45              0C17       RGSUBS     CCNSTANTS
  1E5C              0024       RGSUBS     CVERFLCW SUBSEGMENT
  1EA4              0091       RGMAIN     CPEN
  1F35              0028       RGSUBS     SUBSEG
  1F89              C021       RGMAIN     CLCSE
  1F5D              CC2C       RGSUBS     CCNSTANTS
  1FAA              003C       RGSUBS     LR PRCCESSING

                    03802      SAMPL2     TOTAL CCRE USAGE REQUIRED TC EXECUTE
TOTAL NUMBER CF LIBRARY SECTORS REQUIRED    16
```

Figure B-6 (Part 2 of 2). SAMPL2 Source Program Printout (Models 8, 10, and 12)

```
// CALL $RPSP3,F1
XX LOAD SAMPL1,R1
XX FILE NAME-$SOURCE,UNIT-R1,RETAIN-S,PACK-SYSTEM
XX FILE NAME-DISKOUT,UNIT-R1,RETAIN-T,PACK-SYSTEM,RECCRDS-100
XX RUN
// RUN
```

```
SAMPLE PROGRAM 1 HAS LOADED 1CO RECORDS INTC AN INCEXEC FILE.
KEYS ARE IN ASCENDING SEQUENCE STARTING AT CCCCC5 AND INCREASING IN INCREMENTS CF 5.
SAMPLE PROGRAM 2 WILL PRINT FROM THE INDEXED FILE TC SHCW THAT IT WAS PRCPERLY LCACEC.
```

Figure B-7. SAMPL1 Program Execution Printout (Models 8, 10, and 12)

```
// CALL $RPSP4,F1
XX LOAD SAMPL2,R1
XX FILE NAME-DISKIN,LABEL-DISKOUT,UNIT-R1,PACK-SYSTEM,RETAIN-S
XX RUN
// RUN


   KEY        DESCRIPTION       PAGE      1

000005   RECORD NUMBER    1
000010   RECORD NUMBER    2
000015   RECORD NUMBER    3
000020   RECORD NUMBER    4
000025   RECORD NUMBER    5
000030   RECORD NUMBER    6
000035   RECORD NUMBER    7
000040   RECORD NUMBER    8
000045   RECORD NUMBER    9
000050   RECORD NUMBER   10
000055   RECORD NUMBER   11
000060   RECORD NUMBER   12
000065   RECORD NUMBER   13
000070   RECORD NUMBER   14
000075   RECORD NUMBER   15
000080   RECORD NUMBER   16
000085   RECORD NUMBER   17
000090   RECORD NUMBER   18
000095   RECORD NUMBER   19
000100   RECORD NUMBER   20
000105   RECORD NUMBER   21
000110   RECORD NUMBER   22
000115   RECORD NUMBER   23
000120   RECORD NUMBER   24
000125   RECORD NUMBER   25
000130   RECORD NUMBER   26
000135   RECORD NUMBER   27
000140   RECORD NUMBER   28
000145   RECORD NUMBER   29
000150   RECORD NUMBER   30
000155   RECORD NUMBER   31
000160   RECORD NUMBER   32
000165   RECORD NUMBER   33
000170   RECORD NUMBER   34
000175   RECORD NUMBER   35
000180   RECORD NUMBER   36
000185   RECORD NUMBER   37
000190   RECORD NUMBER   38
000195   RECORD NUMBER   39
000200   RECORD NUMBER   40
000205   RECORD NUMBER   41
000210   RECORD NUMBER   42
000215   RECORD NUMBER   43
000220   RECORD NUMBER   44
000225   RECORD NUMBER   45
000230   RECORD NUMBER   46
000235   RECORD NUMBER   47
000240   RECORD NUMBER   48
000245   RECORD NUMBER   49
000250   RECORD NUMBER   50
000255   RECORD NUMBER   51
000260   RECORD NUMBER   52
000265   RECORD NUMBER   53
000270   RECORD NUMBER   54
000275   RECORD NUMBER   55
```

Figure B-8 (Part 1 of 2). SAMPL2 Program Execution Printout (Models 8, 10, and 12)

```
  KEY        DESCRIPTION      PAGE      2

000280   RECORD NUMBER   56
000285   RECORD NUMBER   57
000290   RECORD NUMBER   58
000295   RECORD NUMBER   59
000300   RECORD NUMBER   60
000305   RECORD NUMBER   61
000310   RECORD NUMBER   62
000315   RECORD NUMBER   63
000320   RECORD NUMBER   64
000325   RECORD NUMBER   65
000330   RECORD NUMBER   66
000335   RECORD NUMBER   67
000340   RECORD NUMBER   68
000345   RECORD NUMBER   69
000350   RECORD NUMBER   70
000355   RECORD NUMBER   71
000360   RECORD NUMBER   72
000365   RECORD NUMBER   73
000370   RECORD NUMBER   74
000375   RECORD NUMBER   75
000380   RECORD NUMBER   76
000385   RECORD NUMBER   77
000390   RECORD NUMBER   78
000395   RECORD NUMBER   79
000400   RECORD NUMBER   80
000405   RECORD NUMBER   81
000410   RECORD NUMBER   82
000415   RECORD NUMBER   83
000420   RECORD NUMBER   84
000425   RECORD NUMBER   85
000430   RECORD NUMBER   86
000435   RECORD NUMBER   87
000440   RECORD NUMBER   88
000445   RECORD NUMBER   89
000450   RECORD NUMBER   90
000455   RECORD NUMBER   91
000460   RECORD NUMBER   92
000465   RECORD NUMBER   93
000470   RECORD NUMBER   94
000475   RECORD NUMBER   95
000480   RECORD NUMBER   96
000485   RECORD NUMBER   97
000490   RECORD NUMBER   98
000495   RECORD NUMBER   99
000500   RECORD NUMBER  100

100 RECORDS WERE READ FROM THE INDEXED FILE.
```

Figure B-8 (Part 2 of 2). SAMPL2 Program Execution Printout (Models 8, 10, and 12)

SYSTEM/3 MODEL 15 RPG II SAMPLE PROGRAMS

Instructions on how to load, compile, and execute the RPG II sample programs for Model 15 are shown in Figure B-9. Listings for SAMPL1 and SAMPL2 are shown in Figure B-10. For a description of the programs, see *Sample Program 1* and *Sample Program 2* earlier in this appendix.

Preparing to Run the System/3 Model 15 RPG II Sample Programs

Ensure that you have a copy of the generated system on F1.

Mount the distribution disk on drive 1 and ready the disk.

Set the program load selector to FIXED DISK.

Ready the printer.

Press PROGRAM LOAD.

The system issues a prompt to allow you to enter a system date. Respond by entering either the system date or a null response.

System date

Press PF12.

Enter the system date in the same format as that allowed on the DATE OCC or the DATE OCL statement.

Null response

Press PF12.

Press ENTER.

A system date must be given via OCL or OCC.

After initial program load (IPL) has taken place, EJ is displayed on the CRT.

Does your system support spooling?

Yes

Cancel spooling as follows:

Press PF10.

Key in CANCEL SPOOL.

Press ENTER.

No

A

Figure B-9, part 2

Figure B-9 (Part 1 of 5). Running the System/3 Model 15 RPG II Sample Programs

A Figure B-9, part 1

Press PF12.

Press ENTER.

Enter the following OCL statements using the system input device selected during system generation unless you have altered the device assignment during IPL by entering an OCL statement or an OCC. The following statements copy the sample programs and procedures from the distribution pack to your system on F1:

```
// LOG 1403
// HALT
// CALL $RPSPL,R1
// RUN
```

The system displays EJ on the CRT when this job step is complete.

Does your system use 5444 disk or 3340 disk?

5444:

Remove the distribution cartridge from R1.

Mount the tailored system disk cartridge on R1 and ready disk.

3340:

Copy the System Control Program (SCP) area from R1 5444 simulation area to a backup area if needed.

Copy the tailored system to R1 5444 simulation area (area name – SYSTEM).

Set the program load selector to REMOVABLE DISK.

Press PROGRAM LOAD.

Press PF12.

Key in the system date in the format you have selected.

B Figure B-9, part 3

Figure B-9 (Part 2 of 5). Running the System/3 Model 15 RPG II Sample Programs

B Figure B-9, part 2

Does your system support spooling?

Yes — No

Cancel spooling as follows:

Press PF10.

Key in CANCEL SPOOL.

Press ENTER.

Press PF10.

Key in SET P1,12 (this sets the partition size to 12K).

Press ENTER.

Compiling the SAMPL1 Program (Model 15)

Note: You must have 20 tracks available for the files you are using. If you do not have 20 tracks available on the unit specified in the FILE statement of the called procedure, you may change these FILE statements in the procedure leaving TRACKS and RETAIN statements as they are. For information on how to change procedures, see the appropriate SCP reference manual listed under *Related Publications* in the Preface.

Press PF12.

Press ENTER.

Enter these OCL statements from the assigned input device:

```
// LOG 1403
// HALT
// CALL $RPSP1,F1
// RUN
```

C Figure B-9, part 4

Figure B-9 (Part 3 of 5). Running the System/3 Model 15 RPG II Sample Programs

C Figure B-9, part 3

When this program stops with EJ dsiplayed on the CRT, the SAMPL1 object program is on R1.

Compiling the SAMPL2 Program

Press PF12.

Press ENTER.

Enter these OCL statements from the assigned input device:

```
// CALL $RPSP2,F1
// RUN
```

When this program stops with EJ displayed on the CRT, the SAMPL2 object program is on R1.

Executing the SAMPL1 Program

Note: If the sample programs are running on a 3340 system, the indexed file will be created on the main data area. The volume must be mounted on D1 and have a label of PID001.

Press PF12.

Press ENTER.

Enter these OCL statements from the assigned input device:

```
// CALL $RPSP3,F1
// RUN
```

When this program stops with EJ displayed on the CRT, the SAMPL1 program will have loaded 100 records into a file. Keys are in ascending sequence starting at 000005 and increased in increments of 5.

D Figure B-9, part 5

Figure B-9 (Part 4 of 5). Running the System/3 Model 15 RPG II Sample Programs

D Figure B-9, part 4

Executing the SAMPL2 Program

Press PF12.

Press ENTER.

Enter these OCL statements from the assigned input device:

```
// CALL $RPSP4,F1
// RUN
```

When this program stops with EJ displayed on the CRT, the SAMPL2 program will print from the indexed file to show it was properly loaded.

Figure B-9 (Part 5 of 5). Running the System/3 Model 15 RPG II Sample Programs

```
// CALL $RPSP1,F1
XX LOAD $RPG,R1
XX FILE NAME-$WORK,UNIT-R1,PACK-SYSTEM,RETAIN-S,TRACKS-10
XX FILE NAME-$SOURCE,UNIT-R1,PACK-SYSTEM,RETAIN-S,TRACKS-10
XX COMPILE UNIT-F1,SOURCE-$SAMP1
XX RUN
// RUN


   SYSTEM/3 MODEL 15                                             12K


        0101 H      008                                                                  SAMPL1


        0102 F************************************************                           SAMPL1
        0103 F*                                              *                           SAMPL1
        0104 F* THIS PROGRAM -                               *                           SAMPL1
        0105 F*                                              *                           SAMPL1
        0106 F*    1. LOADS 100 RECORDS TO AN INDEXED FILE.  *                           SAMPL1
        0107 F*                                              *                           SAMPL1
        0108 F*    2. READS ONE RECORD FROM FILE $SOURCE FOR *                           SAMPL1
        0109 F*       INPUT. THE FILE $SOURCE IS BUILT WHEN  *                           SAMPL1
        01091F*       SAMPLE PROGRAM SAMPL2 IS COMPILED BY   *                           SAMPL1
        01092F*       GIVING A RETAIN-T PARAMETER TO THE     *                           SAMPL1
        01093F*       FILE $SOURCE.                          *                           SAMPL1
        01094F*                                              *                           SAMPL1
        0110 F*    3. CREATES THE OUTPUT DATA USING A        *                           SAMPL1
        0111 F*       LOOP IN THE CALCULATION SPECIFICATIONS.*                           SAMPL1
        0112 F*                                              *                           SAMPL1
        0113 F*    4. USES KEYS FROM 000005 THROUGH 000500   *                           SAMPL1
        0114 F*       IN INCREMENTS OF 5.                    *                           SAMPL1
        0115 F*                                              *                           SAMPL1
        0116 F*    5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2 *                           SAMPL1
        0117 F*       TO VERIFY THAT THE FILE WAS PROPERLY   *                           SAMPL1
        0118 F*       LOADED.                                *                           SAMPL1
        0119 F*                                              *                           SAMPL1
        0120 F************************************************                           SAMPL1
0001    0121 F$SOURCE IP   F 256 256              DISK                                   SAMPL1
0002    0122 FDISKOUT O    F 256 128 06AI       1 DISK                         01        SAMPL1
0003    0123 FPRINTER O    F  96  96              PRINTER                                SAMPL1


0004    0201 I$SOURCE NS  01                                                             SAMPL1
0005    0202 I                                             1    1 NODATA                 SAMPL1


0006    0301 CL0                    Z-ADD0          COUNT   60                           SAMPL1
0007    0302 CL0                    Z-ADD0          RECNBR  30                           SAMPL1
0008    0303 CL0          REPEAT    TAG                                                  SAMPL1
0009    0304 CL0          COUNT     ADD  5          COUNT                                SAMPL1
0010    0305 CL0          RECNBR    ADD  1          RECNBR                               SAMPL1
0011    0306 CL0          COUNT     COMP 505                      02                     SAMPL1
0012    0307 CL0     N02            EXCPT                                                SAMPL1
0013    0308 CL0     N02            GOTO REPEAT                                          SAMPL1
0014    03081CL0                    SETON                      LR                        SAMPL1
0015    0309 CLR          RECNBR    SUB  1          RECNBR                               SAMPL1
```

Figure B-10 (Part 1 of 8). RPG II Sample Program Printout (Model 15)

```
0016   0401 CPRINTER T   204    LR                                                   SAMPL1
0017   0402 C                                      20 'SAMPLE PROGRAM 1 HAS'          SAMPL1
0018   0403 C                                      27 'LOADED'                        SAMPL1
0019   0404 C                         RECNBRZ      31                                 SAMPL1
0020   0405 O                                      39 'RECORDS'                       SAMPL1
0021   0406 O                                      61 'INTO AN INDEXED FILE.'         SAMPL1
0022   0408 C          T   2        LR                                               SAMPL1
0023   0409 O                                      21 'KEYS ARE IN ASCENDING'         SAMPL1
0024   0410 C                                      42 'SEQUENCE STARTING AT'          SAMPL1
0025   0411 C                                      64 '000005 AND INCREASING'         SAMPL1
0026   0412 O                                      84 'IN INCREMENTS OF 5.'           SAMPL1
0027   0413 C          T            LR                                               SAMPL1
0028   0414 C                                      21 'SAMPLE PROGRAM 2 WILL'         SAMPL1
0029   0415 O                                      44 'PRINT FROM THE INDEXED'        SAMPL1
0030   0416 O                                      65 'FILE TO SHOW THAT IT'          SAMPL1
0031   0417 O                                      86 'WAS PROPERLY LOADED.'          SAMPL1
0032   0501 ODISKOUT E             NO2                                               SAMPL1
0033   0502 C                         COUNT         6                                 SAMPL1
0034   0503 O                                      94 'RECORD NUMBER'                 SAMPL1
0035   0504 O                         RECNBR      128                                 SAMPL1


 INDICATORS USED
       LR LC C1 02

RG 305   INDICATORS UNREFERENCED
       01

RG 314 UNREFERENCED FIELD NAMES
 STMT#   NAME
  0005 NODATA

  FIELD NAMES USED
 STMT#   NAME   DEC   LGTH   DISP
  0006 COUNT    0     006    0005
  0007 RECNBR   0     003    0008

  LABELS USED
 STMT#   NAME    TYPE
  0008 REPEAT   TAG

  ERROR NUMBER   STATEMENT NUMBER
    RG 273            0032

  ERROR SEVERITY                                     TEXT
RG   273 W        OUTPUT INDICATORS IN COL 23-31 MISSING OR ALL NEGATIVE.
RG   305 W        INDICATOR ASSIGNED BUT NOT USED TO CONDITION OPERATIONS.
RG   314 W        FIELD, TABLE OR ARRAY NAME DEFINED BUT NEVER USED.
```

Figure B-10 (Part 2 of 8). RPG II Sample Program Printout (Model 15)

```
OL100 I   THE TOTAL CORE USED BY SAMPL1 IS  5269 DECIMAL.
OL101 I   THE START CONTROL ADDRESS OF THIS MODULE IS 4800.
OL104 I   TOTAL NUMBER OF LIBRARY SECTORS REQUIRED IS   13
          NAME-SAMPL1,PACK-SYSTEM,UNIT-R1,RETAIN-T,LIBRARY-O


        OVERLAY LINKAGE EDITOR CORE USAGE MAP

 START    CATEGORY   NAME AND       CODE LENGTH
ADDRESS               ENTRY     HEXADECIMAL DECIMAL

 4000                GLOBAL         07C8     1992
 47C8                COMMON         0009        9
 4800        0       SAMPL1         0100      256
 4900        0       $#RTC2         0123      291
 4A23        0       $#IPCR         004F       79
 4A24                  $@0AC9
 4A72        0       $#OPCR         009C      156
 4A73                  $@0B18
 4B0E        0       $#CON0         00E9      233
 4BF7        0       $#CON1         0005        5
 4BFC        0       $#CON2         000E       14
 4C0A        2       $$CSIP         0027       39
 4C31        2       $$ICUT         005F       95
 4C90        2       $$SRBR         0082      130
 4D12        2       $$SRLA         0026       38
 4D38        2       $$SRTC         001C       28
 4D38                  DMSRLC
 4D49                  DMSRTC
 4D4C                  DMSRER
 4D54        2       $$SRBI         0112      274
 4E66        2       $$SRDF         001C       28
 4E82        2       $$SRMO         00A4      164
 4F26        2       $$SRSB         0046       70
 4F6C        2       $$SRDI         0095      149
 4F91                  DMSRPD
 4F8A                  DMSRRD
 5001        2       $$SRBP         003B       59
 503C        6       $$LPRT         00F0      240
 512C       93       $#OPEN         006D      109
 5199      126       $#INPT         0090      144
 51EB                  $@0C25
 51F2                  $@0C2C
 521F                  $@0C59
 51E3                  $@0C1D
 51E7                  $@0C21
 5229       28       $#IHC1         0008        8
 5231      126       $#TCAL         0052       82
 5283       29       $#EXPT         002D       45
 52B0       28       $#CHC2         000C       12
 52BC      126       $#IFLD         001D       29
 52D5                  $@0CE8
 52D9       93       $#CLOS         002B       43
 52E0                  $@0EC2
 52F1                  $@0ED3
 52F5                  $@0EC7
 5304      107       $#LRCT         0076      118
 537A      126       $#TOUT         000B       11
 5385      126       $#LROF         0024       36
 53A9       71       $#DACF         001D       29
 53C6       28       $#CHC3         000C       12
 53D2      126       $#RCID         0046       70
 5409                  $@0C9A
 5418      126       $#CFLD         0026       38
 543E       11       $$PGRI         0043       67
```

Figure B-10 (Part 3 of 8). RPG II Sample Program Printout (Model 15)

```
// CALL $RPSP2,F1
XX LOAD $RPG,R1
XX FILE NAME-$WORK,UNIT-R1,PACK-SYSTEM,RETAIN-S,TRACKS-10
XX FILE NAME-$SOURCE,UNIT-R1,PACK-SYSTEM,RETAIN-S,TRACKS-10
XX COMPILE UNIT-F1,SOURCE-$SAMP2
XX RUN
// RUN


   SYSTEM/3 MODEL 15                                                  12K


      0101 H      008                                                                         SAMPL2


      0102 F*************************************************                                 SAMPL2
      0103 F*                                               *                                 SAMPL2
      0104 F* THIS PROGRAM -                                *                                 SAMPL2
      0105 F*                                               *                                 SAMPL2
      0106 F*    1. MUST BE PRECEDED BY SAMPLE PROGRAM 1    *                                 SAMPL2
      0107 F*       WHICH LOADS AN INDEXED FILE.            *                                 SAMPL2
      0108 F*                                               *                                 SAMPL2
      0109 F*    2. READS AN INDEXED FILE SEQUENTIALLY.     *                                 SAMPL2
      0110 F*                                               *                                 SAMPL2
      0111 F*    3. USES A BLOCK LENGTH FOR DISK WHICH      *                                 SAMPL2
      0112 F*       IS DIFFERENT FROM THAT USED FOR         *                                 SAMPL2
      0113 F*       LOADING THE FILE IN SAMPLE PROGRAM 1.   *                                 SAMPL2
      0114 F*                                               *                                 SAMPL2
      0115 F*    4. COUNTS THE NUMBER OF RECORDS READ SO    *                                 SAMPL2
      0116 F*       THAT THE USER CAN QUICKLY VERIFY THAT   *                                 SAMPL2
      0117 F*       100 RECORDS WERE LOADED.                *                                 SAMPL2
      0118 F*                                               *                                 SAMPL2
      0119 F*************************************************                                 SAMPL2
0001  0120 FDISKIN  IPE F 512 128 06AI     1 DISK                          01                 SAMPL2
0002  0121 FPRINTER O    F  96  96      OF     PRINTER                                        SAMPL2


0003  0201 IDISKIN   NS   01    1 C0                                                          SAMPL2
0004  0202 I                                          1    6 KEY                              SAMPL2
0005  0203 I                                         82   94 DESC                             SAMPL2
0006  0204 I                                        126 1280RECNBR                            SAMPL2


0007  0301 C   01        COUNT      ADD  1          COUNT    30                               SAMPL2


0008  0401 OPRINTER H   204    1P                                                             SAMPL2
0009  0402 O          OR         OF                                                           SAMPL2
0010  0403 O                                       5 'KEY'                                    SAMPL2
0011  0404 O                                      22 'DESCRIPTION'                            SAMPL2
0012  0405 O                                      30 'PAGE'                                   SAMPL2
0013  0406 O                         PAGE  Z      35                                          SAMPL2
0014  0407 O           D  1      01                                                           SAMPL2
0015  0408 O                         KEY           6                                          SAMPL2
0016  0409 O                         DESC         21                                          SAMPL2
0017  0410 O                         RECNBRZ      25                                          SAMPL2
0018  0411 O           T 3    01 LR                                                           SAMPL2
0019  0412 O                         COUNT Z       3                                          SAMPL2
0020  0413 O                                      26 'RECORDS WERE READ FROM'                 SAMPL2

0021  0414 O                                      44 'THE INDEXED FILE.'                      SAMPL2
```

Figure B-10 (Part 4 of 8). RPG II Sample Program Printout (Model 15)

```
INDICATORS USED
       LR OF 1P 01

 FIELD NAMES USED
STMT#  NAME   DEC   LGTH   DISP
 0013 PAGE     0    004    001C
 0004 KEY           006    0005
 0005 DESC          013    0012
 0006 RECNBR   0    003    0015
 0007 COUNT    0    003    0018
        OVERLAY LINKAGE EDITOR CORE USAGE MAP


 START    CATEGORY  NAME AND       CODE LENGTH
ADDRESS              ENTRY    HEXADECIMAL  DECIMAL

 4000               GLOBAL          04D2     1234
 44D2               COMMON          001D       29
 4500       0       SAMPL2          0100      256
 4600       0       $#RT02          00C4      196
 46C4       0       $#IPCR          004F       79
 46C5                 $@06D3
 4713       0       $#OPCR          009C      156
 4714                 $@0722
 47AF       0       $#CON0          0005        5
 47B4       0       $#CON1          002C       44
 47E0       0       $#CON2          0001        1
 47E1       0       $#CON3          0017       23
 47F8       2       $$ISIP          004F       79
 4847       2       $$SRMO          00A4      164
 48EB       2       $$SRDI          0095      149
 4910                 DMSRPD
 4909                 DMSRRD
 4980       2       $$SRIC          00D3      211
 49D8                 DMSRIF
 4A53       2       $$SRRC          00CA      202
 4B1D       2       $$SRRI          0048       72
 4B65       2       $$SRTC          001C       28
 4B65                 DMSRLO
 4B76                 DMSRTC
 4B79                 DMSRER
 4B81       2       $$SRBP          003B       59
 4BBC       6       $$LPRT          00F0      240
 4CAC      93       $#OPEN          008D      141
 4D39     126       $#INPT          009F      159
 4D9A                 $@0832
 4DA1                 $@0839
 4DCE                 $@0866
 4D92                 $@082A
 4D96                 $@082E
 4DD8      28       $#IH01          0008        8
 4DE0     126       $#IFLD          002C       44
 4DF9                 $@08FB
 4E0C     126       $#DCAL          0010       16
 4E1C      93       $#CLOS          001F       31
 4E1F                 $@09F7
 4E2C                 $@0AC4
 4E3B     107       $#LROT          0030       48
 4E6B     126       $#DOUT          0032       50
 4E9D     126       $#TOUT          000B       11
 4EA8     126       $#LROF          0024       36
 4ECC      71       $#OFOF          0024       36
 4EF0      28       $#CHC2          000C       12
 4EFC      35       $#MF01          0028       40
 4F24     126       $#RCIC          004C       76
 4F5B                 $@08A7
 4F70     126       $#CFLD          0026       38
OL100 I  THE TOTAL CORE USED BY SAMPL2 IS  3990 DECIMAL.
OL101 I  THE START CONTROL ADDRESS OF THIS MODULE IS 4500.
OL104 I  TOTAL NUMBER OF LIBRARY SECTORS REQUIRED IS   11
         NAME-SAMPL2,PACK-SYSTEM,UNIT-R1,RETAIN-T,LIBRARY-O
```

Figure B-10 (Part 5 of 8). RPG II Sample Program Printout (Model 15)

```
// CALL $RPSP3,F1
XX LOAD SAMPL1,R1
XX FILE NAME-$SOURCE,UNIT-R1,RETAIN-S,PACK-SYSTEM
XX FILE NAME-DISKOUT,UNIT-R1,RETAIN-T,PACK-SYSTEM,RECORDS-100
XX RUN
// RUN
```

```
SAMPLE PROGRAM 1 HAS LOADED 100 RECORDS INTO AN INDEXED FILE.
KEYS ARE IN ASCENDING SEQUENCE STARTING AT 000005 AND INCREASING IN INCREMENTS OF 5.
SAMPLE PROGRAM 2 WILL PRINT FROM THE INDEXED FILE TO SHOW THAT IT WAS PROPERLY LOADED.
```

Figure B-10 (Part 6 of 8). RPG II Sample Program Printout (Model 15)

```
// CALL $RPSP4,F1
XX LOAD SAMPL2,R1
XX FILE NAME-DISKIN,LABEL-DISKOUT,UNIT-R1,PACK-SYSTEM,RETAIN-S
XX RUN
// RUN
```

```
  KEY        DESCRIPTION      PAGE      1

000005   RECORD NUMBER    1
000010   RECORD NUMBER    2
000015   RECORD NUMBER    3
000020   RECORD NUMBER    4
000025   RECORD NUMBER    5
000030   RECORD NUMBER    6
000035   RECORD NUMBER    7
000040   RECORD NUMBER    8
000045   RECORD NUMBER    9
000050   RECORD NUMBER   10
000055   RECORD NUMBER   11
000060   RECORD NUMBER   12
000065   RECORD NUMBER   13
000070   RECORD NUMBER   14
000075   RECORD NUMBER   15
000080   RECORD NUMBER   16
000085   RECORD NUMBER   17
000090   RECORD NUMBER   18
000095   RECORD NUMBER   19
000100   RECORD NUMBER   20
000105   RECORD NUMBER   21
000110   RECORD NUMBER   22
000115   RECORD NUMBER   23
000120   RECORD NUMBER   24
000125   RECORD NUMBER   25
000130   RECORD NUMBER   26
000135   RECORD NUMBER   27
000140   RECORD NUMBER   28
000145   RECORD NUMBER   29
000150   RECORD NUMBER   30
000155   RECORD NUMBER   31
000160   RECORD NUMBER   32
000165   RECORD NUMBER   33
```

```
  KEY        DESCRIPTION      PAGE      2

000170   RECORD NUMBER   34
000175   RECORD NUMBER   35
000180   RECORD NUMBER   36
000185   RECORD NUMBER   37
000190   RECORD NUMBER   38
000195   RECORD NUMBER   39
000200   RECORD NUMBER   40
000205   RECORD NUMBER   41
000210   RECORD NUMBER   42
000215   RECORD NUMBER   43
000220   RECORD NUMBER   44
000225   RECORD NUMBER   45
000230   RECORD NUMBER   46
000235   RECORD NUMBER   47
000240   RECORD NUMBER   48
000245   RECORD NUMBER   49
000250   RECORD NUMBER   50
000255   RECORD NUMBER   51
000260   RECORD NUMBER   52
000265   RECORD NUMBER   53
000270   RECORD NUMBER   54
000275   RECORD NUMBER   55
000280   RECORD NUMBER   56
000285   RECORD NUMBER   57
000290   RECORD NUMBER   58
000295   RECORD NUMBER   59
000300   RECORD NUMBER   60
000305   RECORD NUMBER   61
000310   RECORD NUMBER   62
000315   RECORD NUMBER   63
000320   RECORD NUMBER   64
000325   RECORD NUMBER   65
000330   RECORD NUMBER   66
```

Figure B-10 (Part 7 of 8). RPG II Sample Program Printout (Model 15)

```
  KEY        DESCRIPTION     PAGE     3

000335   RECORD NUMBER   67
000340   RECORD NUMBER   68
000345   RECORD NUMBER   69
000350   RECORD NUMBER   70
000355   RECORD NUMBER   71
000360   RECORD NUMBER   72
000365   RECORD NUMBER   73
000370   RECORD NUMBER   74
000375   RECORD NUMBER   75
000380   RECORD NUMBER   76
000385   RECORD NUMBER   77
000390   RECORD NUMBER   78
000395   RECORD NUMBER   79
000400   RECORD NUMBER   80
000405   RECORD NUMBER   81
000410   RECORD NUMBER   82
000415   RECORD NUMBER   83
000420   RECORD NUMBER   84
000425   RECORD NUMBER   85
000430   RECORD NUMBER   86
000435   RECORD NUMBER   87
000440   RECORD NUMBER   88
000445   RECORD NUMBER   89
000450   RECORD NUMBER   90
000455   RECORD NUMBER   91
000460   RECORD NUMBER   92
000465   RECORD NUMBER   93
000470   RECORD NUMBER   94
000475   RECORD NUMBER   95
000480   RECORD NUMBER   96
000485   RECORD NUMBER   97
000490   RECORD NUMBER   98
000495   RECORD NUMBER   99
```

```
  KEY        DESCRIPTION     PAGE     4

000500   RECORD NUMBER 100

100 RECORDS WERE READ FROM THE INDEXED FILE.
```

Figure B-10 (Part 8 of 8). RPG II Sample Program Printout (Model 15)

EXAMPLE PROGRAMS

This example contains specifications sheets for three complete RPG II programs: EXMPL1, EXMPL2, and EXMPL3. The programs are designed to be run in sequence and can be run on any IBM System/3 Disk System that has an MFCU.

Example Program 1

EXMPL1 loads master records into an indexed file and creates a consecutive file of transactions. The transaction file will be processed against the master file in EXMPL2. EXMPL2 should follow EXMPL1. Figure B-11 shows the completed specifications sheets for EXMPL1.

Control Card Specifications

This card should be present in every job. It is the first card in the source deck.

File Description Specifications

These specifications describe the files in the program. The input card file, CARDIN, is read from the primary MFCU hopper. An E in column 17 indicates that the program will end when the last data record in the input file has been processed. The indexed output file, MASTER, will consist of 26-position records with a 5-position key field starting in the second record position. MASTER is a single volume file (01 in columns 68-69). A consecutive output file, TRANS, with a 10-position record length is also specified on the File Description Sheet. TRANS is also a single volume file (01 in columns 68-69). A printer output file, PRINTER, with a record length of 78 is also defined on the File Description Sheet.

Input Specifications

There are two types of records in the input card file, CARDIN: master and transaction. A character M in position 1 of the input records will turn on record identifying indicator 01, indicating a master record. A character A, B, or C in position 1 of the input records will turn on record identifying indicator 02, indicating a transaction record. No sequence checking will occur for either type of record (AA and AB in columns 15-16).

Calculation Specifications

The field named TOTMAS is incremented by one when record identifying indicator 01 is on. This maintains a running total of the master records which have been read from CARDIN and transferred to disk. The field TOTTRN is incremented by one when record identifying indicator 02 is on, maintaining a running total of the transaction records which have been read from CARDIN and transferred to disk.

Output-Format Specifications

Four different output records are described in these specifications: one detail record for the master file (MASTER), one detail record for the transaction file (TRANS), and two total records for the printer file (PRINTER).

The detail records for MASTER are conditioned by record identifying indicator 01. The detail records for TRANS are conditioned by record identifying indicator 02.

Both total lines for PRINTER are printed when the last record identifying indicator is turned on (LR in columns 23-25). The first total line is for total transactions loaded. The printer skips to line 4 before the printing of the first total line and double spacing occurs before the printing of the second total line. The second total line is for total masters loaded. The printer skips to line 1 of the next page after it is printed.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 Printed in U.S.A.

Program: EXAMPLE PROGRAM #1
Programmer: Date:
Punching Instruction: Graphic / Punch
Card Electro Number

Page Ø1 of __ Program Identification (75–80): EXMPL1

Control Card Specifications

Refer to the specific System Reference Library manual for actual entries.

```
  01 H     ØØ8
```

File Description Specification

```
  02 F*************************************************
  03 F*                                                
  04 F* THIS PROGRAM                                   
  05 F*                                                
  06 F*     1. LOADS MASTER RECORDS INTO AN INDEXED    
  07 F*        FILE.                                   
  08 F*                                                
  09 F*     2. CREATES A CONSECUTIVE FILE OF           
  10 F*        TRANSACTIONS.                           
  11 F*                                                
  12 F*     3. COUNTS THE TOTAL NUMBER OF RECORDS LOADED
  13 F*        INTO EACH FILE. THIS TOTAL IS PRINTED AT
  14 F*        THE END OF JOB.                         
  15 F*                                                
  16 F*************************************************
  17 FCARDIN  IPE F  96  96            MFCU1
  18 FMASTER  O   F  52  26 05AI     2 DISK                       Ø1
  19 FTRANS   O   F  30  10            DISK                       Ø1
  20 FPRINTER O   F  78  78            PRINTER
```

Figure B-11 (Part 1 of 3). Specifications for EXMPL1

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094, Printed in U.S.A.

Program: EXAMPLE PROGRAM #1 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page 02 of __ | Program Identification (75–80): EXMPL1

Line	Form Type	Filename	Sequence (OR/AND)	Number (1-N)	Option (O)	Record Identifying Indicator	Rec. ID Code 1 Position	Not (N)	C/Z/D	Character	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
01	I	*																		
02	I	* MASTER RECORD DESCRIPTION																		
03	I	*																		
04	I	CARDIN	AA			01	1		C	M										
05	I										1	1		ID						
06	I										2	6		KEY						
07	I										7	14		DESC						
08	I										15	18	0	VALUEA						
09	I										19	22	0	VALUEB						
10	I										23	26	0	VALUEC						
11	I	*																		
12	I	* TRANSACTION RECORD DESCRIPTIONS																		
13	I	*																		
14	I		AB			02	1		C	A										
15	I		OR			02	1		C	B										
16	I		OR			02	1		C	C										
17	I										1	1		ID						
18	I										2	6		KEY						
19	I										7	10	0	AMT						
20	I																			

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093, Printed in U.S.A.

Program: EXAMPLE PROGRAM #1 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page 03 of __ | Program Identification (75–80): EXMPL1

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	And	And	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus	Minus	Zero	Comments
01	C		01			TOTMAS	ADD	1	TOTMAS	4	0					
02	C		02			TOTTRN	ADD	1	TOTTRN	4	0					
03	C															
04	C															
05	C															

Figure B-11 (Part 2 of 3). Specifications for EXMPL1

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 Printed in U.S.A.

Program: EXAMPLE PROGRAM #1 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page 04 of __ | Program Identification (75–80): EXMPL1

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

Line (3–5)	Form Type (6)	Filename (7–14)	Type (H/D/T/E) (15)	Space Before (17)	Space After (18)	Skip Before (19–20)	Skip After (21–22)	Output Indicators (23–31)	Field Name (32–37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40–43)	P/B/L/R (44)	Constant or Edit Word (45–70)
01	O	MASTER	D					Ø1						
02	O								VALUEC			26		
03	O								VALUEB			22		
04	O								VALUEA			18		
05	O								DESC			14		
06	O								KEY			6		
07	O								ID			1		
08	O	TRANS	D					Ø2						
09	O								AMT			1Ø		
10	O								KEY			6		
11	O								ID			1		
12	O	PRINTER	T		2	Ø4		LR						
13	O											24		'TRANSACTIONS LOADED'
14	O								TOTTRN	Z				
15	O		T				Ø1	LR						
16	O											19		'MASTERS LOADED'
17	O								TOTMAS	Z		4		
18	O													
19	O													
20	O													
	O													
	O													
	O													
	O													
	O													

Figure B-11 (Part 3 of 3). Specifications for EXMPL1

Example Program 2

EXMPL2 must be preceded by EXMPL1. EXMPL2 reads from the transaction file, TRANS, created by EXMPL1 and accumulates totals for A, B, and C records. EXMPL2 also retrieves matching master records for transaction records and prints an error message if a matching master record is not found. Figure B-12 shows the completed specifications sheets for EXMPL2.

Control Card Specifications

This card must be present in every job. It is the first card in the source deck.

File Description Specifications

The input file for EXMPL2, TRANS (the output transaction file for EXMPL1), is read from disk. An E in column 17 indicates that the program will end when the last data record in the input file has been processed. TRANS is a single volume file (01 in columns 68-69). The output file, PRINTER, will consist of 72-position records. An overflow indicator (OF in columns 33-34) is being used to condition printing of records in the file. The indexed file, MASTER, is described as a chained update file to be processed by keys. It consists of 26-position records with a 5-position key field starting in the second record position. It is a single volume file on disk.

Input Specifications

There are two types of files specified on the Input Sheet: transaction and master. A character A, B, or C in position 1 of the input records will turn on record identifying indicator 01, 02, or 03, indicating a transaction record type A, B, or C respectively. A character M in position 1 of the update records will turn on record identifying indicator 04, indicating an update record. No sequence checking will occur for either type (AA and AB in columns 15-16).

Calculation Specifications

When indicator 01, 02, or 03 is on, two operations will occur:

1. A matching master record is retrieved for a transaction record (lines 01, 02, and 03 on the Calculation Sheet).

2. The AMT field of the transaction cards is added to the appropriate value (VALUEA, VALUEB, or VALUEC) on the master card depending on the type of card (record identifying indicator 01, 02, or 03).

If no matching record is found, indicator 10 will be turned on.

Output Specifications

Eight printer output lines are described in these specifications. Four header lines conditioned by the first page indicator (1P in columns 23-25) or an overflow indicator (OF in columns 23-25) are printed. They will be printed at the top of each page of the listing.

Four detail lines are also printed. A detail line is printed for each transaction record with no matching master record (line 20 on page 04 and lines 01-03 on page 05). For each type of transaction record, A, B, or C, the accumulative value is printed (detail lines conditioned by indicators 01, 02, or 03, and not 10). These detail lines are single spaced.

A detail record is written on disk for the indexed update file, MASTER. It is conditioned by two indicators – the record identifying indicator 04 and not 10 which is the record identifying indicator for no matching master record, a match between the master and transaction record.

IBM International Business Machine Corporation

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

GX21-9092 Printed in U.S.A.

Program	Programmer	Date	Punching Instruction	Page	Program Identification (75–80)
EXAMPLE PROGRAM #2			Graphic / Punch	Ø1 of __	EXMPL2

Card Electro Number

Control Card Specifications

Line	Form Type	Core Size to Compile (7–9)	Object Output (10)	Listing Options (11)	Core Size to Execute (12–14)
01	H				ØØ8

Refer to the specific System Reference Library manual for actual entries.

File Description Specification

```
02 F*********************************************
03 F*                                            *
04 F* THIS PROGRAM                               *
05 F*                                            *
06 F*    1. READS EACH TRANSACTION RECORD AND    *
07 F*       DETERMINES IF IT IS AN A, B, OR C RECORD *
08 F*                                            *
09 F*    2. THE KEY FIELD OF EACH TRANSACTION    *
10 F*       IS USED TO DIRECTLY RETRIEVE THE     *
11 F*       MATCHING MASTER RECORD.              *
12 F*                                            *
13 F*    3. 'A' CARD AMOUNTS ADD TO VALUE A. 'B' CARD *
14 F*       AMOUNTS ADD TO VALUE B. 'C' CARD AMOUNTS *
15 F*       ADD TO VALUE C.                      *
16 F*                                            *
17 F*    4. THERE IS ONE TRANSACTION FOR WHICH THERE *
18 F*       IS NO MASTER RECORD IN THE MASTER FILE. *
19 F*       IT WILL BE FLAGGED AS AN ERROR.      *
20 F*                                            *
21 F*********************************************
```

Line	Form Type	Filename	File Type	File Designation	End of File	File Format	Block Length	Record Length	Mode of Processing	Length of Key Field	Record Address Type	Type of File Organization	Overflow Indicator	Key Field Starting Location	Device	Number of Extents
22	F	TRANS	I	P	E	F	3Ø	1Ø							DISK	Ø1
23	F	PRINTER	O			F	72	72					OF		PRINTER	
24	F	MASTER	U	C		F	52	26	R	Ø5	A	I		2	DISK	Ø1
25	F															

Figure B-12 (Part 1 of 4). Specifications for EXMPL2

RPG INPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9094 Printed in U.S.A.

Program: EXAMPLE PROGRAM #2 — Programmer: — Date: — Punching Instruction: Graphic / Punch — Card Electro Number — Page Ø2 of __ — Program Identification (75–80): EXMPL2

Line (3-5)	Form Type (6)	Filename (7-14)	Sequence (15-16)	Number (17)	Option (18)	Record Identifying Indicator (19-20)	Position (21-24)	Not (25)	C/Z/D (26)	Character (27)	Field Location From (44-47)	Field Location To (48-51)	Decimal Positions (52)	Field Name (53-58)
01	I	TRANS	AA			Ø1	1		C	A				
02	I	OR				Ø2	1		C	B				
03	I	OR				Ø3	1		C	C				
04	I										2	6		KEY
05	I										7	1Ø	Ø	AMT
06	I	MASTER	AB			Ø4	1		C	M				
07	I										7	14		DESC
08	I										15	18	Ø	VALUEA
09	I										19	22	Ø	VALUEB
10	I										23	26	Ø	VALUEC
11	I													
12	I													
13	I													
14	I													
15	I													

RPG CALCULATION SPECIFICATIONS

IBM International Business Machine Corporation — Form GX21-9093 Printed in U.S.A.

Program: EXAMPLE PROGRAM #2 — Programmer: — Date: — Punching Instruction: Graphic / Punch — Card Electro Number — Page Ø3 of __ — Program Identification (75–80): EXMPL2

Line (3-5)	Form Type (6)	Control Level (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Length (49-51)	Decimal Positions (52)	Half Adjust (53)	Resulting Indicators High (54-55)	Low (56-57)	Equal (58-59)	Comments (60-74)
01	C		Ø1											
02	C	OR	Ø2											
03	C	OR	Ø3	KEY	CHAIN	MASTER					1Ø			READ MASTER
04	C		1Ø		GOTO	END								REC NOT IN FILE
05	C		Ø1	VALUEA	ADD	AMT	VALUEA							
06	C		Ø2	VALUEB	ADD	AMT	VALUEB							
07	C		Ø3	VALUEC	ADD	AMT	VALUEC							
08	C			END	TAG									
09	C													
10	C													
11	C													
12	C													
13	C													
14	C													
15	C													
16	C													

Figure B-12 (Part 2 of 4). Specifications for EXMPL2

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 Printed in U.S.A.

Program: EXAMPLE PROGRAM #2
Programmer:
Date:
Punching Instruction: Graphic / Punch
Card Electro Number
Page: Ø4 of __
Program Identification (75–80): EXMPL2

Line	Form Type	Filename (7–14)	Type (H/D/T/E) / AND/OR (14–16)	Space After (18)	Skip Before (19–20)	Output Indicators (23–31)	Field Name (32–37)	Edit Codes (38)	End Position in Output Record (40–43)	Constant or Edit Word (45–70)
01	O	PRINTER	H	1	Ø4	1P				
02	O		OR			OF				
03	O								39	'IBM SYSTEM/3'
04	O		H	2		1P				
05	O		OR			OF				
06	O								6Ø	'PAGE'
07	O								42	'SAMPLE UPDATE PROGRAM'
08	O						UDATE	Y	8	
09	O						PAGE	X	65	
10	O		H	1		1P				
11	O		OR			OF				
12	O								27	'NEW'
13	O								37	'NEW'
14	O								47	'NEW'
15	O		H	2		1P				
16	O		OR			OF				
17	O								18	'KEY DESCRIPTION'
18	O								38	'VALUE A VALUE B'
19	O								48	'VALUE C'
20	O		D	1		1Ø				

Figure B-12 (Part 3 of 4). Specifications for EXMPL2

IBM International Business Machine Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090 Printed in U.S.A.

Program: EXAMPLE PROGRAM #2
Programmer:
Date:
Punching Instruction: Graphic / Punch
Card Electro Number:
Page: Ø5 of
Program Identification: EXMPL2

Line	Form Type	Filename	Type (H/D/T/E)	Space After	Output Indicators	Field Name	Blank After (B)	End Position in Output Record	Constant or Edit Word
01	O							72	'***ERROR***'
02	O							26	'NOT IN MASTER FILE'
03	O					KEY		5	
04	O		D	1	Ø1 N1Ø				
05	O					VALUEAJ		28	
06	O					DESC		16	
07	O					KEY		5	
08	O		D	1	Ø2 N1Ø				
09	O					VALUEBJ		38	
10	O					DESC		16	
11	O					KEY		5	
12	O		D	1	Ø3 N1Ø				
13	O					VALUECJ		48	
14	O					DESC		16	
15	O					KEY		5	
16	O	MASTER	D		Ø4 N1Ø				
17	O					VALUEC	B	26	
18	O					VALUEB	B	22	
19	O					VALUEA	B	18	
20	O								

Figure B-12 (Part 4 of 4). Specifications for EXMPL2

Example Program 3

EXMPL3 must be preceded by EXMPL2. EXMPL3 reads from the indexed file, MASTER, and performs the following calculation: value A + value B - value C. If the result is negative a message is printed. Figure B-13 shows the completed specifications sheets for EXMPL3.

Control Card Specifications

This card must be present in every job. It is the first card in the source deck.

File Description Specifications

The input file for EXMPL3, MASTER, is an indexed single volume file (I in column 32 and 01 in columns 68-69). An E in column 17 indicates that the program will end when the last data record in the input file has been processed. It consists of 26-position records with a 5-position key field starting in the second record positon. A printer output file, PRINTER, with a record length of 78 is also defined on the File Description Sheet.

RPG CONTROL CARD AND FILE DESCRIPTION SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9092 Printed in U.S.A.

Program: EXAMPLE PROGRAM #3 — Page 01 of — Program Identification: EXMPL3

Control Card Specifications

```
01 H     008
```

File Description Specification

```
02 F*****************************************************
03 F*                                                   *
04 F* THIS PROGRAM                                      *
05 F*                                                   *
06 F*    1. READS EACH MASTER IN SEQUENCE AND           *
07 F*       PRINTS ONE LINE FROM IT.                    *
08 F*                                                   *
09 F*    2. PRINTS A MESSAGE FOR EACH RECORD WHICH      *
10 F*       HAS A NEGATIVE RESULT OF THE CALCULATION-   *
11 F*       VALUE A + VALUE B - VALUE C.                *
12 F*                                                   *
13 F*****************************************************
14 FMASTER   IPE F  52  26 05AI     2 DISK                         01
15 FPRINTER O     F  78  78    OF     PRINTER
```

Figure B-13 (Part 1 of 3). Specifications for EXMPL3

Input Specifications

A character M in position one of the input records will turn on record identifying indicator 01.

Calculation Specifications

The record identifying indicator 01 conditions all calculations. Values A, B, and C are accumulated (lines 03-05). The calculation, value A plus value B minus value C is performed and accumulated (lines 01, 02, and 06). If the calculation is negative the resulting indicator 22 is set on to condition the printing of a message.

Output Specifications

In these specifications, four header lines are printed, each conditioned by the first page indicator (1P) or an overflow indicator (OF).

One detail line is printed for each program cycle. One total line is also printed when the last record indicator, LR, is on.

IBM International Business Machine Corporation

RPG INPUT SPECIFICATIONS

GX21-9094 Printed in U.S.A.

Program: EXAMPLE PROGRAM #3 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page: Ø2 of | Program Identification: EXMPL3

Line	Form Type	Filename	Sequence	Record Identifying Indicator	Position	C/Z/D	Character	From	To	Decimal Positions	Field Name
01	I	MASTER	AB	Ø1	1	C	M				
02	I							2	6		KEY
03	I							7	14		DESC
04	I							15	18	Ø	VALUEA
05	I							19	22	Ø	VALUEB
06	I							23	26	Ø	VALUEC
07	I										
08	I										

IBM International Business Machine Corporation

RPG CALCULATION SPECIFICATIONS

Form GX21-9093 Printed in U.S.A.

Program: EXAMPLE PROGRAM #3 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page: Ø3 of | Program Identification: EXMPL3

Line	Form Type	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Resulting Indicators: Plus	Minus	Zero
01	C	Ø1	VALUEA	ADD	VALUEB	CALC	5	Ø			
02	C	Ø1	CALC	SUB	VALUEC	CALC				22	
03	C	Ø1	TOTA	ADD	VALUEA	TOTA	5	Ø			
04	C	Ø1	TOTB	ADD	VALUEB	TOTB	5	Ø			
05	C	Ø1	TOTC	ADD	VALUEC	TOTC	5	Ø			
06	C	Ø1	TOTCAL	ADD	CALC	TOTCAL	5	Ø			
07	C										
08	C										
09	C										

Figure B-13 (Part 2 of 3). Specifications for EXMPL3

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 Printed in U.S.A.

Program: EXAMPLE PROGRAM #3 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page: Ø4 of | Program Identification (75–80): EXMPL3

Line	Form Type	Filename (7–14)	Type / AND-OR (14–16)	Space Before (17)	Space After (18)	Skip Before (19–20)	Skip After (21–22)	Output Indicators (23–31)	Field Name (32–37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40–43)	P/B/L/R (44)	Constant or Edit Word (45–70)
01	O	PRINTER	H		1	Ø4		1P						
02	O		OR					OF						
03	O											39		'IBM SYSTEM/3'
04	O		H		2			1P						
05	O		OR					OF						
06	O								PAGE	X		7Ø		
07	O											65		'PAGE'
08	O											47		'LISTING'
09	O											39		'SAMPLE INDEXED FILE'
10	O								UDATE	Y		8		
11	O		H		1			1P						
12	O		OR					OF						
13	O											57		'TOTAL'
14	O		H		2			1P						
15	O		OR					OF						
16	O											18		'KEY DESCRIPTION'
17	O											38		'VALUE A VALUE B'
18	O											57		'VALUE C A+B-C'
19	O		D		1			Ø1						
20	O							22				71		'*NEGATIVE*'

RPG OUTPUT SPECIFICATIONS

IBM International Business Machine Corporation — GX21-9090 Printed in U.S.A.

Program: EXAMPLE PROGRAM #3 | Programmer: | Date: | Punching Instruction: Graphic / Punch | Card Electro Number | Page: Ø5 of | Program Identification (75–80): EXMPL3

Line	Form Type	Filename (7–14)	Type / AND-OR (14–16)	Space Before (17)	Space After (18)	Skip Before (19–20)	Skip After (21–22)	Output Indicators (23–31)	Field Name (32–37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40–43)	P/B/L/R (44)	Constant or Edit Word (45–70)
01	O								CALC	J	B	58		
02	O								VALUEC	K		48		
03	O								VALUEB	K		38		
04	O								VALUEA	K		28		
05	O								DESC			16		
06	O								KEY			5		
07	O		T	1			Ø1	LR						
08	O								TOTCAL	K		58		
09	O								TOTC	K		48		
10	O								TOTB	K		38		
11	O								TOTA	K		28		
12	O											18		'FINAL TOTAL'

Figure B-13 (Part 3 of 3). Specifications for EXMPL3

Appendix C. Detailed RPG II Object Program Logic

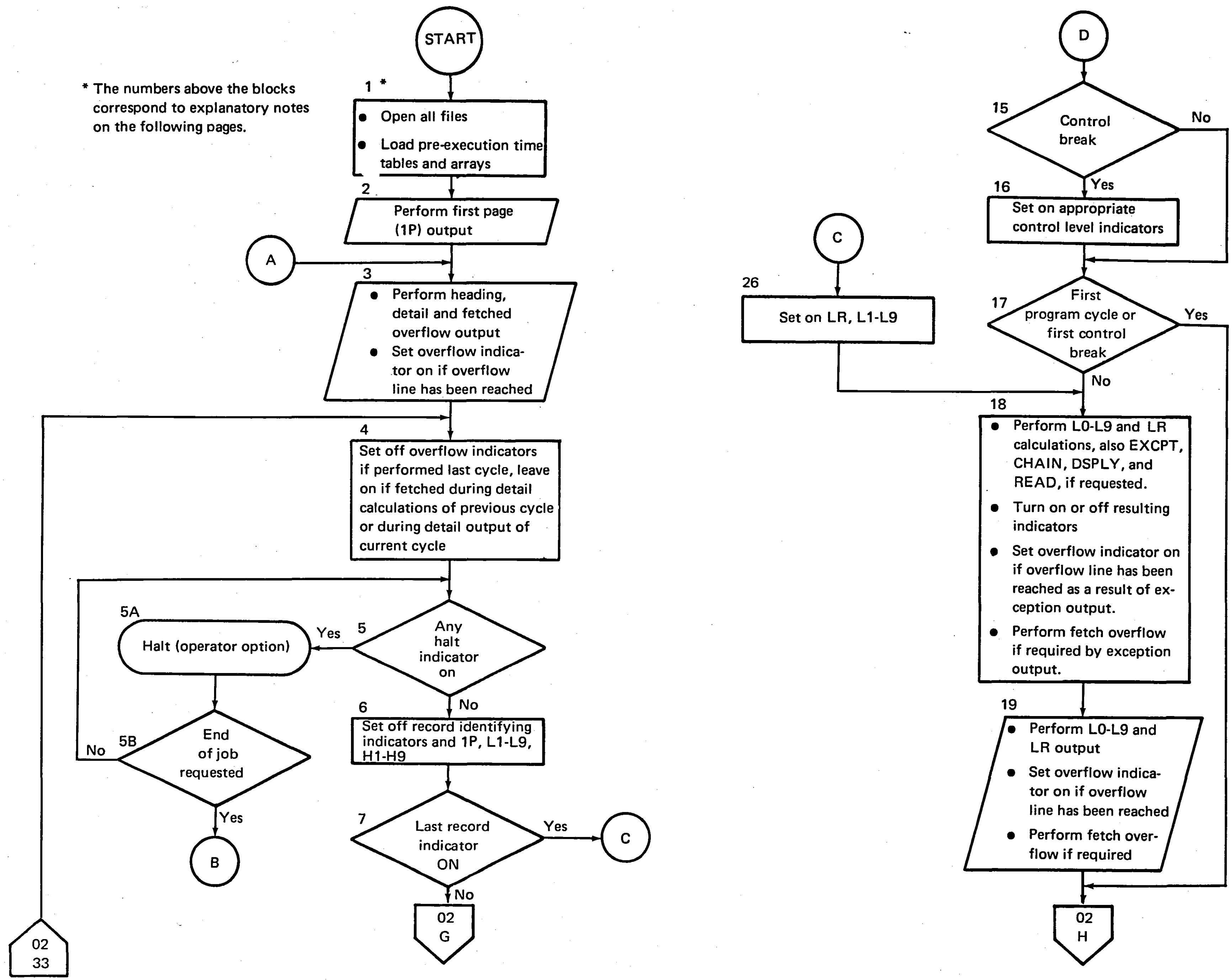


Figure C-1 (Part 1 of 2). Detailed RPG II Object Program Cycle

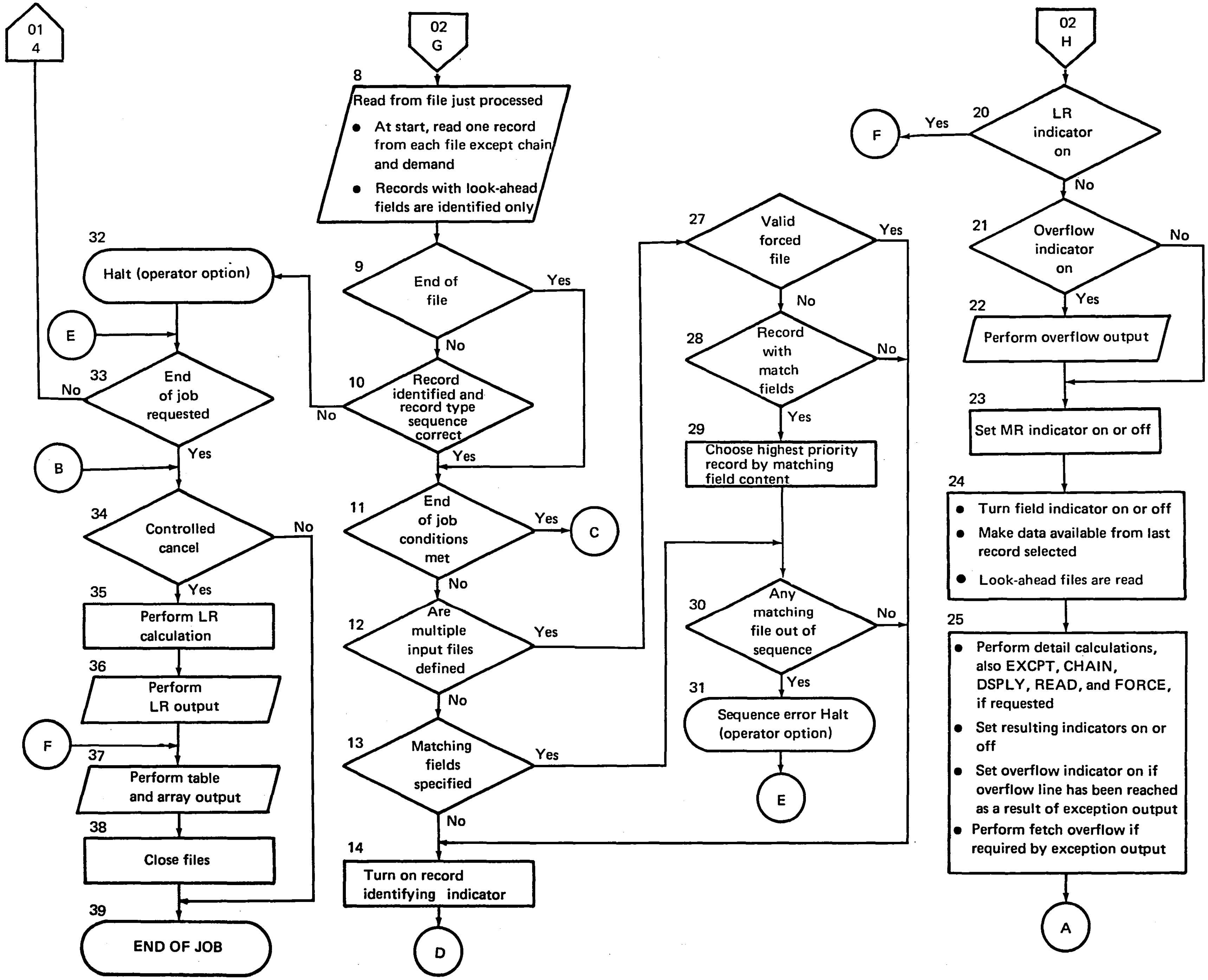


Figure C-1 (Part 2 of 2). Detailed RPG II Object Program Cycle

For each record that is processed, the RPG II object program goes through the same general cycle of operations. After a record is read, there are two different instances in time when calculation operations are performed, and records are written out. These instances in time are called *total time* and *detail time*. During total time, all total calculation operations (those conditioned by control level indicators in columns 7-8 of the Calculation Sheet) and all total output operations (those conditioned by control level indicators) are done. During detail time, all detail calculation operations (those not conditioned by control level indicators in columns 7-8) and all detail output operations are done. Total time includes steps 18 and 19 of the RPG II object program cycle; detail time includes steps 25 and 3 of the cycle.

Total calculations are performed before the information on the record selected for processing is made available. Detail calculations are performed after the information on the selected record is made available. The following discussion describes this concept in more detail.

Whenever a record is read, a check is made to determine if information in a control field (when one has been specified) is different from the control field information on the previous record. A change in the control field information indicates that all records from a particular control group have been read, and a new group is starting. When all records from a group have been read (indicated by control level indicators being turned on), operations may be done using information accumulated from all records in that group. At this time, all calculations conditioned by control level indicators in columns 7-8 are done. Total output operations are performed immediately after all total calculation operations are completed. Remember that information on the record read at the beginning of the program cycle is not used in these operations; only information from records in the previous control group is used.

Detail calculations (all calculations not conditioned by control level indicators in columns 7-8) occur after the information on the selected record has been made available. Detail calculations are used to calculate values needed each time a record is processed. They are also used to calculate totals for the current control group (if control fields are specified). Immediately after detail calculation operations are completed, detail output operations are performed.

The specific steps taken in the program cycle are shown in Figure C-1. The item numbers in the following description refer to the numbers in the figure. A program cycle begins with step 3 and continues through step 25.

1. All data files to be used by the RPG II object program are *opened*; that is, they are prepared to be processed by the object program. Pre-execution time tables and arrays are loaded before the first program cycle.

2. The object program performs all output conditioned by the 1P indicator. This output is performed only once per job and does not fall within the program cycle (steps 3 through 25).

3. The object program performs all specified heading and detail output operations whose conditions are satisfied. This includes specifications that are conditioned by the overflow indicator if the overflow routine has been fetched.

4. The object program performs a test to determine if the overflow line was encountered during detail calculations in the previous cycle or when heading and detail records were written in the current cycle. If it was, the overflow indicator turns on. Otherwise, the indicator turns off, unless the overflow routine was fetched in step 3.

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

5A. The execution of the program is stopped once for each halt indicator that is on. The operator selects one of three options: continue, controlled cancel, or immediate cancel. See *Appendix A* for an explanation of operator options.

5B. If the operator desires to continue the job, the program returns to step 5 to test for other halt indicators. If the operator selects one of the cancel options, a branch is taken to step 34.

6. All record identifying indicators and indicators 1P, L1-L9, and H1-H9 are turned off.

7. The program tests to see if the LR indicator is on. If it is, the program branches to step 26.

8. The program reads (and translates, if necessary), the next input record. At the beginning of processing, one record from each input file (except forced files and demand files) is read. If the file has look-ahead fields, it is read only on the first cycle. After that, records with look-ahead fields are identified only.

9. The 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 11.

10. If an end of file has not occurred, the program performs a test to determine if the input records are in the sequence specified for them on the Input Sheet. If the sequence is incorrect, the program branches to step 32. The program also branches to step 32 if non-sequential input records are specified and the record cannot be identified.

11. If end-of-job conditions have been met, a branch is taken to step 26. All files for which an *E* has been specified in column 17 of the File Description Sheet must be at end of file.

12. When multiple input files are used, it is necessary to select the next record to process. A branch to step 27 is made.

13. If there is only one input file, no record selection is needed. A test is made to determine if sequence checking has been requested. If so, a branch is taken to step 30.

14. The record identifying indicator specified for the current record type turns on. Data from the current record type is not available for processing until step 24.

15. If the record contains control fields, the object program performs a test to determine if a control break has occurred (the contents of the control field are not equal to the contents of a previously stored field). If a control break has not occurred or control fields are not specified, the program branches to step 17.

16. If a control break has occurred, the control level indicator reflecting the condition is turned on. All lower level indicators are also turned on.

17. If this is the first program cycle or first control break, the program bypasses all total calculation and output operations and branches to step 20.

18. All calculations conditioned by control level indicators (columns 7-8 of calculation specifications) are performed and resulting indicators are turned on or off as specified. If the LR indicator is on, calculations conditioned by LR are done after other total calculations. File translation, if specified, is done for exception output, chain, and read operations. Fetch overflow is performed if it is required by exception output. If the overflow line has been reached because of exception output, the overflow indicator is turned on.

19. All total output that is not conditioned by an overflow indicator is performed. The program performs a test to determine if an overflow condition has occurred. If an overflow condition has occurred at any time during this cycle, the overflow indicator turns on. If the LR indicator is on, output conditioned by LR is done after other total output. File translation, if specified, is done for total output. Fetch overflow is performed if required.

20. The program performs a test to determine if the last record indicator (LR) is on. If the indicator is on, the program branches to step 37.

21. The program performs a test to determine if any overflow indicators are on. If no overflow indicators are on, the program branches to step 23.

22. All output operations conditioned by a positive (no *N* preceding the indicator) overflow indicator are performed. File translation, if specified, is done for overflow output.

23. The MR indicator turns on if this is a multifile job and the record to be processed is a matching record. Otherwise, the MR indicator turns off.

24. Field indicators are turned on or off as specified. Data from the last record read and from specified look ahead fields is made available for processing.

25. Any calculations not conditioned by control level indicators (columns 7-8 of the calculation specifications) are performed, and resulting indicators are turned on or off as specified. File translation, if specified, is done for exception output, chain, and read operations. Fetch overflow is performed if it is required by exception output. If the overflow line has been reached because of exception output, the overflow indicator is turned on. Processing continues with step 3.

26. The last record indicator (LR) and all control level indicators (L1-L9) are turned on and processing continues with step 18.

27. If a file has been forced, the next record in that file is selected for processing and a branch is taken to step 14.

28. If a record with no matching fields is found in a normal input file which is not at end of file, it is selected.

29. When matching fields are specified, the normal file with the highest priority matching record field is selected. If two or more files have the equal and highest priority matching record fields, the highest priority file of those is selected. (The primary file has the highest file priority, the first specified secondary file is next, and so forth.)

30. The match field value is compared to the match field value of the last record. If it is in sequence, the record is accepted and processing continues with step 14. If the record is out of sequence, processing goes to step 31.

31. The execution of the program is stopped because a file with matching fields is out of sequence. The operator's option, indicated in step 33, is to bypass (read the next record from the same file) or cancel the job.

32. The execution of the program is stopped because of a record type sequence error or an unidentified record.

33. Step 33 tests the operator's decision either to bypass the record which causes the error condition (branch to step 4) or to cancel the job.

34. If the operator elects to terminate the job by means of a controlled cancel, steps 35 through 39 are performed. If the operator selects an immediate cancel, the job is terminated.

35. All operations conditioned by the LR indicator are done.

36. Same as 35.

37. The program writes out any tables or arrays for which a To Filename is specified on the Extension Sheet. Output tables or arrays are translated, if necessary.

38. All files used by the program are closed (final termination functions are done).

39. End of job occurs.

Type of Operation	Function of Operation	Operation Code (columns 28-32)	Control Level ***	Indicators	Factor 1	Factor 2	Result Field	Field Length	Decimal Position	Half Adjust	Resulting Indicators
Arithmetic Operations	Add Factor 2 to Factor 1.	ADD	O	O	R	R	R	O	O	O	O
	Clear Result Field and add Factor 2.	Z-ADD	O	O	B	R	R	O	O	O	O
	Subtract Factor 2 from Factor 1.	SUB	O	O	R	R	R	O	O	O	O
	Clear Result Field and subtract Factor 2.	Z-SUB	O	O	B	R	R	O	O	O	O
	Multiply Factor 1 by Factor 2.	MULT	O	O	R	R	R	O	O	O	O
	Divide Factor 1 by Factor 2.	DIV	O	O	R	R	R	O	O	O	O
	Move remainder of preceding division to a Result Field.	MVR	O	O	B	B	R	O	O	B	O
	Sum elements of an array and put sum in Result Field.	XFOOT	O	O	B	R	R	O	O	O	O
	Derive the square root of Factor 2.	SQRT	O	O	B	R	R	O	O	O	B
Move Operation	Move Factor 2 into Result Field, right justified.	MOVE	O	O	B	R	R	O	O	B	B
	Move Factor 2 into Result Field, left justified.	MOVEA	O	O	B	R	R	O	B	B	B
	Move Factor 2 into Result Field, left justified.	MOVEL	O	O	B	R	R	O	O	B	B
Move Zone Operation	Move zone from low-order position of Factor 2 to low-order position of Result Field.	MLLZO	O	O	B	R	R	O	O	B	B
	Move zone from high-order position of alphameric Factor 2 to high-order of alphameric Result Field.	MHHZO	O	O	B	R	R	O	B	B	B
	Move zone from low-order position of Factor 2 to high-order position of alphameric Result Field.	MLHZO	O	O	B	R	R	O	B	B	B
	Move zone from high-order position of alphameric Factor 2 to low-order position of Result Field.	MHLZO	O	O	B	R	R	O	O	B	B
Compare and Zone Testing Operations	Compare Factor 1 to Factor 2.	COMP	O	O	R	R	B	B	B	B	R
	Identify the zone in the leftmost position of an alphameric Result Field.	TESTZ	O	O	B	B	R	O	B	B	R
Bit Operations	Set on specified bits.	BITON	O	O	B	R	R	O	B	B	B
	Set off specified bits.	BITOF	O	O	B	R	R	O	B	B	B
	Test specified bits.	TESTB	O	O	B	R	R	O	B	B	R
Setting Indicators	Set one, two, or three specific indicators on.	SETON	O	O	B	B	B	B	B	B	R
	Set one, two, or three specific indicators off.	SETOF	O	O	B	B	B	B	B	B	R
Branching Within RPG II	Branch to another RPG II calculation specification line.	GOTO	O	O	B	R	B	B	B	B	B
	Identify the name in Factor 1 as a destination label to which GOTO may branch.	TAG	O	B	R	B	B	B	B	B	B
Branching to External Subroutines	Branch to user-written Assembler subroutine	EXIT	O	O	B	R	B	B	B	B	B
	Transfer data to user-written Assembler subroutine.	RLABL	O	B	B	B	R	O	O	B	B
Lookup Operations	Table Lookup.	LOKUP	O	O	R	R	O	O	O	B	R
	Array Lookup.	LOKUP	O	O	R	R	B	B	B	B	R
Subroutine	Beginning of the subroutine.	BEGSR	*	B	R	B	B	B	B	B	B
	End of the subroutine.	ENDSR	*	B	O	B	B	B	B	B	B
	Call to execute the subroutine.	EXSR	O	O	B	R	B	B	B	B	B

O - Optional
R - Required
B - Blank

* Columns 7-8 must have an SR entry for all subroutine lines.
** See Columns 54-59 in chapter 8 for more information.
*** The control level entry can be given for any operation code if it is an AN or OR line (see Columns 7-8, chapter 8).

Table D-1 (Part 1 of 2). Operation Codes

Type of Operation	Function of Operation	Operation Code (columns 28-32)	Control Level ***	Indicators	Factor 1	Factor 2	Result Field	Field Length	Decimal Position	Half Adjust	Resulting Indicators
Program Control	Forcing record to be read next.	FORCE	B	O	B	R	B	B	B	B	B
	Forcing output printing.	EXCPT	O	O	B	B	B	B	B	B	B
	A field is printed on the printer-keyboard and/or data is entered via the printer-keyboard into a field.	DSPLY	O	O	O	R	O	O	O	B	B
	A record is read from a demand file	READ	O	O	B	R	B	B	B	B	**
	A record is read from a disk file.	CHAIN	O	O	R	R	B	B	B	B	**
	Sets lower limits for indexed files being processed within limits.	SETLL	O	O	R	R	B	B	B	B	B
Debug Function	Aid in finding programming errors.	DEBUG	O	O	O	R	O	B	B	B	B
Time Operation	Access system time for time and date.	TIME	O	O	B	B	R	O	O	B	B

O - Optional *Columns 7-8 must have an SR entry for all subroutine lines.
R - Required **See Columns 54-59 in chapter 8 for more information.
B - Blank ***The control level entry can be given for any operation code if it is an AN or OR line (see Columns 7-8, chapter 8).

Table D-1 (Part 2 of 2). Operation Codes

Indicator	Where Specified	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 specified field. By Minus in specified 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.	Note 1
LR	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 (Cannot be SETOF) Note 2
L0 (Level Zero)	Internal	Control Level (calc.), Output Indicators	At beginning of the program.	Is never turned off by RPG.	Cannot be SETON or SETOF
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.	

Table D-2 (Part 1 of 2). Summary of Program Indicators

Indicator	Where Specified	Where Used	Turned On	Turned Off	Notes
OA, OB, OC, OD, OE, OF, OG, OV	File Description form	Indicators (calc.), Output Indicators	If the destination of a space, skip, or print operation falls within the forms overflow area.	At the end of the detail cycle.	Note 3 Note 1
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	Calculation form	Indicators (calc.), Output Indicators	By a positive balance in field, by a negative balance in field, by zero balance in field. If Factor 1 > Factor 2. If Factor 1 < Factor 2. If Factor 1 = Factor 2.	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
Look-up operation High Low Equal TESTZ operation High Low Equal Chain operation	Calculation form	Indicators (calc.) Output indicators	If table >Factor 1. If table <Factor 1. If table = Factor 1. If a C zone or & is present. If a D zone or minus (-) is present. If a C or D zone is not present. By a no record found condition.	By a record found condition.	Note 1
1P (First Page)	Internal	Output Indicators	At beginning of processing before any input records are read.	Before the first detail record 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 overflow 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.

Note: When a program is doing multiple reads from one or several demand files during the same RPG II cycle, the record identifying indicators assigned to the file(s) remain on throughout the cycle if the previous READ operations were executed successfully.

When chaining to one or more files during the same RPG II cycle, record identifying indicators assigned to the chained file(s) remain on throughout the cycle if the previous operations were executed successfully.

Table D-2 (Part 2 of 2). Summary of Program Indicators

Table D-3. Valid Indicators

Indicators	File Description Specifications		Input Specifications				Calculation Specifications			Output-Format Specifications
	Overflow Indicator (33-34)	File Conditioning (71-72)	[1] Record Identifying Indicator (19-20)	Control Level (59-60)	[1] Field Record Relation (63-64)	Field Indicator (65-70)	Control Level Indicator (7-8)	Conditioning Indicator (9-17)	Resulting Indicator (54-59)	Conditioning Indicator (23-31)
01-99			X		X	X		X	X	X
H1-H9			X		X	X		X	X	X
1P										X[3]
MR					X[2]			X		X
OA-OG,OV	X							X	X	X[4]
L0							X			X
L1-L9			X	X	X[2]		X	X	X	X
LR			X				X	X	X	X
U1-U8		X[5]			X			X		X

Note: X denotes the indicators that may be used.

1 Not valid on look-ahead fields.

2 When field named is not a match field or a control field.

3 Only for detail or heading lines.

4 Cannot condition an exception line, but may condition fields within the exception record.

5 Not valid for table input files.

Characters grouped by equal zones

	Character	96 Column Card Code
GROUP 1	Blank	No punches
	¢	B-A-8-2
	.	B-A-8-2-1
	<	B-A-8-4
	(	B-A-8-4-1
	+	B-A-8-4-2
	|	B-A-8-4-2-1
GROUP 2	!	B-8-2
	$	B-8-2-1
	*	B-8-4
	)	B-8-4-1
	;	B-8-4-2
	¬	B-8-4-2-1
GROUP 3	/	A-1
	, (comma)	A-8-2-1
	%	A-8-4
	—	A-8-4-1
	>	A-8-4-2
	?	A-8-4-2-1
GROUP 4	:	8-2
	#	8-2-1
	@	8-4
	' (quote)	8-4-1
	=	8-4-2
	"	8-4-2-1
GROUP 5	&	A-8-2
	A	B-A-1
	B	B-A-2
	C	B-A-2-1
	D	B-A-4
	E	B-A-4-1
	F	B-A-4-2
	G	B-A-4-2-1
	H	B-A-8
	I	B-A-8-1
GROUP 6	-	B
	}	B-A
	J	B-1
	K	B-2
	L	B-2-1
	M	B-4
	N	B-4-1
	O	B-4-2
	P	B-4-2-1
	Q	B-8
	R	B-8-1
GROUP 7	S	A-2
	T	A-2-1
	U	A-4
	V	A-4-1
	W	A-4-2
	X	A-4-2-1
	Y	A-8
	Z	A-8-1
GROUP 8	0	A
	1	1
	2	2
	3	2-1
	4	4
	5	4-1
	6	4-2
	7	4-2-1
	8	8
	9	8-1

Characters grouped by equal digits

	Character	96 Column Card Code
GROUP 1	Blank	No punches
	{	A-8-2
	-	B
	}	B-A
	0	A
GROUP 2	/	A-1
	A	B-A-1
	J	B-1
	1	1
GROUP 3	B	B-A-2
	K	B-2
	S	A-2
	2	2
GROUP 4	C	B-A-2-1
	L	B-2-1
	T	A-2-1
	3	2-1
GROUP 5	D	B-A-4
	M	B-4
	U	A-4
	4	4
GROUP 6	E	B-A-4-1
	N	B-4-1
	V	A-4-1
	5	4-1
GROUP 7	F	B-A-4-2
	O	B-4-2
	W	A-4-2
	6	4-2
GROUP 8	G	B-A-4-2-1
	P	B-4-2-1
	X	A-4-2-1
	7	4-2-1
GROUP 9	H	B-A-8
	Q	B-8
	Y	A-8
	8	8
GROUP 10	I	B-A-8-1
	R	B-8-1
	Z	A-8-1
	9	8-1
GROUP 11	¢	B-A-8-2
	!	B-8-2
	:	8-2
GROUP 12	.	B-A-8-2-1
	$	B-8-2-1
	, (comma)	A-8-2-1
	#	8-2-1
GROUP 13	<	B-A-8-4
	*	B-8-4
	%	A-8-4
	@	8-4
GROUP 14	(	B-A-8-4-1
	)	B-8-4-1
	—	A-8-4-1
	' (quote)	8-4-1
GROUP 15	+	B-A-8-4-2
	;	B-8-4-2
	>	A-8-4-2
	=	8-4-2
GROUP 16	|	B-A-8-4-2-1
	¬	B-8-4-2-1
	?	A-8-4-2-1
	"	8-4-2-1

Table D-4. Character Grouping by Zone and Digit

Collating Sequence	Character	Hexadecimal Equivalent
1	Blank	40
2	¢	4A
3	.	4B
4	<	4C
5	(	4D
6	+	4E
7	|	4F
8	&	50
9	!	5A
10	$	5B
11	*	5C
12	)	5D
13	;	5E
14	¬	5F
15	- (minus)	60
16	/	61
17	,	6B
18	%	6C
19	_ (underscore)	6D
20	>	6E
21	?	6F
22	:	7A
23	#	7B
24	@	7C
25	'	7D
26	=	7E
27	"	7F
28	A	C1
29	B	C2
30	C	C3
31	D	C4
32	E	C5
33	F	C6
34	G	C7
35	H	C8
36	I	C9
37	}	D0
38	J	D1
39	K	D2
40	L	D3
41	M	D4
42	N	D5
43	O	D6
44	P	D7
45	Q	D8
46	R	D9
47	S	E2
48	T	E3
49	U	E4
50	V	E5
51	W	E6
52	X	E7
53	Y	E8
54	Z	E9
55	0	F0
56	1	F1
57	2	F2
58	3	F3
59	4	F4
60	5	F5
61	6	F6
62	7	F7
63	8	F8
64	9	F9

Table D-5. Normal Collating Sequence and Hexadecimal Equivalents of Characters

Edit Code	Commas	Decimal Point	Sign For Negative Balance			Print Out On Zero Balance *			Zero Suppress
			No Sign	CR	– (Minus)	Domestic and United Kingdom	World Trade /	World Trade J	
1	Yes	Yes	No Sign			.00 or 0	,00 or 0	0,00 or 0	Yes
2	Yes	Yes	No Sign			Blanks	Blanks	Blanks	Yes
3		Yes	No Sign			.00 or 0	,00 or 0	0,00 or 0	Yes
4		Yes	No Sign			Blanks	Blanks	Blanks	Yes
A	Yes	Yes		CR		.00 or 0	,00 or 0	0,00 or 0	Yes
B	Yes	Yes		CR		Blanks	Blanks	Blanks	Yes
C		Yes		CR		.00 or 0	,00 or 0	0,00 or 0	Yes
D		Yes		CR		Blanks	Blanks	Blanks	Yes
J	Yes	Yes			–	.00 or 0	,00 or 0	0,00 or 0	Yes
K	Yes	Yes			–	Blanks	Blanks	Blanks	Yes
L		Yes			–	.00 or 0	,00 or 0	0,00 or 0	Yes
M		Yes			–	Blanks	Blanks	Blanks	Yes
X **									
Y ***									Yes
Z									Yes

* Zero balances for the World Trade format are written in two ways, depending on the entry made in column 21 of the control card specifications.

** The X code performs no editing.

*** The Y code suppresses the leftmost zero only. The Y code edits a three to six digit field according to the following pattern:

nn/n
nn/nn
nn/nn/n
nn/nn/nn

If a data field of six digits is packed on disk and the Y edit code is used with the data field, an error will occur. To solve this problem, move the data field to another field.

Table D-6. Edit Codes

Record Length	Block Length Computed by RPG II		Input/Output Area Allocated by RPG II		Number of Records per Block	
*	Group A	Group B	Group A	Group B	Group A	Group B
32	256	256	256	256	8	8
60	240	240	256	512	4	4
64	256	256	256	256	4	4
80	240	240	256	512	3	3
96	192	192	256	512	2	2
128	256	256	256	256	2	2
256	256	256	256	256	1	1
512	512	512	512	512	1	1

* Files in Group B can require a larger input/output area than files in Group A.

Group A	*Group B*
Consecutive Output	Consecutive Update
Consecutive Input	Indexed Update
Indexed Input Processed Sequentially	Indexed File Processed Randomly
Indexed Output	Direct File
	Random Input by ADDROUT
	Random Update by ADDROUT

Note: Results are the same for DISK and DISK45.

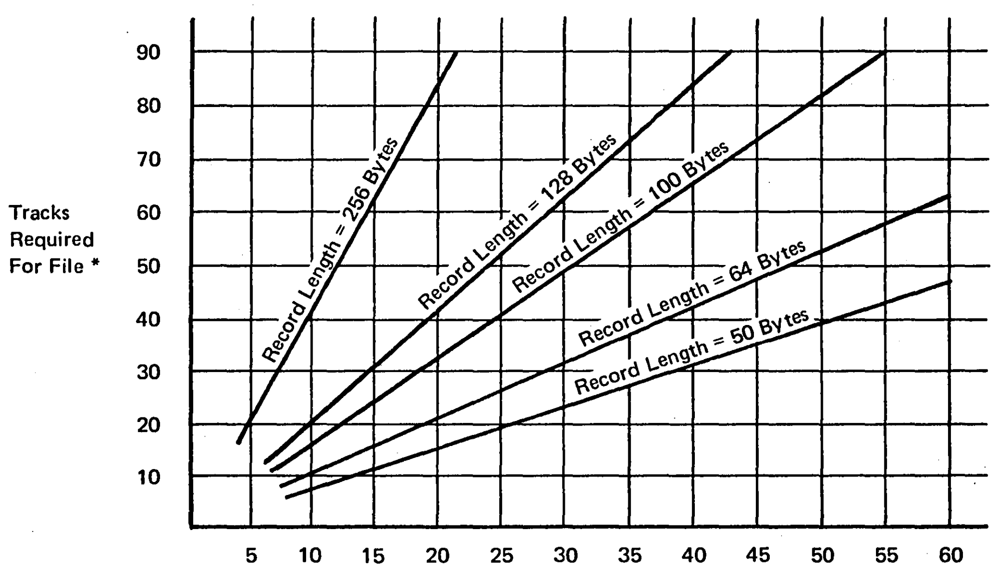
Table D-7. Block Length and Size of Input/Output Area Computed by RPG II for Disk Files

Key Length	2	5	8	10	15	20
11	65	156	247	312	455	611
10	60	132	216	264	396	528
9	44	110	176	220	330	440
8	40	100	150	190	280	370
7	36	81	126	153	225	306
6	24	64	96	120	184	240
5	21	49	77	98	140	189

Number of Records (thousands)
(add one key length for 5445)

* The bytes of main storage required for the Core Index is based on one Core Index entry per track of file index entries (single volume files only).

Table D-8. Bytes of Main Storage Required for Core Index – 5444



Number of Records in File (hundreds) – 5444

* Record storage area only; index area for indexed file is not included.

Table D-9. File Allocation

Appendix E. RPG II Error Messages

This appendix lists the RPG II and BSCA error messages for the IBM System/3 Model 10, Model 12, and Model 15. For each error message, this appendix includes an explanation of the message, a description of any action the system takes, and suggested responses you can give to restart the system or to avoid the message when the job is run again. For information on other types of messages, see the appropriate halt guide listed under *Related Publications* in the Preface.

References in this section to job refer to job step or simply program for the Model 15.

MESSAGE FORMAT

Each message is preceded by an identification code. This code consists of three parts (Figure E-1):

1. Program identification RG (for RPG II).
2. Message number.
3. Significance code:

   W (Warning) – Warning that an abnormal condition exists. Corrective action is required only if condition is unintentional.

   T (Terminal) – An error condition exists that requires corrective action before the system can continue executing the program.

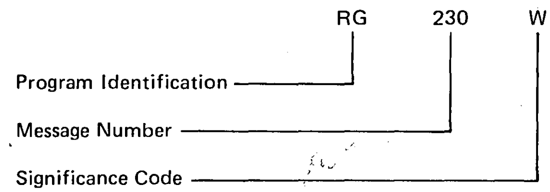


Figure E-1. Message Format

RG001–NO SOURCE

Code:	T–Terminal
Explanation:	You did not supply a source program for this job.
System Action:	The job is terminated.
User Response:	You must supply a source program and resubmit the job.

RG002–INVALID OBJECT OUTPUT ENTRY IN COL 10, ASSUME BLANK

Code:	W–Warning
Specification Type:	H
Explanation:	The entry in column 10 of your header line is not C, D, P, or blank.
System Action:	Blank is assumed and the object program is temporarily written in the object library.
User Response:	If this assumption was wrong, make the proper entry (C, D, or P) in column 10 and resubmit the job.

RG003–INVALID LISTING OPTION IN COL 11, ASSUME BLANK

Code: W–Warning
Specification Type: H
Explanation: The entry in column 11 is neither B nor blank.
System Action: Blank is assumed. Therefore, a source program listing and the object program are produced.
User Response: If this assumption was wrong, make the proper entry in column 11 and resubmit the job.

RG004–INVALID OR BLANK STORAGE SIZE TO EXECUTE ENTRY IN COL 12-14, ASSUME SYSTEM SIZE

Code: W–Warning
Specification Type: H
Explanation: Columns 12-14 are blank or they contain an entry which is greater than 061.
System Action: The size of your system is assumed.
User Response: If this assumption was wrong, make the proper entry in columns 12-14 and resubmit the job.

RG005–INVALID DEBUG CODE IN COL 15, ASSUME BLANK

Code: W–Warning
Specification Type: H
Explanation: The entry in column 15 is neither 1 nor blank.
System Action: Blank is assumed
User Response: If this assumption was wrong, make the proper entry in column 15 and resubmit the job.

RG006–INVALID ENTRY IN COL 16, ASSUME BLANK

Code: W–Warning
Specification Type: H
Explanation: This column is not used; it must be left blank.
System Action: Blank is assumed.
User Response: To avoid this message the next time the job is run, leave column 16 blank.

RG007–ENTRY IN COL 12-14 NOT A MULTIPLE OF 2K

Code: W–Warning
Specification Type: H
Explanation: On Model 15 only, main storage is allocated in 2K increments.
System Action: Entry is rounded up to the next 2K increment.
User Response: If this assumption is incorrect, make the proper entry in Columns 12-14 and resubmit the job.

RG008–INVALID ENTRY IN COL 37 AND/OR COLUMNS 52-54, ASSUME BLANKS.

Code: W–Warning
Specification Type: H
Explanation: The entry in column 37 of your header line is not I, B, or blank (B is not supported by 5704-RG2).
System Action: Blank is assumed.
User Response: If this assumption was wrong, make the proper entry (I or B) in column 37 and resubmit the job.

RG009–INVALID ENTRY IN COL 49, ASSUME BLANK

Code: W–Warning
Specification Type: H
Explanation: Column 49 is not used.
System Action: Blank is assumed.
User Response: To avoid this message the next time the job is run, leave column 49 blank.

RG011–INVALID ENTRY IN COL 17-20, ASSUME BLANK

Code: W–Warning
Specification Type: H
Explanation: These columns are not used; they must be left blank.
System Action: Blanks are assumed.
User Response: To avoid this message the next time the job is run, leave columns 17-20 blank.

RG012–INVALID INVERTED PRINT ENTRY IN COL 21, ASSUME BLANK

Code:	W–Warning
Specification Type:	H
Explanation:	The entry in column 21 of your header line is not I, D, J, or blank.
System Action:	Blank is assumed; the job continues.
User Response:	If this assumption was wrong, make the proper entry in column 21 and resubmit the job.

RG013–INVALID ENTRIES IN COL 22-25, ASSUME BLANKS

Code:	W–Warning
Specification Type:	H
Explanation:	These columns are not used; they must be left blank.
System Action:	Blanks are assumed.
User Response:	To avoid this message the next time the job is run, leave columns 22-25 blank.

RG014–INVALID ALTERNATE COLLATING SEQUENCE ENTRY IN COL 26, ASSUME S

Code:	W–Warning
Specification Type:	H
Explanation:	The entry in column 26 of your header line is neither blank nor S.
System Action:	The entry is assumed to be S. The S entry alters the normal collating sequence.
User Response:	If this assumption was wrong, make the proper entry in column 26 and resubmit the job.

RG015–INVALID ENTRIES IN COL 27-36 AND/OR 38-40, ASSUME BLANKS

Code:	W–Warning
Specification Type:	H
Explanation:	These columns are not used; they must be left blank.
System Action:	Blanks are assumed.
User Response:	To avoid this message the next time the job is run, leave columns 27-36 and 38-40 blank.

RG016–INVALID 1P REPEAT ENTRY IN COL 41, ASSUME 1

Code:	W–Warning
Specification Type:	H
Explanation:	Column 41 of your header line is neither 1 nor blank.
System Action:	1 is assumed; the job continues.
User Response:	If this assumption was wrong, make the proper entry in column 41 and resubmit the job.

RG017–INVALID ENTRY IN COL 42, ASSUME BLANK

Code:	W–Warning
Specification Type:	H
Explanation:	This column is not used; it must be left blank.
System Action:	Blank is assumed; the job continues.
User Response:	To avoid this message the next time the job is run, leave column 42 blank.

RG018–INVALID FILE TRANSLATION ENTRY IN COL 43, ASSUME F

Code:	W–Warning
Specification Type:	H
Explanation:	The entry in column 43 of your header line is neither F nor blank.
System Action:	F is assumed; the job continues.
User Response:	If this assumption was wrong, make the proper entry in column 43 and resubmit the job.

RG019–INVALID ZERO SUPPRESS ENTRY IN COL 44, ASSUME 1

Code:	W–Warning
Specification Type:	H
Explanation:	The entry in column 44 of your header line is neither 1 nor blank.
System Action:	1 is assumed and the job continues.
User Response:	If this assumption was wrong, make the proper entry in column 44 and resubmit the job.

RG020–INVALID NON-PRINTABLE CHARACTER ENTRY IN COL 45; ASSUME 1

Code:	W–Warning
Specification Type:	H
Explanation:	Column 45 must be blank or 1. A blank entry provides a halt on nonprintable characters and a 1 does not.
System Action:	1 is assumed.
User Response:	To avoid this message the next time, make the proper entry in column 45 and resubmit the job.

RG021–INVALID ENTRIES IN COL 46-47, 50-51, OR 55-74, ASSUME BLANKS

Code:	W–Warning
Specification Type:	H
Explanation:	These columns are not used; they must be left blank.
System Action:	Blanks are assumed.
User Response:	To avoid this message the next time this job is run, leave columns 46-47, 50-51, and 55-74 blank.

RG022–INVALID ENTRY IN COL 6 OR SPEC TYPE OUT OF SEQUENCE

Code:	T–Terminal
Specification Type:	F, E, L, T, I, C, or O
Explanation:	The entry in column 6 must be F, E, L, T, I, C, or O and the specifications must be in the proper sequence.
System Action:	The job is terminated and the entire specification line is ignored.
User Response:	Check to see which specifications contain an invalid entry in column 6 or are out of the sequence required in the source program. (Valid entries for column 6 are H, F, E, L, T, I, C, or O, and records must be in that order.) Resubmit the job.

RG023–INVALID FILENAME IN COL 7-14

Code:	T–Terminal
Specification Type:	F, I, L, T, or O
Explanation:	Filename specified in columns 7-14 was invalid.
System Action:	The job is terminated and the entire specification line is ignored.
User Response:	Check your source specifications to determine which have an invalid filename. Make the proper entry and resubmit the job.

RG024–FILENAME PREVIOUSLY DEFINED IN COL 7-14

Code:	T–Terminal
Specification Type:	F
Explanation:	This filename is not unique.
System Action:	The job is terminated and the entire specification line is ignored.
User Response:	Assign a unique name to the file. Resubmit the job.

RG025–INVALID DEVICE NAME IN COL 40-46, ASSUME DISK

Code:	T–terminal
Specification Type:	F
Explanation:	The entry in columns 40-46 is not a valid device name.
System Action:	DISK is assumed, but the job is terminated.
User Response:	Enter the proper device name in columns 40-46 of the File Description sheet and resubmit the job.

RG026–INVALID FILE TYPE ENTRY IN COL 15, ASSUME DEFAULT FOR DEVICE

Code:	T–Terminal
Specification Type:	F
Explanation:	The file type entry in column 15 is not I, O, C, U, D, or the entry is not valid for the device specified.
System Action:	O is assumed for files assigned to PRINTER and PRINTR2; I is assumed for files assigned to TAPE and CONSOLE; C is assumed for files assigned to MFCU1, MFCU2, 1442, MFCM1, MFCM2 SPECIAL, DISKET or BSCA. U is assumed for files assigned to DISK, DISK40, and DISK45. The job is terminated.
User Response:	Enter the proper file type in column 15 and resubmit the job.

RG028–FILE DESIGNATION IN COL 16 IS INVALID FOR EITHER FILE TYPE OR DEVICE. ASSUME SECONDARY.

Code:	W–Warning
Specification Type:	F
Explanation:	The entry in column 16 is not valid for an input, combined or update file.
System Action:	S is assumed and the job continues.
User Response:	If this assumption was wrong, make the proper entry in column 16 and resubmit the job.

RG029–DEVICE NAME IN COL 40-46 BLANK AND COL 15 NOT I OR 0, ASSUME DISK

Code:	T–Terminal
Specification Type:	F
Explanation:	If device independent input or output file is desired (Model 15 only), put an I or 0 in column 15; otherwise enter a proper device name in columns 40-46 of the file description specification.
System Action:	DISK is assumed, but the job is terminated.
User Response:	Make corrections and resubmit the job.

RG030–FILE DESIGNATION ENTRY IN COL 16 INVALID FOR OUTPUT OR DISPLAY FILE, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Column 16 must be blank for output files and display files (O or D in column 15).
System Action:	Blank is assumed.
User Response:	To avoid this message the next time this job is run, make a blank entry in column 16.

RG032–NO PRIMARY FILE SPECIFIED IN COL 16, ASSUME FIRST SECONDARY FILE AS PRIMARY

Code:	W–Warning
Specification Type:	F
Explanation:	No primary file was specified (P in column 16) of your file description specifications.
System Action:	The first secondary file is assumed to be the primary file.
User Response:	If this assumption was wrong, make the proper entry in column 16 and resubmit the job.

RG033–NO PRIMARY OR SECONDARY FILE SPECIFIED IN COL 16 OR NO FILE DESCRIPTION SPEC FOUND

Code:	T–Terminal
Specification Type:	F
Explanation:	No primary or secondary file was specified (P or S in column 16 of the file description specifications) or no file description specifications were supplied.
System Action:	The job is terminated.
User Response:	Supply file description specifications or define an input file and resubmit the job.

RG034–MULTIPLE PRIMARY FILES DEFINED IN COL 16, ASSUME SECONDARY

Code:	W–Warning
Specification Type:	F
Explanation:	More than one primary file (P in column 16) was defined in your file description specifications.
System Action:	All primary files except the first one are assumed to be secondary.
User Response:	If this assumption was wrong, make the proper entry (S in column 16). Resubmit the job.

RG036–INVALID END OF FILE ENTRY IN COL 17, ASSUME E FOR INPUT FILE TYPE WITHOUT RANDOM PROCESSING

Code:	W–Warning
Specification Type:	F
Explanation:	The entry in column 17 of your file description specifications is neither E nor blank.
System Action:	E is assumed for input files not processed randomly; blank is assumed for all other files.
User Response:	If this assumption was wrong, make the proper end-of-file entry in column 17. Resubmit the job.

RG037–INVALID ENTRY IN COL 19

Code:	W–Warning
Specification Type:	F
Explanation:	Column 19 may contain a D, F, or V for tape files only. For all other devices, column 19 must contain an F.
System Action:	D is assumed for variable length ASCII tape files when column 19 contains a V. V is assumed for variable length EBCDIC tape files when column 19 contains a D. F is assumed for all other file types and invalid entries for tape files.
User Response:	If this assumption is wrong, make the proper entry in column 19 and resubmit the job.

RG038–END OF FILE ENTRY IN COL 17 INVALID FOR FILE TYPE

Code:	W–Warning
Specification Type:	F
Explanation:	Column 17 must be blank for output, demand, table, and display files.
System Action:	Blank is assumed.
User Response:	To avoid this message the next time this job is run, leave column 17 blank.

RG039–INVALID SEQUENCE ENTRY IN COL 18, ASSUME PREVIOUS ENTRY

Code:	W–Warning
Specification Type:	F
Explanation:	The entry in column 18 is not A, D, or blank.
System Action:	The entry in column 18 from the previous specification line is assumed.
User Response:	If this assumption was wrong, make the proper entry in column 18.

RG040–ENTRY IN COL 18 INVALID FOR TYPE OF FILE OR MODE OF PROCESSING, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Column 18 must be blank for demand files, output files, record address files, display files, and for any files processed randomly.
System Action:	Blank is assumed.
User Response:	To avoid this message the next time this job is run, leave column 18 blank.

RG041–INVALID RECORD LENGTH ENTRY IN COL 24-27, ASSUME DEFAULT FOR DEVICE

Code:	W–Warning
Specification Type:	F
Explanation:	Incorrect record length was specified in columns 24-27.
System Action:	The maximum record length for the device is assumed, except DISK, DISK40, and DISK45 for which 256 is assumed.
User Response:	If this assumption was wrong, make the proper record length entry and resubmit the job.

RG042–INVALID ENTRIES IN COL 20-23, ASSUME RECORD LENGTH

Code:	W–Warning
Specification Type:	F
Explanation:	The entry in columns 20-23 is neither equal to nor a multiple of the record length specified in columns 24-27. For tape files with variable length records, the block size should be at least the record length plus eight (8).
System Action:	The record length is assumed for all devices, except tape with variable length records for which record length plus eight (8) is assumed.
User Response:	If the assumption was wrong, make the block length a multiple of record length and resubmit the job. If the record length is variable on tape, add eight to the block size.

RG043–DUAL I/O ENTRY IN COL 32 INVALID FOR TYPE OF FILE OR MODE OF PROCESSING, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	(1) Dual I/O (1-9 in column 32) cannot be specified for combined, demand, table, and update files, or for any file processed randomly. (2) Neither can dual I/O be specified if shared I/O has been specified (column 48 of control card specifications). T in column 32 is invalid for a table file.
System Action:	Blank is assumed.
User Response:	If this assumption was wrong, make the proper dual I/O entry and resubmit the job.

RG044–INVALID ENTRY IN COL 32, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	The entry in column 32 was not 1-9, I, T, or blank, or entry is not allowed for device.
System Action:	Blank is assumed.
User Response:	If the assumption was wrong, make the proper entry and resubmit the job.

RG045–OVERFLOW INDICATOR IN COL 33-34 PREVIOUSLY ASSIGNED, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F
Explanation:	The same overflow indicator was assigned to more than one file.
System Action:	Blank is assumed, but the job is terminated.
User Response:	Assign different overflow indicators to each file being described.

RG046–INVALID OVERFLOW INDICATOR IN COL 33-34, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F
Explanation:	The entry in columns 33-34 was not OA-OG, or OV.
System Action:	Blank is assumed, but the job is terminated.
User Response:	Enter OA-OG or OV in columns 33-34 if you want to specify overflow for this file, if not, leave columns 33-34 blank. Resubmit the job.

RG047–OVERFLOW INDICATOR IN COL 33-34 INVALID FOR DEVICE, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	The overflow indicator in columns 33-34 was not assigned to a printer file.
System Action:	Blank is assumed.
User Response:	To avoid this message the next time this job is run, assign overflow indicators to printer files.

RG048–INVALID OR BLANK EXTENSION CODE ENTRY IN COL 39 FOR TABLE OR RECORD ADDRESS FILE, ASSUME E

Code:	W–Warning
Specification Type:	F
Explanation:	The extension code in column 39 was not E for a table or record address file.
System Action:	E is assumed and the job continues.
User Response:	To avoid this message the next time this job is run, enter E in column 39.

RG049–INVALID EXTENSION CODE IN COL 39

Code:	W–Warning
Specification Type:	F
Explanation:	The entry in column 39 is neither L nor blank for output files assigned to the printer.
System Action:	L is assumed and the job continues.
User Response:	If this assumption was wrong, make the entry in column 39 blank. Resubmit the job.

RG051–EXTENSION CODE ENTRY IN COL 39 INVALID WITH DEVICE OR P, S, C, OR D IN COL 16, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Column 39 can only be used with table, record address or printer files.
System Action:	Blank is assumed and the job continues.
User Response:	If this assumption was wrong, make the proper entry in column 39 and resubmit the job.

RG052–DEVICE IN COL 40-46 PREVIOUSLY ASSIGNED TO OUTPUT OR NON-TABLE INPUT FILE

Code:	T–Terminal
Specification Type:	F
Explanation:	The device name in columns 40-46 was assigned to more than one output or non-table input file.
System Action:	The job is terminated and the entire specification line is ignored. This condition may cause other errors to be generated.
User Response:	Make the device name entry in columns 40-46 unique for each output or non-table input file (except those assigned to disk, tape, and console). Resubmit the job.

RG053–INVALID ENTRIES IN COL 47-52, ASSUME BLANKS

Code:	W–Warning
Specification Type:	F
Explanation:	These columns are not used; they must be left blank.
System Action:	Blanks are assumed and the job continues.
User Response:	To avoid this message the next time this job is run, leave columns 47-52 blank.

RG055–ENTRIES IN COL 71-72 INVALID FOR TABLE FILE, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Columns 71-72 must be left blank for table files, since table files cannot be conditioned by U1-U8.
System Action:	Blanks are assumed and the job continues.
User Response:	To avoid this message the next time the job is run, leave columns 71-72 blank for table files.

RG057–INVALID FILE CONDITIONING ENTRIES IN COL 71-72

Code:	T–Terminal
Specification Type:	F
Explanation:	Columns 71-72 of your file description specification are not blank nor do they contain one of the external indicators (U1-U8).
System Action:	The job is terminated.
User Response:	Leave columns 71-72 blank or enter one of the external indicators (U1-U8). Resubmit the job.

RG058–INVALID ENTRIES IN COLS 67 AND/OR 73-74, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Columns 67 and 73-74 are not used; they must be left blank.
System Action:	Blanks are assumed and the job continues.
User Response:	To avoid this message when the job is run again, leave columns 67 and 73-74 blank.

RG060–INVALID ENTRY IN COLUMN 48, ASSUME BLANK

Code:	W–Warning
Specification Type:	H
Explanation:	The shared I/O entry in column 48 is not 1, E, or blank. A 1 is invalid in Model 15 RPGII. Program 5704-RG2 allows an E in column 48 for external disk I/O buffers.
System Action:	Blank is assumed.
User Response:	If this assumption was wrong, make the proper entry in column 48 and resubmit the job.

RG061–INVALID ENTRIES IN COL 7-10, ASSUME BLANK

Code:	W–Warning
Specification Type:	E
Explanation:	Columns 7-10 are not used; they must be left blank.
System Action:	Blanks are assumed and the job continues.
User Response:	To avoid this message when the job is run again, leave columns 7-10 blank.

RG062–INVALID OR UNDEFINED FROM FILENAME ENTRY IN COL 11-18

Code:	T–Terminal
Specification Type:	E
Explanation:	The From Filename in columns 11-18 of your extension specifications is invalid or has not been previously defined in file description specifications. (The From Filename must start in column 11.)
System Action:	The job is terminated.
User Response:	Make the proper From Filename entry in columns 11-18. If columns 11-18 already contain a valid entry, check to make sure that the file was defined in your file description specifications. Resubmit the job.

RG063–TYPE OF FILE INVALID FOR FROM FILE-NAME ENTRY IN COL 11-18

Code:	T–Terminal
Specification Type:	E
Explanation:	The From Filename entry does not refer to a table or record address input file.
System Action:	The job is terminated.
User Response:	Make sure the entry in columns 11-18 refers to a table or record address input file. Resubmit the job.

RG064–INVALID OR UNDEFINED TO FILENAME IN COL 19-26

Code:	T–Terminal
Specification Type:	E
Explanation:	The To Filename in columns 19-26 of your extension specifications is invalid or has not been defined in file description specifications. (The To Filename must start in column 19.)
System Action:	The job is terminated.
User Response:	Make the proper To Filename entry in columns 19-26. If columns 19-26 already contain a valid entry, check to make sure the filename has been previously defined in your file description specifications. Resubmit the job.

RG065–TYPE OF FILE INVALID OR INCORRECT FOR TO FILENAME ENTRY IN COL 19-26

Code:	T–Terminal
Specification Type:	E
Explanation:	The To Filename entry does not refer to an output file, or to a file processed by a record address file.
System Action:	The job is terminated.
User Response:	Be sure the entry in columns 19-26 refers to an output file or to a file processed by a record address file. Resubmit the job.

RG067–INVALID TABLE OR ARRAY NAME IN COL 27-32

Code:	T–Terminal
Specification Type:	E
Explanation:	The table or array name in columns 27-32 was not specified properly. A table or array name must start in column 27. A table name must begin with TAB; an array name must not begin with TAB.
System Action:	The job is terminated.
User Response:	Make the proper table or array name entry in columns 27-32 and resubmit the job.

RG068–INVALID OR MISSING NUMBER OF ENTRIES PER RECORD ENTRY IN COL 33-35, ASSUME 08

Code:	T–Terminal
Specification Type:	E
Explanation:	The entry in columns 33-35 is missing on a specification line which has a From Filename in columns 11-18, or it is not a one to three-digit number (1-999).
System Action:	08 is assumed, but the job is terminated.
User Response:	Define the number of entries per record. To do so, make a numeric entry (1-999) in columns 33-35. Resubmit the job.

RG070–INVALID OR MISSING NUMBER OF ENTRIES PER TABLE OR ARRAY IN COL 36-39, ASSUME 05

Code:	T–Terminal
Specification Type:	E
Explanation:	The entry in columns 36-39 is missing or it is not a one to four-digit number (1-9999).
System Action:	05 is assumed, but the job is terminated.
User Response:	Define the maximum number of entries per table or array. To do so, make a numeric entry (1-9999) in columns 36-39. Resubmit the job.

RG071–NO. OF ENTRIES PER RECORD IN COL 33-35 EXCEEDS NO. OF ENTRIES PER TABLE/ARRAY IN COLUMNS 36-39

Code:	T–Terminal
Specification Type:	E
Explanation:	Number of entries per record specified is greater than the number of entries per table or array specified.
System Action:	The job is terminated.
User Response:	Make the proper entries in columns 33-35 and columns 36-39. The number of entries per record (columns 33-35) can be equal to or less than the number of entries per table or array (columns 36-39). Resubmit the job.

RG072–INVALID OR MISSING LENGTH OF ENTRY IN COL 40-42 OR 52-54, ASSUME 05

Code:	T–Terminal
Specification Type:	E
Explanation:	Length of entry specified is missing or is not a one to three-digit number (1-15 for numeric entries; 1-256 for alphabetic entries).
System Action:	05 is assumed, but the job is terminated.
User Response:	Enter a one to three-digit number in columns 40-42 or 52-54 to define length of table or array entries (1-15 or 1-256). Resubmit the job.

RG073–LENGTH SPECIFIED FOR EACH TABLE/ARRAY RECORD IN COL 33-35 AND COL 40-42 OR 52-54 EXCEEDS RECORD LENGTH

Code:	T–Terminal
Specification Type:	E
Explanation:	Table record length specified (length of entry times number of entries per record) is greater than the record length you specified for the table file in file description specifications.
System Action:	The job is terminated.
User Response:	Make the necessary changes so that the table record length does not exceed the record length in file description specifications. Resubmit the job.

RG074–INVALID ENTRY IN COL 43 OR 55, ASSUME BLANK

Code:	W–Warning
Specification Type:	E
Explanation:	The entry in column 43 or column 55 of your extension specifications is not P, B, or blank.
System Action:	Blank is assumed.
User Response:	Make the entry in column 43 or column 55 P, B, or blank. Resubmit the job.

RG075–PACKED OR BINARY VALID ONLY FOR PRE-EXECUTION TIME TABLE OR ARRAY

Code:	T–Terminal
Specification Type:	E
Explanation:	Packed or binary format can only be specified (column 43 or column 55) for pre-execution time tables or arrays.
System Action:	Job is terminated.
User Response:	To avoid this message the next time this job is run, leave column 43 and column 55 blank for compile time tables or arrays and for execution time arrays.

RG076–INVALID DECIMAL POSITION ENTRY IN COL 44 OR 56, ASSUME 0

Code:	T–Terminal
Specification Type:	E
Explanation:	Decimal position entry in column 44 or column 56 is not a number 0-9 or blank.
System Action:	Zero is assumed, but the job is terminated.
User Response:	Make the proper decimal position entry (0-9, blank) in columns 44 and 56. Resubmit the job.

RG077–INVALID SEQUENCE ENTRY IN COL 45 OR 57, ASSUME BLANK

Code:	T–Terminal
Specification Type:	E
Explanation:	Sequence entry in column 45 or column 57 is not A, D, or blank.
System Action:	Blank is assumed, but the job is terminated.
User Response:	Make the proper sequence entry (A, D, or blank) in column 45 or 57 and resubmit the job.

RG079–INVALID ALTERNATE TABLE/ARRAY NAME IN COL 46-51

Code:	T–Terminal
Specification Type:	E
Explanation:	The table or array name in columns 46-51 was not specified properly. The table or array name must start in column 46; a table name must begin with TAB.
System Action:	The job is terminated.
User Response:	Enter the proper table or array name in columns 46-51 and resubmit the job.

RG080–ALTERNATE TABLE/ARRAY NAME IN COL 46-51 AND/OR 27-32 MISSING FOR ENTRIES IN COLUMNS 33-45 AND/OR 52-57, ASSUME COL 33-57 AND/OR 46-57 BLANK

Code:	T–Terminal
Specification Type:	E
Explanation:	Columns 52-57 contain entries describing an alternating table or array, but no alternating table or array name was specified in columns 46-51 or no table or array name was specified in columns 27-32.
System Action:	The job is terminated.
User Response:	Make a valid table or array name entry in columns 27-32 and in columns 46-51 if an alternating table or array is described. Resubmit the job.

RG082–LENGTH OF TABLE/ARRAY ENTRY IN COL 40-42 OR 52-54 FOR ALPHA FIELDS EXCEEDS MAXIMUM.

Code:	T–Terminal
Specification Type:	E
Explanation:	Length of table or array entry specified in columns 40-42 or 52-54 is too large.
System Action:	256 is assumed for non-compile time tables or arrays. For compile time tables or arrays the READER record length is assumed.
User Response:	Enter 256 or less for the length of table or array entry specifications in columns 40-42 or 52-54.

RG083–LENGTH OF TABLE/ARRAY ENTRY IN COL 40-42 OR 52-54 FOR NUMERIC FIELD EXCEEDS 15, ASSUME 15

Code:	T–Terminal
Specification Type:	E
Explanation:	Length of numeric table or array entry specified in columns 40-42 or 52-54 is too large.
System Action:	15 is assumed, but the job is terminated.
User Response:	Enter 15 or less for the length of a numeric table or array entry in columns 40-42 and/or 52-54.

RG084–FILE AND RECORD TYPE ENTRIES IN COL 7-42 AND FIELD TYPE ENTRIES IN COL 43-74 ON SAME LINE, ASSUME 7-42 BLANK

Code:	T–Terminal
Specification Type:	I
Explanation:	Field type entries (columns 43-74) are not specified one line lower than file and record type entries (columns 7-42).
System Action:	File and record type entries (columns 7-42) are assumed to be blank and the job is terminated.
User Response:	Specify the field type entries (columns 43-74) one line lower than the file and record type entries (columns 7-42). Resubmit the job.

RG085–INVALID, MISSING OR UNDEFINED FILE NAME

Code:	T–Terminal
Specification Type:	L, I, C
Explanation:	Either (1) the filename was missing, (2) the filename was not specified properly, or (3) the filename was not previously defined in your file description specifications.
System Action:	The job is terminated.
User Response:	Make the proper filename entry. Also be sure that the filename has been previously defined in file description specifications. Resubmit the job.

RG086–FILENAME IN COL 7-14 DOES NOT REFER TO PRINTER FILE

Code:	T–Terminal
Specification Type:	L
Explanation:	Filename in your line counter specifications does not refer to a printer file.
System Action:	The job is terminated.
User Response:	Place the proper filename entry in columns 7-14. The filename specified must refer to a printer file. Resubmit the job.

RG087–FORM LENGTH ENTRY IN COL 15-17 INVALID OR GREATER THAN 112

Code:	T–Terminal
Specification Type:	L
Explanation:	The form length entry in columns 15-17 is not properly specified, or is too large.
System Action:	The job is terminated.
User Response:	Enter 112 or less for the form length entry in columns 15-17. Resubmit the job.

RG088–INVALID OR MISSING FL ENTRY IN COL 18-19, ASSUME FL

Code:	W–Warning
Specification Type:	L
Explanation:	Columns 18-19 were left blank or the entry specified is not FL.
System Action:	FL is assumed.
User Response:	To avoid this message when this job is run again, enter FL in columns 18-19.

RG089–OVERFLOW LINE ENTRY IN COL 20-22 INVALID OR GREATER THAN 112

Code:	T–Terminal
Specification Type:	L
Explanation:	The overflow line entry in columns 20-22 is invalid or a number greater than 112.
System Action:	The job is terminated.
User Response:	Columns 20-22 must be a number from 1-112.

RG090–INVALID OR MISSING OL ENTRY IN COL 23-24, ASSUME OL

Code:	W–Warning
Specification Type:	L
Explanation:	Columns 23-24 were left blank or the entry specified is not OL.
System Action:	OL is assumed.
User Response:	To avoid the message when this job is run again, enter OL in columns 23-24.

RG091–OVERFLOW LINE IN COL 20-22 EXCEEDS FORM LENGTH IN COL 15-17, ASSUME FORM LENGTH

Code:	T–Terminal
Specification Type:	L
Explanation:	Overflow line specified is too large.
System Action:	Form length is assumed, but the job is terminated.
User Response:	Make the overflow line entry (columns 20-22) equal to or less than the form length entry (columns 15-17).

RG092–INVALID OR UNDEFINED FILENAME IN COL 7-14

Code:	T–Terminal
Specification Type:	L, I, T
Explanation:	The filename entry is not specified properly, or it was not previously defined in your file description specifications.
System Action:	The job is terminated.
User Response:	Make the proper filename entry in columns 7-14. Also, make sure the filename has been previously defined in file description specifications. Resubmit the job.

RG093–FILE AND RECORD TYPE ENTRIES IN COL 7-42 AND FIELD TYPE ENTRIES IN COL 43-74 ON SAME LINE, ASSUME 43-74 BLANK

Code:	T–Terminal
Specification Type:	I
Explanation:	Field description entries (columns 43-74) are not specified one line lower than file and record identification entries (columns 7-42).
System Action:	Field type entries (columns 43-74) are assumed to be blank and the job is terminated.
User Response:	Specify the field type entries (columns 43-74) one line lower than the file and record type entries columns 7-42). Resubmit the job.

RG094–FILE AND RECORD TYPE DESCRIPTION MUST PRECEDE THIS SPECIFICATION

Code:	T–Terminal
Specification Type:	I
Explanation:	File and record type entries in columns 7-42 do not precede the related field description entries in columns 43-74.
System Action:	The job is terminated.
User Response:	Enter the file and record type entries in columns 7-42 of the specification line immediately preceding the related field description entries in columns 43-74. Resubmit the job.

RG095–AND OR OR LINE OUT OF ORDER

Code:	T–Terminal
Specification Type:	I, C
Explanation:	AND or OR line does not follow the proper file or record type entries or is on the first line of calculation specifications. (The system may have dropped your file and record type specifications because of other errors in your program.)
System Action:	The job is terminated.
User Response:	Make sure that the AND or OR line follows the proper file and record type entries and is not the first line in calculation specifications. Resubmit the job.

RG096–AND LINE FOLLOWS LINE WITH NO RECORD IDENTIFICATION CODES

Code:	T–Terminal
Specification Type:	I
Explanation:	The specification line which precedes your AND line does not contain record identification codes.
System Action:	The job is terminated.
User Response:	Make the proper record identification entries in the line preceding the AND line. Resubmit the job.

RG097–NO FIELDS DESCRIBED FOR THIS OR PREVIOUS RECORD

Code:	W–Warning
Specification Type:	I
Explanation:	No field description entries were specified for this or the previous record.
System Action:	No action is taken.
User Response:	Make sure that all fields to be used from input records are described.

RG098–INVALID SEQUENCE ENTRY IN COL 15-16, ASSUME ALPHABETIC SEQUENCE ENTRY

Code:	W–Warning
Specification Type:	I
Explanation:	The sequence entry in columns 15-16 is neither a two-digit number nor a two-character alphabetic entry.
System Action:	A two-character alphabetic entry is assumed.
User Response:	If this assumption was wrong, make the proper sequence entry and resubmit the job.

RG101–NUMERIC SEQUENCE ENTRY IN COL 15-16 NOT IN ASCENDING ORDER OR THE FIRST IS NOT 01, ASSUME PREVIOUS NUMERIC SEQUENCE OR 01 IF FIRST NUMERIC RECORD

Code:	W–Warning
Specification Type:	I
Explanation:	Either the first numeric sequence entry is not 01 or your numeric sequence entries are not in ascending order.
System Action:	If this is the first numeric sequence entry, 01 is assumed; otherwise, the numeric sequence entry from the previous specification line is assumed.
User Response:	If this assumption was wrong, specify the numeric sequence entries in columns 15-16 in ascending order starting with 01, and resubmit the job.

RG102–INVALID NUMBER ENTRY IN COL 17 FOR NUMERIC SEQUENCE, ASSUME N

Code:	W–Warning
Specification Type:	I
Explanation:	The number entry in column 17 is neither 1 nor N.
System Action:	N is assumed.
User Response:	If this assumption was wrong, make the proper number entry in column 17 and resubmit the job.

RG103–INVALID OPTION ENTRY IN COL 18 FOR NUMERIC SEQUENCE, ASSUME O

Code:	W–Warning
Specification Type:	I
Explanation:	The option entry is neither O nor blank.
System Action:	O is assumed.
User Response:	If this assumption was wrong, leave column 18 blank and resubmit the job.

RG104–NUMBER/OPTION ENTRIES IN COL 17-18 INVALID WITH ALPHAMERIC SEQUENCE ENTRIES

Code:	T–Terminal
Specification Type:	I
Explanation:	Columns 17 and 18 must be blank when columns 15-16 contain an alphabetic sequence entry.
System Action:	The job is terminated.
User Response:	Make columns 17 and 18 blank when columns 15-16 contain an alphabetic entry. Resubmit the job.

RG105–NUMBER/OPTION ENTRIES IN COL 17-18 INVALID FOR AND OR OR LINE, ASSUME BLANK

Code:	W–Warning
Specification Type:	I
Explanation:	Columns 17 and 18 must be blank in an AND or OR line.
System Action:	Blanks are assumed.
User Response:	To avoid the message when this job is run again, leave columns 17 and 18 of an AND or OR line blank.

RG106–INVALID POSITION ENTRY FOR RECORD ID CODES IN COL 21-14, 28-31, OR 35-38, OR TO POSITION COL 48-51, ASSUME 1

Code:	T–Terminal
Specification Type:	I
Explanation:	The position entry for record ID codes or the To position for a field exceeds the record length.
System Action:	One is assumed; the job is terminated.
User Response:	Make the proper position entry for record ID codes or To position for a field and resubmit the job.

RG107–INVALID NOT ENTRY IN COL 25, 32, OR 39, ASSUME N

Code:	W–Warning
Specification Type:	I
Explanation:	The entry in column 25, 32, or 39 is not N or blank.
System Action:	N is assumed.
User Response:	If this assumption was wrong, leave column 25, 32, or 39 blank and resubmit the job.

RG108–INVALID C/Z/D ENTRY IN COL 26, 33, OR 40, ASSUME C

Code:	W–Warning
Specification Type:	I
Explanation:	The entry in column 26, 33, or 40 is not C, Z, or D.
System Action:	C is assumed.
User Response:	If this assumption was wrong, make the proper entry in column 26, 33, or 40 and resubmit the job.

RG109–INVALID STACKER SELECT ENTRY IN COL 42 OR NOT ALLOWED WITH DEVICE, ASSUME BLANK

Code:	W–Warning
Specification Type:	I
Explanation:	Column 42 must be left blank, or contain a number from 1-4.
System Action:	Blank is assumed.
User Response:	To avoid this message the next time this job is run, leave column 42 blank or enter a number from 1-4.

RG110–STACKER SELECT ENTRY IN COL 42 INVALID WITH AN AND LINE; ASSUME BLANK

Code:	W–Warning
Specification Type:	I
Explanation:	The entry in column 42 is not blank.
System Action:	Blank is assumed.
User Response:	To avoid this message the next time this job is run, leave column 42 blank.

RG111–INVALID ENTRY IN COL 43, ASSUME BLANK

Code:	W–Warning
Specification Type:	I
Explanation:	The entry in column 43 is not P, B, or blank.
System Action:	Blank is assumed.
User Response:	If the assumption was wrong, make the proper entry in column 43 and resubmit the job.

RG112–INVALID OR BLANK FROM OR TO ENTRY IN COL 44-51, ASSUME 1 FOR BOTH ENTRIES

Code:	T–Terminal
Specification Type:	I
Explanation:	Columns 44-47 and/or 48-51 do not contain an entry from 1 to 4096.
System Action:	1 is assumed for columns 44-47 or columns 48-51, or for both; but the job is terminated.
User Response:	Make the proper From or To entry in columns 44-47 and/or 48-51 and resubmit the job.

RG113–FROM ENTRY IN COL 44-47 EXCEEDS TO ENTRY IN COL 48-51, ASSUME TO ENTRY EQUAL TO FROM ENTRY

Code:	T–Terminal
Specification Type:	I
Explanation:	From entry specified in columns 44-47 is larger than the To entry specified in columns 48-51.
System Action:	To entry is assumed to be equal to the From entry, but the job is terminated.
User Response:	Make the From entry (columns 44-47) equal to or less than the To entry (columns 48-51). Resubmit the job.

RG114–LENGTH OF NUMERIC FIELD IN COL 44-51 EXCEEDS 15, ASSUME 15

Code:	T–Terminal
Specification Type:	I
Explanation:	Length specified in columns 44-51 for numeric field is too large.
System Action:	Length of 15 is assumed, but the job is terminated.
User Response:	Make the length (columns 44-51) 15 or less. Resubmit the job.

RG115–ALPHAMERIC FIELD SPECIFIED AS PACKED OR BINARY, ASSUME NUMERIC FIELD

Code:	T–Terminal
Specification Type:	I
Explanation:	Column 43 must be blank for alphameric fields.
System Action:	The field is assumed to be numeric, but the job is terminated.
User Response:	Leave column 43 blank for alphameric fields or make an entry (0-9) in column 52 for numeric fields, and resubmit the job.

RG116–INVALID DECIMAL POSITION ENTRY IN COL 52; ASSUME 0

Code:	T–Terminal
Specification Type:	I
Explanation:	Decimal position entry in column 52 is not 0-9 or blank.
System Action:	Zero is assumed, but the job is terminated.
User Response:	Make the proper decimal position entry in column 52 and resubmit the job.

RG117–DECIMAL POSITION ENTRY IN COL 52 INVALID FOR ARRAY; ASSUME BLANK

Code:	W–Warning
Specification Type:	I
Explanation:	No decimal position entry can be specified in column 52 for an array.
System Action:	Blank is assumed.
User Response:	Leave column 52 blank for an array. Decimal positions for arrays must be specified in your extension specifications. Resubmit the job.

RG118–FIELD NAME IN COL 53-58 MISSING OR INVALID

Code:	T–Terminal
Specification Type:	I
Explanation:	The field name entry in columns 53-58 is missing or is not specified properly.
System Action:	The job is terminated.
User Response:	Make a valid field name entry starting in column 53. Resubmit the job.

RG119–INVALID CONTROL LEVEL INDICATOR IN COL 59-60; ASSUME BLANK

Code:	T–Terminal
Specification Type:	I
Explanation:	The control level entry in columns 59-60 is neither L1-L9 nor blank.
System Action:	Blank is assumed, but the job is terminated.
User Response:	Make the proper control level entry in columns 59-60 and resubmit the job.

RG120–INVALID MATCHING FIELD ENTRY IN COL 61-62; ASSUME M1

Code:	T–Terminal
Specification Type:	I
Explanation:	The matching field entry in columns 61-62 is not M1-M9 or blank.
System Action:	M1 is assumed, but the job is terminated.
User Response:	Make the proper matching fields entry in columns 61-62 and resubmit the job.

RG121–FROM FILE CANNOT HAVE AN E IN COL 17 OF FILE DESCRIPTION SPECIFICATION WHEN TO FILE IS A DEMAND FILE

Code:	T–Terminal
Specification Type:	E
Explanation:	End of file, E in column 17 of the file description specifications, cannot be used for a record address file which is used to process a demand file.
System Action:	Job is terminated.
User Response:	Leave column 17 blank and resubmit the job.

RG122–FIELD WAS PREVIOUSLY DEFINED WITH DIFFERENT LENGTH OR DECIMAL POSITIONS, ASSUME FIRST DEFINITION–OR FIELD IS NOW DEFINED AS A LOOK AHEAD FIELD

Code:	W–Warning
Specification Type:	I or C
Explanation:	A conflicting length or number of decimal positions has been detected, or field is now used as a look ahead field.
System Action:	The length or decimal positions of the first entry are assumed, or second field is a look ahead field.
User Response:	If the assumption was wrong, define the field correctly and resubmit the job.

RG123–INVALID ENTRY IN COL 7-8

Code:	T–Terminal
Specification Type:	C
Explanation:	The control level entry in columns 7-8 is not AN, OR, L0-L9, LR, SR, or blank.
System Action:	The job is terminated.
User Response:	Make the proper control level entry in columns 7-8 and resubmit the job.

RG124–INVALID NOT ENTRY IN COL 9, 12, OR 15; ASSUME N

Code:	W–Warning
Specification Type:	C
Explanation:	The entry in column 9, 12, or 15 is not N or blank.
System Action:	N is assumed.
User Response:	If this assumption was wrong, leave column 9, 12, or 15 blank and resubmit the job.

RG125–INVALID FIELD NAME OR CONSTANT FOR FACTOR 1 IN COL 18-27

Code:	T–Terminal
Specification Type:	C
Explanation:	The field name or constant in columns 18-27 is not specified properly. Both must begin in column 18.
System Action:	The job is terminated.
User Response:	Make the proper field name or constant entry in columns 18-27. Resubmit the job.

RG126–LENGTH OF TABLE/ARRAY EXCEEDS MAXIMUM CORE STORAGE

Code:	T–Terminal
Specification Type:	E
Explanation:	The number of entries per table or array (columns 36-39) multiplied by the length of entry (columns 40-42) exceeds maximum storage.
System Action:	Job is terminated.
User Response:	Reduce the number of entries or the length of the entries.

RG127–ENTRY IN COL 49-51 INVALID WITH NO RESULT FIELD, ASSUME 49-51 BLANK

Code:	W–Warning
Specification Type:	C
Explanation:	This calculation specification contains a field length entry (columns 49-51) but no result field entry (columns 43-48).
System Action:	Blank in columns 49-51 is assumed.
User Response:	If a result field is being described, place the proper entry in columns 43-48 and resubmit the job.

RG128–INVALID OPERATION CODE ENTRY IN COL 28-32

Code:	T–Terminal
Specification Type:	C
Explanation:	Operation code is not specified properly.
System Action:	The job is terminated.
User Response:	Enter the proper RPG II operation code in columns 28-32, and resubmit the job.

RG129–FACTOR 2 FIELD NAME IN COL 33-42 EXCEEDS SIX CHARACTERS

Code:	T–Terminal
Specification Type:	C
Explanation:	The field name or label specified in Factor 2 is too large.
System Action:	The job is terminated.
User Response:	Make the field name or label in Factor 2 (columns 33-42) six characters or less. Resubmit the job.

RG130–TO FILE MUST BE A LIMITS FILE IF FROM FILE IS A RECORD ADDRESS FILE, OR TO FILE MUST BE A RANDOM ACCESS FILE IF FROM FILE IS AN ADDROUT FILE

Code:	T–Terminal
Specification Type:	E
Explanation:	The file types specified on the Extension specification are not used properly.
System Action:	Job is terminated.
User Response:	Make To file a limits file if From file is a record address file, or make To file a random access file if From file is an ADDROUT file. Resubmit the job.

RG131–FACTOR 2 IN COL 33-42 INVALID

Code:	T–Terminal
Specification Type:	C
Explanation:	The field name or constant in columns 33-42 is not specified properly. Entry must start in column 33.
System Action:	The job is terminated.
User Response:	Make the proper field name or constant entry in columns 33-42. Resubmit the job.

RG132–FACTOR 2 MUST BE A FILENAME

Code:	T–Terminal
Specification Type:	C
Explanation:	For this operation the entry in Factor 2 must be a filename.
System Action:	The job is terminated.
User Response:	Make the proper filename entry in Factor 2 (columns 33-42) for this operation, Resubmit the job.

RG133–NUMERIC FIELD LENGTH EXCEEDS 15; ASSUME 15

Code:	T–Terminal
Specification Type:	C
Explanation:	Length specified in columns 49-51 for numeric field is too large.
System Action:	Length of 15 is assumed, but the job is terminated.
User Response:	Make the length (columns 49-51) 15 or less. Resubmit the job.

RG134–ALPHAMERIC FIELD LENGTH EXCEEDS 256; ASSUME 256

Code:	T–Terminal
Specification Type:	C
Explanation:	Length specified in columns 49-51 for an alphameric field is too large.
System Action:	Length of 256 is assumed, but the job is terminated.
User Response:	Make the length (columns 49-51) 256 or less. Resubmit the job.

RG135–INVALID RESULT FIELD ENTRY IN COL 43-53

Code:	T–Terminal
Specification Type:	C
Explanation:	The result field entry in columns 43-53 is not specified properly.
System Action:	The job is terminated.
User Response:	Make the proper result field entries, and resubmit the job.

RG136–FIELD LENGTH EXCEEDS CAPACITY OF DEVICE USED WITH DSPLY OP CODE

Code:	T–Terminal
Specification Type:	C
Explanation:	Models 10 and 12: Field to be displayed exceeds 125 characters. Model 15: Field to be displayed exceeds 35 characters.
System Action:	The job is terminated.
User Response:	Make the field length equal to or less than the maximum length for the appropriate system and resubmit the job.

RG137–INVALID RESULT FIELD LENGTH IN COL 49-51; ASSUME 15 FOR NUMERIC OR 256 FOR ALPHAMERIC FIELD

Code:	T–Terminal
Specification Type:	C
Explanation:	The field length entry in columns 49-51 is not specified properly.
System Action:	15 is assumed for numeric fields; 256 is assumed for alphameric fields. The job is terminated.
User Response:	Enter 15 or less in columns 49-51 for numeric fields, 256 or less for alphameric fields. Resubmit the job.

RG138–DECIMAL POSITION ENTRY IN COL 52 INVALID WITH NO FIELD LENGTH ENTRY IN COL 49-51; ASSUME BLANK

Code:	T–Terminal
Specification Type:	C
Explanation:	Column 52 must be blank when columns 49-51 are blank.
System Action:	Blank in column 52 is assumed, but the job is terminated.
User Response:	Leave column 52 (decimal position) blank when columns 49-51 (field length) are blank. Resubmit the job.

RG139–INVALID DECIMAL POSITION ENTRY IN COL 52; ASSUME 0

Code:	T–Terminal
Specification Type:	C
Explanation:	The decimal position entry is not a number from 0 to 9 or blank.
System Action:	Zero is assumed, but the job is terminated.
User Response:	Make the proper decimal position entry (0-9 or blank) in column 52 and resubmit the job.

RG140–INVALID HALF ADJUST ENTRY IN COL 53; ASSUME H

Code:	W–Warning
Specification Type:	C
Explanation:	The half adjust entry in column 53 is neither H nor blank.
System Action:	H is assumed.
User Response:	If this assumption was wrong, leave column 53 blank and resubmit the job.

RG141–DEBUG CALCULATION OPERATION USED, BUT DEBUG OPTION NOT SPECIFIED IN THE CONTROL CARD

Code:	W–Warning
Specification Type:	C
Explanation:	The DEBUG operation code was used in your calculation specifications, but you had not specified the DEBUG option (1 in column 15) in your control card specifications.
System Action:	DEBUG operations are not executed.
User Response:	Specify the DEBUG option (1 in column 15) in your control card specifications if you have DEBUG statements to be executed, and resubmit the job.

RG142–FILE AND RECORD IDENTIFICATION ENTRIES IN COL 7-31 AND FIELD DESCRIPTION ENTRIES IN COL 32-74 ON SAME LINE

Code:	T–Terminal
Specification Type:	O
Explanation:	Your field description entries in columns 23-74 are not specified one line lower than the file and record identification entries in columns 7-31.
System Action:	Blanks are assumed for columns 7-31 and the job is terminated.
User Response:	Place the field description entries (columns 32-74) one line lower than the file and record identification entries (columns 7-31). Resubmit the job.

RG143–INVALID LINE TYPE ENTRY IN COL 15

Code:	T–Terminal
Specification Type:	O
Explanation:	The line type entry in column 15 is not H, D, T, or E. An E can be used only if an EXCPT operation is used in calculations.
System Action:	H is assumed; the job is terminated.
User Response:	Make the proper line type entry in column 15 and resubmit the job.

RG144–AND OR OR LINE NOT PRECEDED BY RECORD IDENTIFICATION

Code:	T–Terminal
Specification Type:	O
Explanation:	An AND or OR line is not preceded by record identification entries in columns 15-31.
System Action:	The job is terminated.
User Response:	Make sure that record identification entries in columns 15-31 precede any AND or OR lines. Resubmit the job.

RG145–INVALID ENTRIES IN COL 17-22 FOR AND LINE, ASSUME BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	Columns 17-22 of an AND line contain space/skip entries; they should be blank.
System Action:	Blanks are assumed:
User Response:	To avoid this message when the job is run again, remove all space/skip entries (columns 17-22) from an AND line.

RG146–INVALID FILENAME OR ENTRY IN COL 15 MISSING ON FIRST OUTPUT SPECIFICATION

Code:	T–Terminal
Specification Type:	O
Explanation:	Either columns 7-14 contain an invalid filename or no line type entry was specified in column 15 of the specification line.
System Action:	The job is terminated.
User Response:	Check to make sure the proper filename entry is made in columns 7-14 and that the proper line type entry is made in column 15. Resubmit the job.

RG147–INVALID NOT ENTRY IN COL 23, 26, OR 29; ASSUME N

Code:	W–Warning
Specification Type:	O
Explanation:	The entry in column 23, 26, or 29 is neither N nor blank.
System Action:	N is assumed.
User Response:	If this assumption was wrong, make the proper entry in column 23, 26, or 29 and resubmit the job.

RG148–INVALID FIELD NAME IN COL 32-37

Code:	T–Terminal
Specification Type:	O
Explanation:	The field name entry in columns 32-37 is not specified properly or was not defined previously in input or calculation specifications.
System Action:	The job is terminated.
User Response:	Make the proper field name entry starting in column 32 and resubmit the job.

RG149–INVALID OR MISSING CONSTANT

Code:	T–Terminal
Specification Type:	O
Explanation:	The constant in columns 45-70 is not specified properly.
System Action:	The job is terminated.
User Response:	Make the proper entry in columns 45-70 and resubmit the job.

RG150–INVALID BLANK AFTER ENTRY IN COL 39; ASSUME BLANK

Code:	T–Terminal
Specification Type:	O
Explanation:	The blank after entry in column 39 is neither B nor blank.
System Action:	Blank is assumed, but the job is terminated.
User Response:	Make the proper entry in column 39 and resubmit the job.

RG151–MISSING OR INCORRECTLY SPECIFIED END POSITION IN COL 40-43; ASSUME END POSITION 1

Code:	T–Terminal
Specification Type:	O
Explanation:	The end positions entry in columns 40-43 is either missing or is not specified properly.
System Action:	The job is terminated.
User Response:	Make the proper numeric entry in columns 40-43; the entry must end in column 43. Resubmit the job.

RG152–INVALID ENTRY IN COL 44; ASSUME BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	The entry in column 44 is not P, B, or blank.
System Action:	Blank is assumed.
User Response:	If the assumption was wrong, make the proper entry in column 44 and resubmit the job.

RG153—END POSITION IN COL 40-43 INVALID FOR *PRINT; ASSUME BLANK

Code:	W—Warning
Specification Type:	O
Explanation:	End position may not be specified for *PRINT.
System Action:	No action taken.
User Response:	To avoid this message the next time the job is run, remove the end position (columns 40-43) for the *PRINT.

RG154—ENTRIES IN COL 7-22 INVALID FOR A FIELD DESCRIPTION SPECIFICATION; ASSUME BLANK

Code:	T—Terminal
Specification Type:	O
Explanation:	The file and record identification entries in columns 7-22 are not specified one line above the first related field description entries.
System Action:	The job is terminated.
User Response:	Place your file and record identification entries (columns 7-22) one line above the field description entries (columns 32-74). Resubmit the job.

RG155—INVALID POSITION ENTRY IN COL 71-74; ASSUME BLANK

Code:	T—Terminal
Specification Type:	I or O
Explanation:	Columns 71-74 should be blank.
System Action:	Blank is assumed and job is terminated.
User Response:	Columns 71-74 should be blank.

RG158—TABLE NAME INVALID FOR A FIELD NAME ENTRY IN COL 53-58

Code:	T—Terminal
Specification Type:	I
Explanation:	The field name entry in columns 53-58 refers to a table.
System Action:	The job is terminated.
User Response:	Place the proper field name entry in columns 53-58; the entry must not be a table name. Resubmit the job.

RG159—MISSING RECORD IDENTIFYING INDICATOR IN COL 19-20

Code:	W—Warning
Specification Type:	I
Explanation:	No record identifying indicator is specified in columns 19-20.
System Action:	No action taken.
User Response:	Check your input specifications to determine whether or not a record identifying indicator should be entered in columns 19-20. If so, make the proper entry and resubmit the job.

RG160—FILE NAMED IN COL 7-14 NOT SPECIFIED AS AN INPUT, COMBINED, UPDATE-PRIMARY, SECONDARY, DEMAND, OR CHAINED FILE

Code:	T—Terminal
Specification Type:	I
Explanation:	The file named in columns 7-14 was not previously defined in file description specifications as an input, combined, or update file with a designation of primary, secondary, demand, or chained.
System Action:	The job is terminated.
User Response:	Make sure the file named in columns 7-14 is properly defined in file description specifications. Resubmit the job.

RG161—AND OR OR LINE INVALID WITH LOOK AHEAD RECORDS OR RLABL

Code:	T—Terminal
Specification Type:	I, C
Explanation:	An AND or OR line was used with look ahead fields or RLABL.
System Action:	The job is terminated.
User Response:	Make sure that AND or OR lines are not specified for look ahead fields (**in columns 19-20) or for RLABL. Resubmit the job.

RG162—RECORD IDENTIFYING INDICATOR IN COL 19-20 INVALID FOR AN AND LINE

Code: W—Warning
Specification Type: I
Explanation: A record identifying indicator is in columns 19-20 of an AND line.
System Action: Blanks are assumed.
User Response: To avoid this message next time this job is run, leave columns 19-20 of the AND line blank.

RG163—ENTRIES IN COL 17-18 AND 21-42 INVALID FOR LOOK AHEAD RECORD. ENTRIES IN 59-74 INVALID FOR LOOK AHEAD FIELD

Code: T—Terminal
Specification Type: I
Explanation: Columns 17-18 and 21-42 must be blank for look ahead records, columns 59-74 must be blank for look ahead fields.
System Action: The job is terminated.
User Response: Leave columns 17-18 and 21-42 blank for look ahead records; leave columns 59-74 blank for look ahead fields. Resubmit the job.

RG164—STACKER SELECT ENTRY IN COL 42 INVALID FOR DEVICE SPECIFIED; ASSUME BLANK

Code: W—Warning
Specification Type: I
Explanation: Column 42 must be blank for a console, disk, or SPECIAL file.
System Action: Blank is assumed.
User Response: Remove the entry from column 42.

RG165—INDICATORS IN COL 65-70 INVALID FOR A TABLE/ARRAY

Code: T—Terminal
Specification Type: I
Explanation: Field indicators cannot be used if columns 53-58 contain a table or array name.
System Action: The job is terminated.
User Response: Use the field indicators to test numeric fields.

RG166—PLUS OR MINUS INDICATOR IN COL 65-68 INVALID FOR ALPHAMERIC FIELD

Code: T—Terminal
Specification Type: I
Explanation: A Plus or Minus indicator in columns 65-68 cannot be used to test an alphameric field.
System Action: Blank is assumed; the job is terminated.
User Response: Use Plus or Minus indicators only to test numeric fields. An alphameric field can only be tested for a blank condition (entry in columns 69-70). Resubmit the job.

RG167—RECORD ID POSITION 21-38 OR TO ENTRY IN COL 48-51 EXCEEDS RECORD LENGTH, ASSUME RECORD LENGTH

Code: T—Terminal
Specification Type: I
Explanation: Field location entries (columns 21-38 and 48-51) exceed record length specified in file description specifications.
System Action: Record length is assumed; the job is terminated.
User Response: Make the field location entries (columns 21-38 and 48-51) equal to or less than the record length specified on file description specifications. Resubmit the job.

RG168—FIELD NAME IN COL 53-58 IS A RESERVED WORD OTHER THAN PAGE

Code: T—Terminal
Specification Type: I
Explanation: The field name entry in columns 53-58 is a reserved word other than PAGE.
System Action: The job is terminated.
User Response: Make the proper field name entry in columns 53-58 (PAGE is the only RPG II reserved word that can be entered in these columns). Resubmit the job.

RG169–CONTROL OR MATCHING FIELDS INVALID FOR ARRAY OR TRAILER RECORD

Code:	T–Terminal
Specification Type:	I
Explanation:	Control or matching fields must not be specified for arrays or trailer records.
System Action:	The job is terminated.
User Response:	Make sure no control or matching fields are specified for array or trailer records. Resubmit the job.

RG170–MATCHING OR CONTROL FIELDS INVALID WITH DEMAND OR CHAIN FILES

Code:	T–Terminal
Specification Type:	I
Explanation:	Matching or control fields cannot be specified for demand or chain files.
System Action:	The job is terminated.
User Response:	Make sure that matching or control fields are not specified for demand or chain files. Resubmit the job.

RG171–LOOK AHEAD RECORDS INVALID WITH DEMAND FILES, CHAIN FILES, FILES CONTAINING SPREAD CARDS, OR WITH THIS DEVICE

Code:	T–Terminal
Specification Type:	I
Explanation:	Look ahead records cannot be specified for demand files, or chained files, files containing spread cards or with this device.
System Action:	The job is terminated.
User Response:	Make sure that look ahead records are not specified for demand or chained files, for a file containing spread cards, or with this device.

RG172–INCORRECT SEQUENCE OF INPUT SPECIFICATIONS

Code:	T–Terminal
Specification Type:	I
Explanation:	All records from one input, update, or combined files are not specified consecutively.
System Action:	The job is terminated.
User Response:	Specify all records from one input, update, or combined file consecutively before starting to describe records from a different file.

RG173–NO FIELDS SPECIFIED FOR LOOK AHEAD RECORD

Code:	T–Terminal
Specification Type:	I
Explanation:	A look ahead record is specified (**in columns 19-20), but no look ahead fields are defined (columns 53-58).
System Action:	The job is terminated.
User Response:	Make the proper look ahead field specifications in columns 53-58 for a look ahead record. Resubmit the job.

RG174–LIMITS FILE NOT PROCESSED BY R/A FILE OR SETLL OP CODE

Code:	T–Terminal
Specification Type:	F or C
Explanation:	A limits file is being processed sequentially within limits, but doesn't have a record address file or a SETLL op code associated with it.
System Action:	Job is terminated.
User Response:	Make the proper entries for a limits file and resubmit the job.

RG175–INVALID FILE TYPE FOR SETLL OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	The file to be processed by a SETLL operation must be a limits file not already specified to be processed using a record address file.
System Action:	The job is terminated.
User Response:	Make the necessary corrections and resubmit the job.

RG178–BINARY INVALID WITH CONTROL OR MATCHING FIELDS

Code:	T–Terminal
Specification Type:	I
Explanation:	Binary fields have been used as control or matching fields.
System Action:	The job is terminated.
User Response:	Do not specify a binary field as a control or matching field.

RG180–ARRAY LENGTH EXCEEDS LENGTH SPECIFIED IN COL 36-42 IN EXTENSION SPECIFICATIONS OR NOT A MULTIPLE OF THE ENTRY LENGTH IN COL 40-42 IN EXTENSION SPECIFICATIONS

Code:	T–Terminal
Specification Type:	I
Explanation:	The array length either exceeds the length specified in columns 36-42 of your extension specification, or is not a multiple of the length entry in columns 40-42 of the extension specification, or both.
System Action:	The job is terminated.
User Response:	Make the array length equal to or less than the length specified in columns 36-42 of extension specifications. The length must also be a multiple of the length of an array element (columns 40-42 of extension specifications). Resubmit the job.

RG181–INCONSISTENT FIELD LENGTHS FOR CONTROL OR MATCHING FIELDS OF ONE LEVEL. ASSUME FIRST VALID LENGTH

Code:	T–Terminal
Specification Type:	I
Explanation:	All control or matching fields of one level were not assigned the same field length.
System Action:	The job is terminated.
User Response:	Assign the same field length to all control of matching fields of the same level. Resubmit the job.

RG182–INVALID SPLIT CONTROL FIELD SPECIFICATION, ASSUME PREVIOUS TOTAL LENGTH FOR THIS LEVEL

Code:	T–Terminal
Specification Type:	I
Explanation:	Specifications for split control fields of the same level are not specified on successive lines.
System Action:	The job is terminated.
User Response:	Place the specifications for split control fields of the same level on successive lines. Resubmit the job.

RG183–CONTROL OR MATCHING FIELDS OF A LEVEL SPECIFIED AS BOTH ALPHAMERIC AND NUMERIC, ASSUME NUMERIC

Code:	W–Warning
Specification Type:	I
Explanation:	All control and matching fields assigned the same level are not the same type (alphameric or numeric).
System Action:	Numeric is assumed for all fields assigned the same control or matching level. If any of the fields specified as alphameric are greater than 15 characters, only a portion of the field will be used.
User Response:	If this assumption is wrong, make the proper entry and resubmit the job.

RG184–ALL OF THE VALID MATCH LEVELS WERE NOT REFERENCED IN LAST RECORD GROUP

Code:	T–Terminal
Specification Type:	I
Explanation:	The same number of match levels are not specified to all record types in a file.
System Action:	The job is terminated.
User Response:	Make sure that all record types in a file either have no match levels or have the same number of match levels specified. Resubmit the job.

RG186–MATCH OR CONTROL FIELDS WITHOUT FIELD RECORD RELATION ENTRIES MUST PRECEDE MATCH OR CONTROL FIELDS WITH FIELD RECORD RELATION ENTRIES, ASSUME PART OF A NEW GROUP OF MATCH FIELDS

Code:	T–Terminal
Specification Type:	I
Explanation:	All match or control fields without field record relation entries (columns 63-64) do not precede those fields that do have field record relation entries.
System Action:	The job is terminated.
User Response:	Place all match or control fields without field record relation entries before those match or control fields with field record relation entries. Resubmit the job.

RG187–MATCH AND CONTROL FIELDS WITH FIELD RECORD RELATION ENTRIES MUST BE GROUPED ACCORDING TO THE FIELD RECORD RELATION INDICATOR. ASSUME NEW GROUP OF MATCH FIELDS

Code:	T–Terminal
Specification Type:	I
Explanation:	When field record relation is used, all match and control fields assigned the same indicator (columns 63-64) must be grouped together.
System Action:	A new group is assumed, but the job is terminated.
User Response:	Group all match and control fields with the same field record relation indicator together. Resubmit the job.

RG188–FIELD RECORD RELATION INDICATOR USED IMPROPERLY WITH MATCH OR CONTROL FIELDS

Code:	T–Terminal
Specification Type:	I
Explanation:	When used with match or control fields, the field record relation indicator in columns 63-64 does not match a record identifying indicator used for this file.
System Action:	The job is terminated.
User Response:	Make the field record relation indicator in columns 63-64 match a record identifying indicator for this file. Resubmit the job.

RG189–INVALID SEQUENCE FOR CALCULATION SPECIFICATIONS OR SR NOT SPECIFIED IN COLUMNS 7-8 WITH BEGSR OR ENDSR

Code:	T–Terminal
Specification Type:	C
Explanation:	Calculation specifications are not specified in this order: detail, total, and subroutines.
System Action:	The job is terminated.
User Response:	Place calculation specifications in this order: detail, total, and subroutines. Resubmit the job.

RG190–INVALID SEQUENCE FOR BEGSR AND ENDSR OPERATION CODES

Code:	T–Terminal
Specification Type:	C
Explanation:	BEGSR operation code does not precede ENDSR operation code.
System Action:	The job is terminated.
User Response:	Place the BEGSR specification before the ENDSR specification in a subroutine. Resubmit the job.

RG191–A SUBROUTINE MUST NOT CALL ITSELF

Code:	T–Terminal
Specification Type:	C
Explanation:	An EXSR specification within a subroutine must not call the subroutine it is in.
System Action:	The job is terminated.
User Response:	If you wish to branch to another point within the same subroutine use a GOTO and TAG operation. Resubmit the job.

RG192–BRANCHING BETWEEN SUBROUTINE AND OTHER CALCULATIONS INVALID

Code:	T–Terminal
Specification Type:	C
Explanation:	Branching (GOTO and TAG) can only occur within a subroutine. You cannot branch into a subroutine or out of a subroutine.
System Action:	The job is terminated.
User Response:	When using subroutines, make sure branching between a subroutine and other calculations is not specified. Make the necessary changes and resubmit the job.

RG193–BRANCHING BETWEEN DETAIL, TOTAL AND LR CALCULATIONS INVALID

Code:	T–Terminal
Specification Type:	C
Explanation:	Branching must be from detail operation to detail operation or from total operation to total operation. It cannot be from detail to total operation or vice versa.
System Action:	The job is terminated.
User Response:	Remove any operations that attempt to branch from detail to total calculations or vice versa. Resubmit the job.

RG194–SETOF OPERATION INVALID FOR LR INDICATOR

Code:	T–Terminal
Specification Type:	C
Explanation:	The LR indicator cannot be turned off by the SETOF operation code.
System Action:	The job is terminated.
User Response:	Remove the SETOF LR specification and resubmit the job.

RG195–LENGTH OF SEARCH WORD NOT EQUAL TO LENGTH OF ELEMENT IN TABLE OR ARRAY

Code:	T–Terminal
Specification Type:	C
Explanation:	Length of search word (Factor 1) is not equal to length of element in table or array being searched.
System Action:	The job is terminated.
User Response:	Make the length of the search word (Factor 1) equal to the length of the element in the table or array being searched. Resubmit the job.

RG196–FACTOR 2 OR RESULT FIELD INVALID FOR LOKUP OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	Either Factor 2 or Result Field is invalid for this LOKUP operation.
System Action:	The job is terminated.
User Response:	Specify LOKUP operation with table or array name in Factor 2 or in Result Field. Resubmit the job.

RG197–SEARCH TABLE HAS MORE ENTRIES THAN ITS RELATED TABLE

Code:	W–Warning
Specification Type:	C
Explanation:	The search table (Factor 2) contains more entries than its related table.
System Action:	No action taken.
User Response:	To avoid this message the next time this job is run, make the number of entries in the table being searched (Factor 2) equal or less than the number of entries in the related table (result field).

RG198–INDICATOR ENTERED IN COL 54-57 INVALID WITH LOKUP ON AN UNSEQUENCED TABLE OR ARRAY

Code:	W–Warning
Specification Type:	C
Explanation:	You should not specify a search for high or low in a LOKUP operation on an unsequenced table or array.
System Action:	Accept indicator as used.
User Response:	Specify the LOKUP operation on an unsequenced table or array for an equal condition only (indicator in columns 58-59). Resubmit the job.

RG199–TEST FOR BOTH HIGH AND LOW INVALID FOR LOKUP OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	You must not specify a search for both high and low in the same LOKUP operation.
System Action:	The job is terminated.
User Response:	Specify the LOKUP for either high or low, but not both. Resubmit the job.

RG200–RESULTING INDICATORS IN COL 54-59 REQUIRED OR NOT ALLOWED FOR OPERATION SPECIFIED

Code:	T–Terminal
Specification Type:	C
Explanation:	The resulting indicator entry in columns 54-59 is not specified properly.
System Action:	The job is terminated.
User Response:	Check to determine whether resulting indicators are required for this operation. If so, make the proper entries (01-99, H1-H9, L1-L9, LR, OA-OG, OV, or KA-KN, KP, KQ), resubmit the job.

RG201–HALF ADJUST ENTRY IN COL 53 FOR DIVISION OPERATION FOLLOWED BY A MVR OPERATION: ASSUME NO HALF ADJUST

Code:	W–Warning
Specification Type:	C
Explanation:	When an MVR operation follows a DIV operation, the DIV operation must not be half adjusted.
System Action:	No half adjusting is done.
User Response:	To avoid this message the next time this job is run, leave column 53 (Half Adjust) blank.

RG202–MVR OPERATION CODE DOES NOT FOLLOW DIV OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	The MVR operation must immediately follow a DIV operation.
System Action:	The job is terminated.
User Response:	Place an MVR operation immediately after a DIV operation or remove the MVR operation and resubmit the job.

RG204–HALF ADJUST ENTRY IN COL 53 INVALID FOR OPERATION OR NUMBER OF DECIMAL POSITIONS SPECIFIED; ASSUME BLANK

Code:	W–Warning
Specification Type:	C
Explanation:	Half adjusting (H in column 53) cannot be done for this operation or half adjusting is invalid for the number of decimal positions specified.
System Action:	Column 53 is assumed to be blank; therefore no half adjusting is done.
User Response:	To avoid this message the next time the job is run, leave column 53 blank for this operation.

RG205–COMP, TESTZ, OR MVR INVALID FOR AN ARRAY

Code:	T–Terminal
Specification Type:	C
Explanation:	COMP, TESTZ, and MVR must not be specified for an array.
System Action:	The job is terminated.
User Response:	Delete any COMP, TESTZ and MVR operations specified for an array. Resubmit the job.

RG206–INVALID USE OF COMP OR LOKUP

Code:	T–Terminal
Specification Type:	C
Explanation:	COMP or LOKUP operation specified improperly.
System Action:	The job is terminated.
User Response:	Make sure that Factor 1 and Factor 2 of a COMP operation are both alphameric or both numeric. Make sure the search word and the table or array to be searched are both alphameric or both numeric. Resubmit the job.

RG207–FIELD TYPE, ALPHAMERIC OR NUMERIC, INVALID FOR OPERATION SPECIFIED

Code:	T–Terminal
Specification Type:	C
Explanation:	This operation requires a different field type (alphameric or numeric).
System Action:	The job is terminated.
User Response:	Make the proper field type entry (alphameric or numeric) and resubmit the job.

RG208–FORCE OPERATION INVALID AT TOTAL TIME

Code:	T–Terminal
Specification Type:	C
Explanation:	FORCE operation must be specified at detail time only.
System Action:	The job is terminated.
User Response:	Specify the Force operation at detail time and resubmit the job.

RG209–FILE TYPE INVALID FOR USE WITH THIS OPERATION CODE

Code:	T–Terminal
Specification Type:	C
Explanation:	DEBUG and EXCPT must be used with an output file; FORCE must be used with an input or combined primary or secondary file.
System Action:	The job is terminated.
User Response:	Specify the proper file type or a different operation code, and resubmit the job.

RG211–DEBUG SPECIFIED FOR MORE THAN ONE OUTPUT FILE

Code:	T–Terminal
Specification Type:	C
Explanation:	The filename entered in Factor 2 is not the same for all DEBUG operations.
System Action:	The job is terminated.
User Response:	Place the same filename in Factor 2 for all DEBUG operations and resubmit the job.

RG212–EXCPT OPERATION CODE SPECIFIED BUT NO EXCPT OUTPUT RECORDS SPECIFIED

Code:	W–Warning
Specification Type:	C
Explanation:	The EXCPT operation code is used but no EXCPT records are specified (E in column 15 of the output specifications).
System Action:	No action taken.
User Response:	To avoid this message, either delete the EXCPT operation code or specify the proper exception records in output specifications.

RG213–PROGRAM CONTAINS UNASSOCIATED OR MISSING EXSR/BEGSR LABEL

Code:	T–Terminal
Specification Type:	C
Explanation:	The label in Factor 2 of an EXSR operation is not the same as the label in Factor 1 of a BEGSR operation, or the label in a calculation subroutine is specified (BEGSR) but is never referenced by an EXSR.
System Action:	The job is terminated.
User Response:	(1) Make the label in Factor 2 of the EXSR operation the same as the label in Factor 1 of the associated BEGSR operation. (2) Avoid this message by either deleting the entire calculation subroutine or by specifying associated EXSR operation.

RG214–GOTO BRANCHES TO A BEGSR NAME

Code:	T–Terminal
Specification Type:	C
Explanation:	The label in Factor 2 of a GOTO operation must be the same as the label in Factor 1 of a TAG operation.
System Action:	The job is terminated.
User Response:	Make the label in Factor 2 of a GOTO operation the same as the label in Factor 1 of a TAG operation. Resubmit the job.

RG215–FACTOR 1 ENTRY IN COL 18-27 MISSING

Code:	T–Terminal
Specification Type:	C
Explanation:	No entry specified in Factor 1 for this operation.
System Action:	The job is terminated.
User Response:	Make the proper entry in Factor 1 and resubmit the job.

RG216–FACTOR 1 ENTRY IN COL 18-27 INVALID FOR THIS OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	An entry must not be specified in Factor 1 for this operation.
System Action:	The job is terminated.
User Response:	Make Factor 1 blank for this operation and resubmit the job.

RG217–FACTOR 2 ENTRY IN COL 33-43 MISSING

Code:	T–Terminal
Specification Type:	C
Explanation:	No entry specified in Factor 2 for this operation.
System Action:	The job is terminated.
User Response:	Make the proper entry in Factor 2 and resubmit the job.

RG218–FACTOR 2 ENTRY IN COL 33-42 INVALID FOR THIS OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	An entry must not be specified in Factor 2 for this operation.
System Action:	The job is terminated.
User Response:	Make Factor 2 blank for this operation, and resubmit the job.

RG219–RESULT FIELD ENTRY IN COL 43-48 MISSING

Code:	T–Terminal
Specification Type:	C
Explanation:	No entry specified in the Result Field for this operation.
System Action:	The job is terminated.
User Response:	Make the proper entry in the Result Field for this operation and resubmit the job.

RG220–RESULT FIELD ENTRY IN COL 43-48 INVALID FOR THIS OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	An entry must not be specified in Result Field for this operation.
System Action:	The job is terminated.
User Response:	Make the Result Field blank for this operation and resubmit the job.

RG221–RESULT FIELD LENGTH MAY NOT BE LARGE ENOUGH

Code:	W–Warning
Specification Type:	C
Explanation:	The result field specified may not be large enough to hold the largest possible result obtained in the calculation operations specified.
System Action:	No action taken.
User Response:	Check to make sure the result field specified is large enough. If it is not, make it larger and resubmit the job.

RG223–SUBROUTINE SPECIFICATIONS ARE THE ONLY CALCULATION SPECIFICATIONS

Code:	T–Terminal
Specification Type:	C
Explanation:	Subroutine specifications do not follow detail and total calculations.
System Action:	The job is terminated.
User Response:	Place detail and total calculations before subroutine operations. Resubmit the job.

RG224–A ZERO CONSTANT INVALID AS DIVISOR IN COL 33-42

Code:	T–Terminal
Specification Type:	C
Explanation:	The constant entered in Factor 2 of a DIV operation must not be zero.
System Action:	The job is terminated.
User Response:	Place the proper divisor in Factor 2 of the DIV operation and resubmit the job.

RG225–CONDITIONING INDICATORS IN COL 9-17 INVALID WITH TAG, BEGSR, ENDSR, OR RLABL OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	Conditioning indicators must not be specified in columns 9-17 for TAG, BEGSR, ENDSR, or RLABL operations.
System Action:	The job is terminated
User Response:	Make columns 9-17 blank for TAG, BEGSR, ENDSR, or RLABL operations. Resubmit the job.

RG226–A RESERVED WORD OTHER THAN PAGE INVALID

Code:	T–Terminal
Specification Type:	C, I, O
Explanation:	No reserved word other than PAGE can be specified as a result field. CONTD is a reserved word, for compatibility with other systems.
System Action:	The job is terminated.
User Response:	Make sure no reserved word other than PAGE is specified in columns 43-48 as the result field. Resubmit the job.

RG227–RESULT FIELD IN COL 43-48 IS A LOOK AHEAD FIELD OR CONSTANT

Code:	T–Terminal
Specification Type:	C
Explanation:	The result field must not be a look ahead field or a constant.
System Action:	The job is terminated.
User Response:	Make the proper result field entry in columns 43-48 and resubmit the job.

RG228–INVALID INDEX

Code:	T–Terminal
Specification Type:	C
Explanation:	Array index not specified properly. Index field name must contain a valid combination of characters. Index constant of field value must be a positive number which does not exceed the number of elements in the array and have zero decimal positions.
System Action:	The job is terminated.
User Response:	Make the proper array index entry and resubmit the job.

RG229–INDEXING INVALID FOR TABLES OR FIELDS

Code:	T–Terminal
Specification Type:	C
Explanation:	Indexing must be specified for arrays only.
System Action:	The job is terminated.
User Response:	Remove specifications for indexing tables or fields. Resubmit the job.

RG231–GOTO DOES NOT BRANCH TO A TAG

Code:	T–Terminal
Specification Type:	C
Explanation:	The label in Factor 2 of this GOTO operation is not the same as the label in Factor 1 of a TAG or ENDSR operation.
System Action:	The job is terminated.
User Response:	Make the label in Factor 2 of the GOTO operation the same as the label in Factor 1 of a TAG or ENDSR operation. Resubmit the job.

RG232–THIS NAME WAS PREVIOUSLY USED ON A TAG, BEGSR, OR ENDSR

Code:	T–Terminal
Specification Type:	C
Explanation:	The label in Factor 1 was previously specified in another TAG, BEGSR, or ENDSR operation.
System Action:	The job is terminated.
User Response:	Make the label in Factor 1 of each TAG, BEGSR, and ENDSR operation unique. Resubmit the job.

RG233–CONFIGURATION, COLUMN 15, CONTAINS AN ENTRY OTHER THAN P, S, M, OR BLANK. IF CONTROL/TRIBUTARY, COLUMN 17, IS BLANK, ASSUME SWITCHED NETWORK; IF COLUMN 17 IS NOT BLANK, ASSUME MULTIPOINT NETWORK

Code:	T–Terminal
Specification Type:	T
Explanation:	The configuration entry in column 15 of your telecommunications specifications is not P, S, M or blank.
System Action:	The job is terminated.
User Response:	Make the proper entry (P, S, M, or blank) in column 15 and resubmit the job.

RG234–TRANSMITTER/RECEIVER, COLUMN 16, DOES NOT CONTAIN T OR R

Code:	T–Terminal
Specification Type:	T
Explanation:	The type of station entry in column 16 is neither T nor R.
System Action:	The job is terminated.
User Response:	Enter T (for a transmitter station) or R (for a receiver station) and resubmit the job.

RG235–CONTROL/TRIBUTARY, COLUMN 17, CONTAINS A CHARACTER OTHER THAN T OR BLANK. IF THIS IS A SWITCHED OR POINT-TO-POINT NETWORK, COLUMN 15, ASSUME BLANK; IF MULTIPOINT, ASSUME T

Code:	W–Warning
Specification Type:	T
Explanation:	The type of control entry in column 17 is neither T nor blank.
System Action:	Blank is assumed if this is a switched network or a point-to-point leased line; T is assumed if this is a multipoint leased line.
User Response:	To avoid this message when this job is run again, enter T in column 17 for tributary on multipoint network. Leave column 17 blank if switched line or point-to-point line is used.

RG236–ASCII/EBCDIC, COLUMN 18, IS NOT U, A, E, OR BLANK. ASSUME EBCDIC

Code:	W–Warning
Specification Type:	T
Explanation:	The type of code entry in column 18 is not U or A for ASCII, or E or blank for EBCDIC.
System Action:	EBCDIC is assumed.
User Response:	If the assumption was wrong, make the proper entry and resubmit the job.

RG237–TRANSPARENT FEATURE, COLUMN 19, IS NOT Y, N, OR BLANK. ASSUME NO TRANSPARENCY

Code:	W–Warning
Specification Type:	T
Explanation:	The entry in column 19 is not Y for transparency or N or blank for no transparency.
System Action:	No transparency is assumed.
User Response:	If the assumption was wrong, make the proper entry and resubmit the job.

RG238–AUTOCALL/AUTOANSWER, COLUMN 20, IS NOT E, S, M, A, B, OR BLANK. COLUMNS 21-31 WILL BE IGNORED

Code:	T–Terminal
Specification Type:	T
Explanation:	The entry in column 20 is not E, S, M, A, B, or blank.
System Action:	Entries in columns 21-31 are ignored; the job is terminated.
User Response:	Make the proper entry in column 20 and resubmit the job.

RG239–ENTRY FOR DIAL NUMBER, COLUMNS 21-31, IS NOT VALID FOR THE AUTOCALL/AUTOANSWER ENTRY IN COLUMN 20

Code:	T–Terminal
Specification Type:	T
Explanation:	The entry in columns 21-31 is not valid for the entry in column 20.
System Action:	The job is terminated.
User Response:	Enter dial number in columns 21-31 if the entry in column 20 is E; enter symbolic name in columns 21-31 if the entry in column 20 is S. Resubmit the job.

RG240–IDENTIFICATION TYPE FOR THIS STATION, COLUMN 32, IS NOT S, E, OR BLANK. COLUMNS 33-39 WILL NOT BE CHECKED

Code:	W–Warning
Specification Type:	T
Explanation:	The station identification entry in column 32 is not S, E, or blank.
System Action:	Columns 33-39 will not be checked for a valid entry.
User Response:	If the assumption is not correct, make the identification entry in column 32 (S, E, or blank) that properly describes this station and resubmit the job.

RG241–IDENTIFICATION FOR THIS STATION, COLUMNS 33-39 CONTAINS AN INVALID ENTRY FOR THE ID TYPE INDICATED IN COLUMN 32

Code:	T–Terminal
Specification Type:	T
Explanation:	The identification entry in columns 33-39 is invalid for the identification type specified in column 32.
System Action:	The job is terminated.
User Response:	Enter identification sequence in columns 33-39 if column 32 contains an E; enter symbolic name in columns 33-39 if column 32 contains an S. Resubmit the job.

RG242–IDENTIFICATION TYPE FOR THE REMOTE STATION, COLUMN 40, IS NOT S, E, OR BLANK. COLUMNS 41-47 WILL NOT BE CHECKED

Code:	W–Warning
Specification Type:	T
Explanation:	The identification entry in column 40 is not S, E, or blank.
System Action:	Columns 41-47 will not be checked for an entry.
User Response:	Make the identification entry in column 40 (S, E, or blank) that properly describes the remote station. Resubmit the job.

RG243–IDENTIFICATION FOR REMOTE STATION, COLUMNS 41-47, CONTAINS AN INVALID ENTRY FOR THE ID TYPE GIVEN IN COLUMN 40

Code:	T–Terminal
Specification Type:	T
Explanation:	The identification entry specified for a remote station in columns 41-47 is invalid for the identification type (column 40).
System Action:	No action taken.
User Response:	Be sure the entry in columns 41-47 is valid for the identification type (S, E, or blank) specified in column 40. Resubmit the job.

RG244–INVALID REMOTE TERMINAL SPECIFIED, COLUMNS 48-51

Code:	T–Terminal
Specification Type:	T
Explanation:	The entry in columns 48-51 is not a valid remote terminal.
System Action:	The job is terminated.
User Response:	Specify a valid remote terminal and resubmit the job.

RG245–ITB, COLUMN 52, IS NOT I OR BLANK ASSUME I

Code:	W–Warning
Specification Type:	T
Explanation:	The entry in column 52 is neither I nor blank.
System Action:	I is assumed.
User Response:	If the assumption was wrong, leave column 52 blank and resubmit the job.

RG246–PERMANENT ERROR INDICATOR, COLUMNS 53-54, IS INVALID

Code:	T–Terminal
Specification Type:	T
Explanation:	The indicator specified in columns 53-54 is not 01-99, L1-L9, LR, or H1-H9.
System Action:	The indicator is ignored and the job is terminated.
User Response:	Make the proper entry in columns 53-54 and resubmit the job.

RG247–WAIT TIME, COLUMNS 55-57, IS INVALID. ASSUME SYSTEM CONVENTION FOR TIMEOUT, 180 SECONDS

Code:	W–Warning
Specification Type:	T
Explanation:	The wait time entry specified in columns 55-57 is not 1-999 or blank.
System Action:	System convention for timeout, 180 seconds, will be assumed.
User Response:	If the assumption was not acceptable, make the proper entry (1-999 or blank) and resubmit the job.

RG248—RECORD AVAILABLE INDICATOR, COLUMNS 58-59, IS INVALID

Code:	T—Terminal
Specification Type:	T
Explanation:	The record available indicator specified in columns 58-59 is not 01-99, L1-L9, LR, or H1-H9.
System Action:	The indicator is ignored and the job is terminated.
User Response:	Make the proper entry in columns 58-59 and resubmit the job.

RG249—LAST FILE PROCESSED, COLUMN 60, IS NOT L OR BLANK

Code:	T—Terminal
Specification Type:	T
Explanation:	The last record processed entry in column 60 is not L or blank.
System Action:	The job is terminated.
User Response:	Enter L in column 60 if the BSC input file must be processed last; blank if not. Resubmit the job.

RG250—POLLING CHARACTERS, COLUMNS 61-62, CONTAIN AN INVALID CHARACTER FOR THE CODE TYPE ENTRY IN COLUMN 18

Code:	T—Terminal
Specification Type:	T
Explanation:	The polling characters specified in columns 61-62 are invalid, or are missing on a line configuration that requires them.
System Action:	The job is terminated.
User Response:	Make the proper entry in columns 61-62. (A list of the valid polling characters is included in the *IBM System/3 RPGII Telecommunications Reference Manual,* SC21-7507.) Resubmit the job.

RG251—ADDRESSING CHARACTERS, COLUMNS 63-64, ARE INVALID FOR THE CODE TYPE ENTRY IN COLUMN 18. THE ENTRY IS IGNORED

Code:	T—Terminal
Specification Type:	T
Explanation:	The addressing characters in columns 63-64 are invalid for the code type specified in column 18, or are missing on a line configuration that requires them.
System Action:	The job is terminated.
User Response:	Make the proper entry in columns 63-64. (A list of the valid addressing characters is included in the *IBM System/3 RPG II Telecommunications Reference Manual,* SC21-7507.) Resubmit the job.

RG252—NUMBER OF BSCA FILES SPECIFIED EXCEEDS MAXIMUM ALLOWED

Code:	T—Terminal
Specification Type:	T
Explanation:	The number of BSCA files specified exceeds the maximum allowed. Refer to *IBM System/3 RPG II Telecommunications Reference Manual,* SC21-7507, for the BSCA/other file combinations allowed.
System Action:	The job is terminated.
User Response:	Reduce number of BSCA files in the program.

RG254—REMOTE DEVICE SPECIFIED WHEN REMOTE TERMINAL IS BLANK OR INVALID; ASSUME COLUMNS 65-70 BLANK

Code:	W—Warning
Specification Type:	T
Explanation:	A remote device cannot be specified if a remote terminal is not specified.
System Action:	Blank is assumed for columns 65-70.
User Response:	If this assumption is wrong, specify a valid remote terminal and resubmit the job.

RG256–STACKER SELECT ENTRY IN COL 16 INVALID FOR OUTPUT DEVICE; ASSUME BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	Stacker select is only allowed for MFCU, MFCM, or 1442 files.
System Action:	Blank is assumed.
User Response:	Leave column 16 blank.

RG257–INVALID STACKER SELECT ENTRY IN COL 16; ASSUME DEFAULT STACKER

Code:	W–Warning
Specification Type:	O
Explanation:	Column 16 was not a blank, a number from 1-4 for MFCU or a 1 or 2 for 1442.
System Action:	On MFCU assume stacker 1 for file entered in primary hopper; assume stacker 4 for file entered in secondary hopper. On 1442 assume stacker 1.
User Response:	If the assumption is wrong, correct column 16 and resubmit the job.

RG258–SPACE AND/OR SKIP ENTRIES IN COL 17-22 INVALID FOR DEVICE

Code:	W–Warning
Specification Type:	O
Explanation:	The space and/or skip entries in columns 17-22 are invalid for the device. PRINT84 requires space after of at least one.
System Action:	Blank is assumed for invalid space and/or skip entries. A space after of one is assumed for PRINT84.
User Response:	To avoid this message when the job is run again, leave columns 17-22 blank for all devices except the console and the printer.

RG259–INVALID SKIP ENTRIES IN COL 19-22 OR GREATER THAN THE FORM LENGTH SPECIFIED, ASSUME BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	The skip entries in columns 19-22 are not specified properly or they exceed the form length in your line counter specifications.
System Action:	Blanks are assumed.
User Response:	If this assumption was wrong, make the proper skip entries and resubmit the job.

RG260–INVALID SPACE ENTRIES IN COL 17-18; ASSUME SPACE 1 AFTER OR BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	The space entries in columns 17-18 are not a number from 0 to 3 or blank.
System Action:	If space and skip before entries are invalid and the skip after entry is blank, a space after of 1 is assumed. When skip and space before entries are valid but space after is not, space after is assumed blank.
User Response:	If the assumption was wrong, make the proper space entries in columns 17-18 and resubmit the job.

RG261–FETCH OVERFLOW ENTRY IN COL 16 INVALID FOR DEVICE; ASSUME BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	The fetch overflow entry specified in column 16 is invalid for the device
System Action:	Blank is assumed; therefore, no fetch overflow is done.
User Response:	To avoid the message when the job is run again, specify fetch overflow for printer files only.

RG262–OVERFLOW INDICATOR INVALID FOR AN EXCPT RECORD

Code:	T–Terminal
Specification Type:	O
Explanation:	An overflow indicator must not be specified for an exception record (E in column 15).
System Action:	The job is terminated.
User Response:	Remove overflow indicators from exception output lines. Resubmit the job.

RG263–FETCH OVERFLOW INVALID WITH OVERFLOW INDICATOR ENTERED IN COL 23-31; ASSUME NO FETCH

Code:	W–Warning
Specification Type:	O
Explanation:	An overflow indicator and fetch overflow (F in column 16) must not be specified on the same output line.
System Action:	Blank in column 16 is assumed; therefore, no fetch overflow is done.
User Response:	If this assumption was wrong, make the proper fetch overflow specification and resubmit the job.

RG264–OVERFLOW INDICATOR USED IS NOT ASSIGNED TO THIS FILE

Code:	T–Terminal
Specification Type:	O
Explanation:	The overflow indicator specified was not assigned to this file in your file description specifications.
System Action:	The job is terminated.
User Response:	Assign the overflow indicator to this file in file description specifications. Resubmit the job.

RG265–1P INDICATOR INVALID WITH TOTAL OR EXCPT RECORDS

Code:	W–Warning
Specification Type:	O
Explanation:	First page (1P) indicator must not be specified for total or exception records.
System Action:	No action taken.
User Response:	To avoid this message when this job is run again, specify the 1P indicator with heading and detail records only.

RG266–FETCH OVERFLOW INVALID WITH 1P INDICATOR, ASSUME NO FETCH OVERFLOW

Code:	W–Warning
Specification Type:	O
Explanation:	A fetch overflow line (F in column 16) must not be conditioned by the 1P indicator.
System Action:	No fetch overflow is assumed.
User Response:	To avoid this message when this job is run again, remove the 1P indicator from lines in which fetch overflow is specified; or if the assumption was wrong, remove the 1P indicator.

RG267–1P INDICATOR INVALID FOR A COMBINED FILE

Code:	T–Terminal
Specification Type:	O
Explanation:	The 1P indicator must not be specified for records in a combined file.
System Action:	The job is terminated.
User Response:	Specify the 1P indicator to condition records in an output file only.

RG268–SPECIFIED OR IMPLIED SPACE BEFORE OF ZERO IS INVALID FOR CONSOLE FILE. ASSUME SPACE BEFORE OF ONE

Code:	W–Warning
Specification Type:	O
Explanation:	The console forces one space before printing. A zero or blank entry in space before will be defaulted to one.
System Action:	Space before of one is assumed.
User Response:	To avoid this message specify at least one for Space Before.

RG269–INVALID INDICATORS USED IN AN AND RELATIONSHIP WITH 1P

Code:	T–Terminal
Specification Type:	O
Explanation:	Only external indicators (U1-U8) can be specified in an AND relationship with the 1P indicator.
System Action:	The job is terminated.
User Response:	Specify the 1P indicator in an AND relationship with external indicators only. Resubmit the job.

RG270–END POSITION ENTRY IN COL 40-43 FOR CONSTANT, EDIT WORD, FIELD, OR ARRAY EXCEEDS RECORD LENGTH

Code:	T–Terminal
Specification Type:	O
Explanation:	The end position entry in columns 40-43 exceeds the records length specified in your file description specifications.
System Action:	The job is terminated.
User Response:	Make the proper end position entry in columns 40-43; it must be equal to or less than the record length. Resubmit the job.

RG271–LENGTH OF ARRAY, ARRAY ELEMENT, OR FIELD EXCEEDS RECORD LENGTH

Code:	T–Terminal
Specification Type:	O
Explanation:	Length specified for array, array element, or field exceeds the record length specified in your file description specifications.
System Action:	The job is terminated.
User Response:	Make the proper entry; it must be equal to or less than the record length or increase the record length entry to handle the length. Resubmit the job.

RG272–END POSITION ENTRY IN COL 40-43 FOR CONSTANT, EDIT WORD, FIELD, OR ARRAY TOO LOW

Code:	T–Terminal
Specification Type:	O
Explanation:	The end position entry in columns 40-43 is too small to allow the first field, array, or array element to be written, printed, or punched in its entirety.
System Action:	The job is terminated.
User Response:	Make the end position entry large enough for the field, array, or array element to be written, printed, or punched. Resubmit the job.

RG273–OUTPUT INDICATORS IN COL 23-31 MISSING OR ALL NEGATIVE

Code:	W–Warning
Specification Type:	O
Explanation:	No output indicators are specified in columns 23-31 or all those indicators specified are negative. Output may not be written when desired.
System Action:	No action taken.
User Response:	To avoid this message when this job is run again, specify at least one positive indicator to condition output records to ensure that output is written only when desired.

RG274–OUTPUT INDICATORS MISSING FOR AN AND OR OR LINE

Code:	W–Warning
Specification Type:	O
Explanation:	No conditioning indicators were specified in columns 23-31 or an AND or OR line.
System Action:	No action taken.
User Response:	To avoid this message when this job is run again, place the proper conditioning indicators in columns 23-31 of the AND or OR line. Resubmit the job.

RG276–INVALID EDIT CODE IN COL 38

Code:	T–Terminal
Specification Type:	O
Explanation:	The edit code specified in column 38 is not one of the following: 1-4, A-D, J-M, X, Y, Z, or blank.
System Action:	The job is terminated.
User Response:	Make the proper edit code entry in column 38 and resubmit the job.

RG277–INVALID EDIT WORD SIZE

Code:	T–Terminal
Specification Type:	O
Explanation:	The number of replaceable characters in this edit word (columns 45-70) exceed the length of the field to be edited.
System Action:	The job is terminated.
User Response:	Make the number of replaceable characters in the edit word equal to the length of the field to be edited. Resubmit the job.

RG278–EDIT CODES INVALID WITH FIELDS OTHER THAN UNPACKED NUMERIC FIELDS OR CONSTANTS OTHER THAN * OR $

Code:	T–Terminal
Specification Type:	O
Explanation:	Edit codes cannot be specified with edit words or with constants other than $ or * or with fields other than unpacked numeric fields.
System Action:	The job is terminated.
User Response:	Make the proper edit code entry and resubmit the job.

RG279–CONSTANTS IN COL 45-70 INVALID FOR X, Y, AND Z EDIT CODES

Code:	T–Terminal
Specification Type:	O
Explanation:	Edit codes X, Y, and Z must not be specified for edit words with '$' or '*' in columns 45-47.
System Action:	The job is terminated.
User Response:	Use either edit codes or edit words, but not both. Resubmit the job.

RG280–FIELD LENGTH FOR Y EDIT CODE LESS THAN 3 OR GREATER THAN 6

Code:	W–Warning
Specification Type:	O
Explanation:	Field edited by Y edit code is not from 3 to 6 characters long.
System Action:	If less than 3 characters long, field is not edited; if the field is more than 6 characters long, only the six low order digits are edited.
User Response:	Make the field to be edited by Y edit code 3 to 6 characters long, or change the edit code to prevent error message.

RG281–DECIMAL POSITIONS INVALID FOR FIELD EDITED BY Y CODE

Code:	T–Terminal
Specification Type:	O
Explanation:	Decimal positions must not be specified for field edited by Y code.
System Action:	The job is terminated.
User Response:	Specify no decimal positions for field edited by Y edit code, or use a different edit code. Resubmit the job.

RG282–NAME OF FIELD TO BE EDITED, BY CODE SPECIFIED IN COL 38, MISSING

Code:	T–Terminal
Specification Type:	O
Explanation:	An edit code is specified in column 38, but the name of the field to be edited is not entered in columns 32-37.
System Action:	The job is terminated.
User Response:	Specify the name of the field to be edited in columns 32-37 and resubmit the job.

RG283–INVALID FILE TYPE FOR OUTPUT RECORD

Code:	T–Terminal
Specification Type:	O
Explanation:	The file specified in columns 7-14 of your output specifications is not a combined file, update file, output file, or a file associated with ADD.
System Action:	The job is terminated.
User Response:	Make sure the file specified in output specifications is a combined file, update file, output file, or a file associated with ADD. Resubmit the job.

RG285–T OR E ENTRY IN COL 15 INVALID FOR COMBINED FILE

Code:	T–Terminal
Specification Type:	O
Explanation:	Column 15 does not contain an H or D for a combined file. Combined files cannot be written or stacker selected at total exception time.
System Action:	The job is terminated.
User Response:	Correct column 15 and resubmit the job.

RG286–*PRINT INVALID FOR DEVICE

Code:	T–Terminal
Specification Type:	O
Explanation:	*PRINT can only be used with the MFCU.
System Action:	The job is terminated.
User Response:	Remove *PRINT and resubmit the job.

RG287–OPERATION IN COL 40 INVALID FOR DEVICE

Code:	T–Terminal
Specification Type:	O
Explanation:	* (asterisk) in column 40 invalid for device.
System Action:	The job is terminated.
User Response:	Remove * (asterisk) and resubmit the job.

RG288–BLANK AFTER ENTRY IN COL 39 INVALID WITH RESERVED WORD OTHER THAN PAGE; ASSUME BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	Column 39 contains a B entry with a reserved word other than PAGE.
System Action:	Blank is assumed.
User Response:	Leave column 39 blank and resubmit the job.

RG289–*PRINT PRECEDES ALL FIELD NAMES AND CONSTANTS

Code:	T–Terminal
Specification Type:	O
Explanation:	*PRINT must be specified after all fields and constants are to be printed.
System Action:	The job is terminated.
User Response:	Correct the position of the *PRINT and resubmit the job.

RG290–*PLACE PRECEDES ALL FIELD NAMES AND CONSTANTS

Code:	T–Terminal
Specification Type:	O
Explanation:	When *PLACE is used, it must be specified after fields which are to be placed in different location.
System Action:	The job is terminated.
User Response:	Specify the fields to be moved before you specify *PLACE and resubmit the job.

RG291–INVALID ENTRIES IN COL 38, 39, OR 44-74 FOR OUTPUT OPERATION, ASSUME BLANKS

Code:	T–Terminal
Specification Type:	O
Explanation:	Blank after, edit codes, edit words and sterling cannot be specified for *PRINT or *PLACE.
System Action:	Blanks are assumed; the job is terminated.
User Response:	Leave columns 38, 39, and 44-74 blank for *PRINT and *PLACE. Resubmit the job.

RG292–TOO MANY AND/OR LINES

Code:	T–Terminal
Specification Type:	I or O
Explanation:	More than 20 AND/OR lines specified in your input or output specifications.
System Action:	The job is terminated.
User Response:	Make the number of AND/OR lines specified 20 or less. Resubmit the job.

RG293–BLANK AFTER SPECIFIED FOR A CONSTANT

Code:	W–Warning
Specification Type:	O
Explanation:	Blank after should not be specified for a constant since constants will be blanked out whenever they are used.
System Action:	Blank is assumed.
User Response:	If you do not want the message to come out, do not specify blank after (B in columns 39) for constants.

RG300–VALUE OF ARRAY INDEX EXCEEDS NUMBER OF ARRAY ELEMENTS

Code:	T–Terminal
Specification Type:	O
Explanation:	The array index specified exceeds the number of elements in the array.
System Action:	The job is terminated.
User Response:	Specify the proper array index value; the index must not exceed the number of array elements specified for the array in columns 36-39 of your extension specifications. Resubmit the job.

RG302–BLANK AFTER ENTRY IN COL 39 INVALID FOR LOOK AHEAD FIELD; ASSUME BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	Column 39 must be blank for a look ahead field.
System Action:	Blank is assumed.
User Response:	To avoid this message the next time the job is run, leave column 39 blank for look ahead field.

RG304–INVALID INDICATOR OR IMPROPER USE OF A VALID INDICATOR

Code:	T–Terminal
Specification Type:	I, C, or O
Explanation:	The indicator specified is invalid or used improperly.
System Action:	The job is terminated.
User Response:	If the indicator is invalid, make the proper indicator entry (only indicators 01-99, H1-H9, L1-L9, LR, U1-U8, OA-OG, OV, KA-KN, KP, KQ can be assigned). If the indicator has been used improperly, see the restrictions concerning proper use of indicators under *Operation Codes, Setting Indicators.* Resubmit the job.

RG305–INDICATOR ASSIGNED BUT NOT USED TO CONDITION OPERATIONS

Code:	W–Warning
Specification Type:	I, C, or O
Explanation:	The indicator was assigned but was not used to condition an operation.
System Action:	No action taken.
User Response:	Determine whether the indicator assigned is needed to condition any operation. If not, remove this indicator to avoid this message the next time this job is run.

RG306–INDICATOR USED TO CONDITION OPERATIONS BUT NOT ASSIGNED

Code:	T–Terminal
Specification Type:	I, C, or O
Explanation:	All indicators except LR, MR, 1P, and L0 must be assigned before they can be used to condition operations.
System Action:	The job is terminated.
User Response:	Make sure the indicator is assigned before it is used to condition operations. Resubmit the job.

RG307–FILE NAME DEFINED BUT NEVER USED. SPECIFICATION IS DROPPED

Code:	W–Warning
Specification Type:	F
Explanation:	A filename was defined in columns 7-14 but no input or output specifications exist for this file.
System Action:	Specification is dropped.
User Response:	To avoid this message when this job is run again, remove the filename in columns 7-14 in the fields not used in the program.

RG308–SEQUENCING INVALID FOR FILE WITH NO MATCH FIELD, ASSUME COLUMN 18 ON FILE DESCRIPTION SPECIFICATION BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Sequence checking specified in column 18 for a file with no match fields
System Action:	Assume column 18 is blank.
User Response:	Leave column 18 blank for files with no match fields.

RG309–SEQUENCE ENTRY IN COL 18 INVALID OR BLANK FOR FILES WITH MATCH FIELDS SPECIFIED, ASSUME FIRST VALID SEQUENCE OR A

Code:	W–Warning
Specification Type:	F
Explanation:	No sequence entry or an invalid sequence entry is specified in column 18 for a file with match fields.
System Action:	For a primary file, A is assumed. If no valid sequence entry is specified for a secondary file, the primary sequence value is assumed.
User Response:	If this assumption was wrong, make the proper sequence entry (A or D) in column 18 and resubmit the job.

RG310–EXTENSION CODE SPECIFIED IN COL 39 ON FILE DESCRIPTION SPECIFICATION FOR THIS FILE, BUT EXTENSION SPECIFICATION MISSING

Code:	T–Terminal
Specification Type:	F
Explanation:	(1) An extension code is specified (E in column 39) in your file description specifications, but no extension specifications were supplied. (2) An extension code is specified (L in column 39) in your file description specifications, but no line counter specifications were supplied.
System Action:	The job is terminated.
User Response:	You must either supply the proper extension specifications or delete the E or L for column 39 of your file description specifications if no extension specifications or line counter specifications are required for this program. Resubmit the job.

RG311–AN EXTENSION OR LINE COUNTER SPECIFICATION WAS PROVIDED FOR THIS FILE BUT AN EXTENSION CODE WAS NOT ENTERED IN COL 39 ON THE FILE DESCRIPTION SPECIFICATION

Code:	W–Warning
Specification Type:	F
Explanation:	Extension or line specifications were supplied, but no extension code (E in column 39 of file description) was specified.
System Action:	No action is taken.
User Response:	To avoid this message the next time this job is run, enter E or L in column 39 for this file.

RG312–STACKER SELECT NOT VALID WITH DUAL I/O; ASSUME BLANK

Code:	W–Warning
Specification Type:	I, O
Explanation:	Stacker select must not be specified for input or output files with dual I/O areas.
System Action:	Blank is assumed.
User Response:	To avoid this message on the next run, remove the dual I/O specification.

RG313–*PRINT SPECIFIED MORE THAN ONCE FOR A RECORD

Code:	W–Warning
Specification Type:	O
Explanation:	*PRINT may be used only once for each record.
System Action:	Extra *PRINT specifications are ignored.
User Response:	To avoid this message on the next run, remove the extra *PRINT statements.

RG314–FIELD, TABLE OR ARRAY NAME DEFINED BUT NEVER USED

Code:	W–Warning
Specification Type:	I, E, or C
Explanation:	A name is assigned to a field, table, or array but the field, table, or array is not used in the program.
System Action:	No action taken.
User Response:	To avoid this message when this job is run again, remove the field table, or array name if it is not used.

RG315–FIELD NAME USED BUT NEVER DEFINED OR TABLE NAME OR ARRAY ELEMENT USED AS AN ARRAY INDEX

Code:	T–Terminal
Specification Type:	C or O
Explanation:	(1) The field name is used in calculation or output operations but was not defined, or (2) a table name or array element is used as an array index.
System Action:	The job is terminated.
User Response:	(1) Be sure the field is defined before it is used in calculation or output operations, or (2) be sure that the array index is not a table name or array element. Resubmit the job.

RG316–INVALID DEFINITION FOR RESERVED WORD; ASSUME VALID DEFINITION

Code:	T–Terminal
Specification Type:	I or C
Explanation:	The field named by one of the RPG II reserved words is not specified according to the predefined format.
System Action:	The predefined format for this reserved word is assumed, but the job is terminated.
User Response:	Make the proper entry for the reserved word and resubmit the job.

RG317–NUMBER OF DECIMAL POSITIONS SPECIFIED EXCEEDS FIELD LENGTH

Code:	T
Specification Type:	I, C, or O
Explanation:	The number of decimal positions specified exceeds the field length.
System Action:	The job is terminated.
User Response:	Make the proper decimal position entry, it can be equal to or less than the field length. Resubmit the job.

RG318–MISSING A RECORD CONDITIONED BY 1P AND FORMS POSITIONING SPECIFIED ON CONTROL CARD

Code:	W–Warning
Specification Type:	H and O
Explanation:	Repetitive 1P output for forms positioning is specified in your control card specifications but 1P is not used to condition an output record
System Action:	No action taken.
User Response:	Use 1P to condition the proper output record to avoid this message the next time this job is run.

RG319–NO DATA FOR ALTERNATE COLLATING SEQUENCE, OR FILE TRANSLATION

Code:	T–Terminal
Specification Type:	H
Explanation:	Alternate collating sequence or file translation is specified in your header line, but no alternate collating sequence table or file translation table was supplied.
System Action:	The job is terminated.
User Response:	Provide the proper tables for alternate collating sequence or file translation or delete the specifications. Resubmit the job.

RG320–INVALID ALTERNATE COLLATING SEQUENCE DATA RECORD

Code:	T–Terminal
Specification Type:	Not applicable.
Explanation:	Columns 1-6 in your alternate collating sequence data records do not contain ALTSEQ.
System Action:	The job is terminated.
User Response:	Check your alternate collating sequence data records to make sure the data is specified properly; each record must contain ALTSEQ in columns 1-6. Resubmit the job.

RG321–INVALID, UNDEFINED, OR TABLE FILENAME ON FILE TRANSLATION DATA RECORD

Code:	T–Terminal
Specification Type:	Not applicable
Explanation:	The entry in columns 1-8 of the file translation data record is invalid, not previously defined, or is a table filename.
System Action:	The job is terminated.
User Response:	Make the entry in columns 1-8 of each file translation data record a filename previously defined in file description specifications or the characters *FILESȸȸ (ȸ = blank). The entry must not be a table filename. Resubmit the job.

RG322–ALTERNATE COLLATING SEQUENCE OR FILE TRANSLATION DATA INVALID

Code: T–Terminal
Specification Type: Not applicable
Explanation: The data supplied for alternate collating sequence or file translation is invalid (not 0-9 and A-F).
System Action: The job is terminated.
User Response: Make the data specified for alternate collating sequence or file translation consist of the characters A-F and 0-9. Resubmit the job.

RG323–MULTIPLY DEFINED TABLES/ARRAYS

Code: T–Terminal
Specification Type: E
Explanation: The table/array is multiply defined.
System Action: The job is terminated.
User Response: Make sure that all tables/arrays are defined only once.

RG324–TOTAL LENGTH OF ALL CONTROL OR ALL MATCHING FIELDS EXCEEDS 144 CHARACTERS

Code: T–Terminal
Specification Type: I
Explanation: The total length of all control or all matching fields is too large.
System Action: The job is terminated.
User Response: Make the total length of all matching fields (M1-M9) or all control fields L1-L9) equal to or less than 144. Resubmit the job.

RG325–ALL INPUT, UPDATE, AND COMBINED FILES CONDITIONED BY EXTERNAL INDICATORS

Code: W–Warning
Specification Type: I
Explanation: When all input, update, and combined files are conditioned by external indicators (U1-U8), be sure all indicators are not off. If they are all off, the job will not be done.
System Action: No action taken.
User Response: When all input, update, or combined files are conditioned by external indicators, be sure all indicators are not off.

RG326–COMPILE-TIME TABLES SPECIFIED NO DATA FOUND

Code: T–Terminal
Specification Type: Not applicable
Explanation: Compile time table specified (From filename in columns 11-18 of extension specifications blank), but no table input records were supplied after the source program.
System Action: The job is terminated.
User Response: For compile time tables, supply the table input records immediately after the source program. Resubmit the job.

RG327–SPLIT CONTROL FIELDS SPECIFIED MAY NOT HAVE PARTS THAT ARE PACKED

Code: T–Terminal
Specification Type: I
Explanation: All parts of a split control field must be either packed or unpacked.
System Action: The job is terminated.
User Response: Make proper entries so that all parts of the split control field are either packed or unpacked. Resubmit the job.

RG329–PACKED OR BINARY DATA NOT VALID FOR DEVICE

Code: W–Warning
Specification Type: I or O
Explanation: Packed or binary data should be specified only for disk, tape, BSCA, MFCM, 2501, and 1442 files.
System Action: Data errors may occur if program is executed.
User Response: Specify packed or binary data for disk, tape, BSCA, MFCM, 2501, and 1442 files only. Resubmit the job.

RG330–ALPHAMERIC FIELD SPECIFIED AS PACKED OR BINARY

Code:	T–Terminal
Specification Type:	O
Explanation:	Packed data cannot be specified for alphameric fields.
System Action:	The job is terminated.
User Response:	Specify packed data for numeric fields only. Resubmit the job.

RG331–NO INPUT SPECIFICATIONS FOUND

Code:	T–Terminal
Specification Type:	Not applicable
Explanation:	No valid input specifications are supplied for this job.
System Action:	The job is terminated.
User Response:	Supply valid input specifications and resubmit this job.

RG332–SEQUENCE ERROR FOUND IN COMPILE TIME TABLE/ARRAY

Code:	T–Terminal
Specification Type:	Not applicable
Explanation:	Compile time table or array is not in the sequence specified in columns 45 or 57.
System Action:	The job is terminated.
User Response:	Make sure the data is in the sequence specified (A or D) in column 45 or 57. Resubmit the job.

RG333–TABLE/ARRAY FULL OR NO TABLE/ ARRAYS FOR FOLLOWING DATA

Code:	W–Warning
Specification Type:	Not applicable
Explanation:	Either too much data is supplied for the table or array or no table or array is defined for the data supplied.
System Action:	No more data is accepted for tables or arrays.
User Response:	Make sure the data supplied does not exceed the maximum table size or that a table or array is defined for the data you supply. Resubmit the job.

RG334–SHORT TABLE

Code:	W–Warning
Specification Type:	Not applicable
Explanation:	The number of entries supplied is less than the maximum number of entries the table can contain.
System Action:	The remaining entries are filled with blanks or zeros.
User Response:	None required.

RG335–EDIT WORD SPECIFIED WITH OTHER THAN UNPACKED NUMERIC FIELDS

Code:	T–Terminal
Specification Type:	O
Explanation:	Edit words are allowed only with unpacked numeric fields.
System Action:	The job is terminated.
User Response:	Specify edit words for unpacked numeric fields only. Resubmit the job.

RG337–INCORRECT SPECIFICATION OF EXIT AND/OR RLABL STATEMENTS

Code:	T–Terminal
Specification Type:	C
Explanation:	The RLABL operation code does not immediately follow an EXIT operation; or, for SUBR89 and SUBR95, there is an incorrect number of RLABL operations; or the RLABL operations are coded incorrectly.
User Response:	Correct the errors and resubmit the job.

RG338–SUBR MUST BE USED WITH EXIT OP CODE

Code:	T–Terminal
Specification Type:	C
Explanation:	The entry specified in Factor 2 of an EXIT operation does not start with SUBR.
System Action:	The job is terminated.
User Response:	Make sure the subroutine name in Factor 2 starts with SUBR. Resubmit the job.

RG339–AN OUTPUT REFERENCE IS REQUIRED FOR EACH COMBINED OR UPDATE FILE, OR IF ADD IS SPECIFIED

Code:	T–Terminal
Specification Type:	F
Explanation:	The proper output specifications have not been specified for the combined or update file or file that has add specified.
System Action:	The job is terminated.
User Response:	Specify the proper output specifications for the combined or update file. A table output specification will meet the requirements for a combined file. Resubmit the job.

RG340–CONTROL/TRIBUTARY, COLUMN 17, CONTAINS A BLANK FOR A MULTIPOINT LINE. ASSUME T

Code:	W–Warning
Specification Type:	T
Explanation:	Column 17 was left blank for a multipoint line (M in column 15).
System Action:	T is assumed.
User Response:	To avoid this message when this job is run again, enter a T in column 17, or change the configuration entry in column 15.

RG341–CONTROL/TRIBUTARY, COLUMN 17, CONTAINS A T FOR A SWITCHED OR A POINT TO POINT NETWORK. ASSUME BLANK

Code:	W–Warning
Specification Type:	T
Explanation:	Column 17 contains a T for a point-to-point network (P in column 15).
System Action:	Blank is assumed.
User Response:	To avoid this message when this job is run again, leave column 17 blank, or change the configuration entry in column 15.

RG342–TRANSPARENT MODE IS SPECIFIED, COLUMN 19, WHEN ASCII CONTROL CHARACTERS, COLUMN 18, ARE TO BE USED

Code:	T–Terminal
Specification Type:	T
Explanation:	The transparent mode cannot be specified on an adapter using ASCII data link characters.
System Action:	The job is terminated.
User Response:	Make the proper entry in column 19 and resubmit the job.

RG343–AUTOCALL/AUTOANSWER, COLUMN 20, IS NOT BLANK FOR NON-SWITCHED NETWORK

Code:	T–Terminal
Specification Type:	T
Explanation:	Column 20 contains an entry for a network that is not switched.
System Action:	The job is terminated.
User Response:	Leave column 20 blank for a network that is not switched. Resubmit the job.

RG344–SYMBOL FOR DIAL NUMBER, COLUMNS 21-31, IS AN ARRAY

Code:	T–Terminal
Specification Type:	T
Explanation:	An array name was used as the dial number.
System Action:	The job is terminated.
User Response:	Enter the table element or field name to be used as the dial number in columns 21-31. If you want to use an array element as the dial number, you must use calculation specifications to move the contents of the array element into the field you specify in columns 21-31. Resubmit the job.

RG345–FIELD OR TABLE HOLD AREA FOR THE DIAL NUMBER WAS NOT DEFINED AS NUMERIC

Code:	T–Terminal
Specification Type:	T
Explanation:	The field or table hold area for the dial number specified in columns 21-31 was not defined as numeric.
System Action:	The job is terminated.
User Response:	Define the field or table hold area for the dial number specified in columns 21-31 as numeric. Resubmit the job.

RG346–COLUMN 32 IS NOT BLANK FOR A NON-SWITCHED NETWORK

Code:	T–Terminal
Specification Type:	T
Explanation:	Column 32 was not left blank for a non-switched network.
System Action:	The job is terminated.
User Response:	Leave column 32 blank for a non-switched network and resubmit the job.

RG347–IDENTIFICATION FOR THIS STATION, COLUMNS 33-39, CONTAINS AN ARRAY

Code:	T–Terminal
Specification Type:	T
Explanation:	An array name was used as the station identification.
System Action:	The job is terminated.
User Response:	Enter the table element or field name to be used as the station identification in columns 33-39. If you want to use an array element as the station identification, you must use calculation specifications to move the contents of the array element into the field you specify in columns 33-39. Resubmit the job.

RG348–COLUMN 40 IS NOT BLANK FOR A NON-SWITCHED NETWORK

Code:	T–Terminal
Specification Type:	T
Explanation:	Column 40 was not left blank for a non-switched network.
System Action:	The job is terminated.
User Response:	Leave column 40 blank for a non-switched network and resubmit the job.

RG349–IDENTIFICATION FOR THE REMOTE STATION, COLUMNS 41-47, CONTAINS AN ARRAY

Code:	T–Terminal
Specification Type:	T
Explanation:	An array name was used as the remote station identification.
System Action:	The job is terminated.
User Response:	Enter the table element or field name to be used as the remote station identification in columns 41-47. If you want to use an array element as the remote station identification, you must use calculation specifications to move the contents of the array element into the field you specify in columns 41-47. Resubmit the job.

RG350–RECORD AVAILABLE INDICATOR IS PRESENT ON TRANSMIT FILE, OR IN A PROGRAM WITH ONLY 1 BSCA FILE. INDICATOR IS DROPPED

Code:	W–Warning
Specification Type:	T
Explanation:	A record available indicator was specified for a transmit file or in a program which has only one BSCA file.
System Action:	The indicator is ignored.
User Response:	Remove the record available indicator or define the other BSCA file if a transmit interspersed with a receive program is desired. Resubmit the job.

RG351–LAST FILE PROCESSED, COLUMN 60, IS NOT BLANK ON A TRANSMIT FILE OR A PRIMARY INPUT FILE. THE ENTRY IS IGNORED

Code:	W–Warning
Specification Type:	T
Explanation:	L was entered in column 60 for a transmit file or for a primary output file.
System Action:	The entry is ignored.
User Response:	Remove the L from column 60 if the file is a transmit file. If it is a primary input file, remove the L or change the file designation to secondary. Resubmit the job.

RG352–POLLING CHARACTERS WERE GIVEN ON OTHER THAN A TRANSMIT FILE ON A MULTIPOINT NETWORK. THE ENTRY IS IGNORED

Code:	W–Warning
Specification Type:	T
Explanation:	Polling characters are specified in columns 61-62 for a file other than a transmit file on a multipoint network.
System Action:	The entry in columns 61-62 is ignored.
User Response:	To avoid this message when this job is run again, remove the entry from columns 61-62.

RG353–THERE IS AN ENTRY IN THE ADDRESSING CHARACTERS, COLUMNS 63-64, ON A FILE THAT IS NOT A MULTIPOINT RECEIVER FILE. THE ENTRY IS IGNORED

Code:	W–Warning
Specification Type:	T
Explanation:	Addressing characters are specified in columns 63-64 for a file that is not a multipoint receiver file.
System Action:	The entry in columns 63-64 is ignored.
User Response:	To avoid this message when this job is run again, remove the entry from columns 63-64.

RG354–CORRESPONDING FILE DESCRIPTION SPEC FILE IS NOT A BSC FILE

Code:	T–Terminal
Specification Type:	T
Explanation:	A BSC device entry was not made for this file on the File Description sheet.
System Action:	The job is terminated.
User Response:	Make a BSC device entry for this file on the File Description sheet. Resubmit the job.

RG355–A CONVERSATIONAL FILE WAS DEFINED WHEN NO CONVERSATIONAL FILE IS ALLOWED

Code:	T–Terminal
Specification Type:	T
Explanation:	A conversational file is not allowed with 2770/2780.
System Action:	The job is terminated.
User Response:	Correct the telecommunications specification and resubmit the job.

RG356–PACKED FIELD OR BINARY FIELD SPECIFIED IN A FILE WITHOUT THE TRANSPARENT FEATURE

Code:	T–Terminal
Specification Type:	T
Explanation:	A packed or binary field was specified for a file that does not have the transparent feature.
System Action:	The job is terminated.
User Response:	Be sure packed or binary fields are only specified for files with the transparent feature. Resubmit the job.

RG357–THE FILE CORRESPONDING TO THIS TRANSMITTER SPECIFICATION IS NOT A COMBINED OR AN OUTPUT FILE ON THE FILE DESCRIPTION SPECIFICATION

Code:	T–Terminal
Specification Type:	T
Explanation:	The transmitter file was not defined as a combined or output file on the File Description sheet.
System Action:	The job is terminated.
User Response:	Define the transmitter file as a combined file or an output file on the File Description sheet. Resubmit the job.

RG358–CORRESPONDING FILE DESCRIPTION SPEC FILE IS NOT DEFINED AS A COMBINED OR AN INPUT FILE FOR THIS RECEIVE FILE

Code:	T–Terminal
Specification Type:	T
Explanation:	The receive file was not defined on the File Description sheet as a combined file or as an input file.
System Action:	The job is terminated.
User Response:	Define the receive file as a combined file or as an input file on the File Description sheet. Resubmit the job.

RG359–BLOCKED RECORD DEFINED FOR A FILE WITH CONVERSATIONAL RESPONSES. ASSUME NO BLOCKING

Code:	W–Warning
Specification Type:	T
Explanation:	Blocked records must not be defined for a file with conversational responses.
System Action:	No blocking is assumed.
User Response:	To avoid this message when this job is run again, remove the blocked records specification.

RG360–THERE IS NO TELECOMMUNICATIONS SPEC FOR A FILE DEFINED AS A BSCA FILE ON THE FILE DESCRIPTION SPECS

Code:	T–Terminal
Specification Type:	T
Explanation:	No telecommunications specifications were supplied for a file that was described as a BSCA file on the File Description sheet.
System Action:	The job is terminated.
User Response:	Supply the proper telecommunications specifications and resubmit the job.

RG361–LOOK AHEAD FIELDS SPECIFIED FOR BSC FILE

Code:	T–Terminal
Specification Type:	T
Explanation:	Look ahead fields are not allowed for a BSC file.
System Action:	The job is terminated.
User Response:	Remove the look ahead specification for BSC file and resubmit the job.

RG362–MATCHING FIELDS DEFINED ON A TRANSMIT FILE WITH CONVERSATIONAL RESPONSE

Code:	T–Terminal
Specification Type:	T
Explanation:	Matching fields are not allowed for a transmit file with conversational responses.
System Action:	The job is terminated.
User Response:	Remove the matching fields definition for transmit file with conversational responses.

RG363–MATCHING FIELDS DEFINED FOR A FILE DESIGNATED TO BE THE LAST FILE PROCESSED IN COLUMN 60 OF THE TELECOMMUNICATIONS SPEC

Code:	T–Terminal
Specification Type:	T
Explanation:	Matching fields were defined for a file designated as the last file to be processed (L in column 60).
System Action:	The job is terminated.
User Response:	Remove the matching fields definition if the file was the last one to be processed, or remove the L entry in column 60. Resubmit the job.

RG364–FOR A TRANSMIT THEN RECEIVE BSCA PROGRAM, IF END-OF-FILE IS SPECIFIED FOR ANY INPUT FILE, E IS ASSUMED IN COLUMN 17 OF THE BSCA INPUT FILE

Code:	W–Warning
Specification Type:	T
Explanation:	E was entered in column 17 of some input files, but not for the BSCA file which has an L in column 60 of the Telecommunications sheet.
System Action'	E (EBCDIC) is assumed if end of file (E in column 17 of the File Description sheet) is specified for any input file the program uses.
User Response:	If the assumption was wrong, remove the L from column 60 or make the proper end of file entry on the Input sheet. Resubmit the job.

RG365–ITB IS SPECIFIED ON A FILE WITHOUT BLOCKED RECORDS. ITB IS DROPPED

Code:	W–Warning
Specification Type:	T
Explanation:	Intermediate block check (ITB) was specified for a file which does not have blocked records.
System Action:	The intermediate block check specification (I in column 52) is ignored.
User Response:	To avoid this message when this job is run again, remove the I from column 52 or define blocked records. Resubmit the job.

RG366–AUTOCALL/AUTOANSWER, COLUMN 20, IS BLANK FOR A SWITCHED NETWORK

Code:	T–Terminal
Specification Type:	T
Explanation:	Column 20 was left blank for a switched network.
System Action:	The job is terminated.
User Response:	Make the proper entry (M, E, S, A, or B) in column 20 for a switched network.

RG367–A TRANSMIT WITH CONVERSATIONAL RESPONSE FILE IS USED WITH FORCE OR READ OP CODE OR AS A PRIMARY FILE

Code.	T–Terminal
Specification Type:	T
Explanation:	(1) Neither the FORCE nor the READ operation code can be used with a transmit file which has conversational responses. (2) A transmit file with conversational responses cannot be a primary file.
System Action:	The job is terminated.
User Response:	Remove the FORCE or READ operation code or change the file designation from primary. Resubmit the job.

RG368–THE FIELD OR TABLE HOLD AREA USED FOR A STATION IDENTIFICATION, COLUMNS 33-39 OR COLUMNS 41-47, IS MORE THAN FIFTEEN CHARACTERS IN LENGTH, OR DIAL NUMBER IS MORE THAN TWELVE DIGITS

Code:	T–Terminal
Specification Type:	T
Explanation:	Either the field or table hold area used for a station identification (columns 33-39 or 41-47) contains more than 15 characters, or the dial number (columns 21-31) contains more than 12 digits.
System Action:	The job is terminated.
User Response:	Be sure that the field or table hold area used for a station identification is numeric and from 2 to 15 characters long. If you specify a dial number, be sure it is not more than 12 characters long. Resubmit the job.

RG369–WARNING: ONLY ONE I/O AREA WAS SPECIFIED ON A NON-CONVERSATIONAL FILE. THROUGHPUT MAY BE SLOW

Code:	W–Warning
Specification Type:	T
Explanation:	Because only one I/O area is specified for a non-conversational file, processing time is likely to be slow.
System Action:	No action taken.
User Response:	To avoid this message when the job is run again, specify dual I/O areas if the program size permits.

RG370–THE LINE CONFIGURATION AND LINE CONTROL ENTRIES, COLUMN 15 OR 17-47, ARE NOT THE SAME ON EACH TELECOMMUNICATIONS SPEC

Code:	T–Terminal
Specification Type:	T
Explanation:	The line configuration and line control entries (column 15 or 17-47) are not the same for each BSC file.
System Action:	The job is terminated.
User Response:	Make the same entries in columns 15 and 17-47 for each BSC file in the program. Resubmit the job.

RG371–WARNING: THE STATION IDENTIFICATION, COLUMNS 33-39 OR 41-47, HAS BEEN DEFINED AS ONLY ONE CHARACTER IN LENGTH. THE CHARACTER WILL BE DUPLICATED SO A TWO CHARACTER IDENTIFICATION WILL BE USED

Code:	W–Warning
Specification Type:	T
Explanation:	The station identification entry (columns 33-39 or 41-47) was specified as a 1-character field.
System Action:	The character is duplicated to provide a two-character identification field.
User Response:	If the assumption was wrong, specify a station identification which is at least 2 characters, but no more than 12 characters long. Resubmit the job.

RG372–A B IN COLUMN 37 OF THE CONTROL CARD IS AN INVALID ENTRY IN A BSCA PROGRAM

Code:	T–Terminal
Specification Type:	H
Explanation:	A B entry must not be specified in column 37 of the control card specifications for a BSCA program.
System Action:	The job is terminated.
User Response:	Remove the B entry from column 37 of the control card specifications and resubmit the job.

RG373–THE SAME FILENAME WAS GIVEN ON TWO TELECOMMUNICATIONS SPECS

Code:	T–Terminal
Specification Type:	T
Explanation:	A BSCA file must not have multiple definitions.
System Action:	The job is terminated.
User Response:	Specify a unique filename on each Telecommunications sheet used in this program. Resubmit the job.

RG374–ENTRY IN COL 16 INVALID

Code:	W–Warning
Specification Type:	F
Explanation:	The entry in column 16 of the file description specifications is not P, S, C, R, T, D, or blank.
System Action:	Blank is assumed if the file is an output file; otherwise, S is assumed.
User Response:	If the assumption is wrong, make the proper entry in column 16 and resubmit the job.

RG375–ID IN COL 75-80 OF CONTROL CARD MUST NOT BE BLANK WHEN C IS SPECIFIED IN COL 10, ASSUME BLANK IN COL 10

Code:	W–Warning
Specification Type:	H
Explanation:	A C is specified in column 10 of your control card specifications, but no program identification is specified in columns 75-80.
System Action:	Column 10 is assumed to be blank.
User Response:	When C is specified in column 10 of your control card specification, place the proper program name in columns 75-80. Resubmit the job.

RG376–INVALID NAME IN COLS 75-80 OF CONTROL CARD, ASSUME BLANK

Code:	W–Warning
Specification Type:	H
Explanation:	The entry in columns 75-80 of your header line is neither a valid RPG program name nor blanks.
System Action:	Blanks are assumed.
User Response:	If this assumption was wrong, make the proper program name entry and resubmit the job.

RG377–RAF, COLUMN 31, IS NOT ALLOWED ON A BSCA FILE

Code:	T–Terminal
Specification Type:	F
Explanation:	A BSCA file cannot be specified as a record address file.
System Action:	The job is terminated.
User Response:	Remove the record address file specification for a BSCA file and resubmit the job.

RG378–NO LINE COUNTER SPECIFICATION FOR THIS BSCA FILE, ASSUME PAGE SIZE-66, OVERFLOW LINE-60

Code:	W–Warning
Specification Type:	T
Explanation:	Entries must be specified if the page size and overflow line differ from assumed values.
System Action:	Page size of 66 is assumed; overflow line of 60 is assumed.
User Response:	Verify that page size of 66 is correct for this job.

RG379–MULTI-POINT INVALID WITH 2770 OR 2780

Code:	T–Terminal
Specification Type:	T
Explanation:	Column 15 must be P, S, or blank.
System Action:	The job is terminated.
User Response:	Correct column 15 and resubmit the job.

RG380–2770 AND 2780 CANNOT BE SPECIFIED IN THE SAME PROGRAM

Code:	T–Terminal
Specification Type:	T
Explanation:	Both 2770 and 2780 have been specified in the same job.
System Action:	The job is terminated.
User Response:	Specify either 2770 or 2780 but not both.

RG381–INVALID DEVICE SPECIFIED FOR THE REMOTE TERMINAL USED

Code:	T–Terminal
Specification Type:	T
Explanation:	Device specified in columns 65-70 is not a valid remote device.
System Action:	The job is terminated.
User Response:	Specify a valid device for the remote terminal used.

RG382–INVALID REMOTE DEVICE FOR FILE TYPE SPECIFIED

Code:	T–Terminal
Specification Type:	T
Explanation:	An output device was specified for an input file or an input device was specified for an output file.
System Action:	The job is terminated.
User Response:	Specify a valid remote device for the type of operation being performed and resubmit the job.

RG383–ITB AND TRANSPARENCY SPECIFIED FOR 2770. ITB IS DROPPED.

Code:	W–Warning
Specification Type:	T
Explanation:	When 2770 is specified, specify either ITB (column 52) or transparency (column 19) but not both.
System Action:	Blank is assumed for column 52 (ITB)
User Response:	Verify that the assumption is correct for this job.

RG388–FACTOR 1 MUST BE EITHER A FIELD NAME OR A LITERAL WHEN USED WITH THIS OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	Factor 1 can only be a field name or a literal when the DEBUG or SETLL operation is specified.
System Action:	The job is terminated.
User Response:	Make Factor 1 either a field name or a literal and resubmit the job.

RG390–SEQUENCE CHECKING IS NOT PERFORMED ON EXECUTION TIME ARRAYS

Code:	W–Warning
Specification Type:	E
Explanation:	Sequence must be specified if high or low LOKUP is to be done; however, no sequence checking is done at input time.
System Action:	A sequenced array is assumed.
User Response:	Be sure the array is in ascending or descending sequence.

RG391–A FIELD WITH A LENGTH GREATER THAN 8 CHARACTERS CANNOT BE USED IN FACTOR 1 WITH DEBUG OPERATION

Code:	T–Terminal
Specification Type:	C
Explanation:	The length of a Factor 1 field cannot be greater than eight characters when a DEBUG operation is specified.
System Action:	The job is terminated.
User Response:	Limit the length of the Factor 1 field to eight characters. Resubmit the job.

RG392–LAST ENTRY IN ONE OR MORE COMPILE TIME TABLE/ARRAYS WAS BLANK

Code:	W–Warning
Specification Type:	E
Explanation:	The compile time table/array contains fewer entries than the number of entries specified in columns 36-39 of the Extension specifications.
System Action:	A warning message is given.
User Response:	If the assumption was wrong, review your compile time tables/arrays and fill the table.

RG394–'ADD' IN COL 16-18 NOT ALLOWED ON AND/OR LINES, ASSUME BLANK

Code:	T–Terminal
Specification Type:	O
Explanation:	ADD was specified in columns 16-18 of an AND/OR line in output specifications.
System Action:	Blank is assumed, but the job is terminated.
User Response:	Remove the ADD entry from columns 16-18 of the AND/OR line and resubmit the job.

RG397–FILE DESCRIBED AS 'ADD' TYPE FILE, EACH OUTPUT LINE MUST HAVE 'ADD' IN COL 16-18. ASSUME 'ADD'

Code:	W–Warning
Specification Type:	O
Explanation:	The ADD function (A in column 66) was specified in the file description specifications for this file, but ADD was not specified in columns 16-18 of the Output sheet for each record type output line to be written.
System Action:	ADD in columns 16-18 is assumed.
User Response:	To avoid this message the next time this job is run, remove the A from column 66 of the file description specifications or specify ADD in columns 16-18 of the output specifications for each record type output line to be written.

RG398–COLS 54-59, INVALID FOR DEVICE, OR WRONG ENTRY, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F
Explanation:	Columns 54-59 contain an entry for a file which was not assigned to a SPECIAL device (SPECIAL in columns 40-46).
System Action:	Blank is assumed, but the job is terminated.
User Response:	Leave columns 54-59 blank for file not assigned to a SPECIAL device. Resubmit the job.

RG399–INVALID ENTRY IN COLS 54-59

Code:	T–Terminal
Specification Type:	F
Explanation:	The entry in columns 54-59 of your file description specifications for a SPECIAL file is neither SUBRxx (x = any alphabetic character) nor SRyzzz (y = one of 15 valid characters; z = one of 16 valid characters).
System Action:	The job is terminated.
User Response:	Enter the name of the user-written subroutine (SUBRxx) or IBM-written subroutine (SRyzzz) which will perform the input/output operations for the SPECIAL file. Resubmit the job.

RG400–INVALID MODE OF PROCESSING ENTRY IN COLUMN 28

Code:	T–Terminal
Specification Type:	F
Explanation:	The entry in column 28 is not R, L, or blank.
System Action:	R is assumed for valid file type or mode of processing; the job is terminated.
User Response:	Make proper mode of processing entry in column 28 and resubmit the job.

RG401–ONLY ONE TABLE/ARRAY PER FILENAME ALLOWED FOR THIS DEVICE

Code:	T–Terminal
Specification Type:	E
Explanation:	Only one table or array can be specified per file (except for a card file).
System Action:	The job is terminated.
User Response:	Specify only one table or array per file (except for card files) and resubmit the job.

RG403–INVALID LENGTH OF KEY FIELD IN COLUMN 29-30, ASSUME 03

Code:	T–Terminal
Specification Type:	F
Explanation:	The length of key field entry in columns 29-30 is not specified properly. The entry must be 29 or less for unpacked keys, 8 for packed keys.
System Action:	03 is assumed, but the job is terminated.
User Response:	Make the length of key field entry in columns 29-30 a valid key length. Resubmit the job.

RG404–INVALID RECORD ADDRESS TYPE ENTRY IN COLUMN 31, ASSUME A

Code:	T–Terminal
Specification Type:	F
Explanation:	The entry in column 31 is not A, I, P, or blank, or is incorrect for the file type specified.
System Action:	A is assumed; the job is terminated.
User Response:	Make the proper record address type entry in column 31 and resubmit the job.

RG405–INVALID KEY START LOCATION ENTRY IN COLUMNS 35-38, ASSUME 1

Code:	T–Terminal
Specification Type:	F
Explanation:	Columns 35-38 do not contain a number from 1-4096 for an indexed file.
System Action:	1 is assumed; the job is terminated.
User Response:	Make the proper key start location entry in columns 35-38 and resubmit the job.

RG406–INVALID CORE INDEX ENTRY IN COLS 60-65, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Columns 60-65 contain an incorrect number.
System Action:	Blank is assumed.
User Response:	If this assumption was wrong, make the proper core index entry in columns 60-65 and resubmit the job.

RG407–INVALID FILE ADDITION OR UNORDERED ENTRY IN COLUMN 66, ASSUME A

Code:	T–Terminal
Specification Type:	F
Explanation:	The file addition or unordered load entry in column 66 is not A, U, or blank.
System Action:	A is assumed; the job is terminated.
User Response:	Make the proper file addition or unordered load entry in column 66 and resubmit the job.

RG408–NUMBER OF EXTENTS ENTRY IN COLS 68-69 IS INVALID OR NOT ALLOWED WITH DEVICE, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F
Explanation:	Not allowed for device other than disk or for disk using shared I/O. For disk not using shared I/O, entry must be 01-50.
System Action:	Blank is assumed; the job is terminated.
User Response:	Make the proper entry in columns 68-69 and resubmit the job.

RG409–ENTRY OF K MADE IN COLUMN 31 FOR RECORD ADDRESS TYPE, ASSUME A

Code:	W–Warning
Specification Type:	F
Explanation:	An entry of K is not allowed in column 31 for record address type.
System Action:	A is assumed.
User Response:	If this assumption was wrong, make the proper entry in column 31 and resubmit the job.

RG410–EXTENSION SPECIFICATION SHEET BLANK

Code:	T–Terminal
Specification Type:	Not applicable
Explanation:	An E was specified in column 39 of a File Description sheet, but no Extension specifications were entered.
System Action:	The job is terminated.
User Response:	You must supply the proper extension specifications and resubmit the job.

RG411–RESERVED COLUMNS 71-74 ARE NOT BLANK

Code:	W–Warning
Specification Type:	T
Explanation:	Columns 71-74 on the Telecommunications Specifications are reserved and should be blank.
System Action:	Blanks are assumed.
User Response:	Leave columns blank.

RG450–BUFOFF SPECIFIED ON AN OUTPUT FILE

Code:	T–Terminal
Specification Type:	F
Explanation:	System/3 cannot create tapes with a block prefix.
System Action:	BUFOFF entry is ignored; job is terminated.
User Response:	Remove BUFOFF and resubmit the job.

RG451–CONTINUATION (K IN COL 53) INVALID FOR DEVICE

Code:	T–Terminal
Specification Type:	F
Explanation:	Continuation is only allowed on tape files, or SPECIAL files.
System Action:	Continuation is ignored; job is terminated.
User Response:	Remove the continuation (K in column 53) and resubmit the job.

RG452–ENTRY IN COL 54-59 OF A CONTINUATION CARD IS INVALID OR MISSING

Code:	T–Terminal
Specification Type:	F
Explanation:	For a tape file, the only valid entries in columns 54-59 of a continuation card are ASCII and BUFOFF. For a SPECIAL file you must enter a table or array name in columns 54-59. The table or array must be defined in the extension specifications. Blanks are invalid.
System Action:	The continuation card is ignored; job is terminated.
User Response:	Correct or remove the entry.

RG453–CONTINUATION ENTRY IN COL 54-59 IS REPEATED FOR A FILE, SECOND ENTRY IGNORED

Code:	W–Warning
Specification Type:	F
Explanation:	Each of the continuation entries ASCII and BUFOFF may appear only once for any one tape file. For a SPECIAL file only one continuation is allowed.
System Action:	The second usage of the entry is ignored.
User Response:	To avoid this message on the next run remove the repeated continuation entry.

RG454–INVALID BUFFER OFFSET SPECIFIED ON COL 60-65

Code:	T–Terminal
Specification Type:	F
Explanation:	The buffer offset must have a value between 0 and 99.
System Action:	The continuation card is ignored. The job is terminated.
User Response:	Correct the value in columns 60-65 and resubmit the job.

RG455–COLUMNS 7-52 AND 66-72 ARE NOT BLANK FOR A CONTINUATION LINE, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	If continuation is specified, these columns must be blank.
System Action:	Entries in columns 7-52 and 66-72 are ignored.
User Response:	If this assumption is incorrect, remove the continuation entries and resubmit the job.

RG456–RECORD LENGTH SPECIFIED FOR A TAPE FILE IS LESS THAN 18

Code:	T–Terminal
Specification Type:	F
Explanation:	The minimum record size allowed on tape files is 18 characters.
System Action:	The job is terminated.
User Response:	Correct the record length to 18 or greater and resubmit the job.

RG457–ENTRY IN COL 70 INVALID FOR DEVICE, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	An entry is allowed in column 70 only on tape files.
System Action:	The entry in column 70 is ignored.
User Response:	To avoid this message on the next run, leave column 70 blank.

RG458–BUFOFF SPECIFIED IN COL 54-59 FOR A NON-ASCII TAPE FILE, ASSUME ASCII

Code:	W–Warning
Specification Type:	F
Explanation:	The BUFOFF entry is valid only on files that are ASCII files.
System Action:	An ASCII file with BUFOFF is assumed.
User Response:	If this assumption is wrong, remove BUFOFF from columns 54-59 and resubmit the job.

RG459–COLUMNS 60-65 ARE NOT BLANK WHEN ASCII IS ENTERED IN COL 54-59

Code:	W–Warning
Specification Type:	F
Explanation:	If ASCII is specified, no entry is allowed in columns 60-65.
System Action:	The entry in columns 60-65 is ignored.
User Response:	To avoid this message on the next run, leave columns 60-65 blank.

RG460–INVALID ENTRY IN COL 53, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Valid entries are K or blank.
System Action:	Blank is assumed.
User Response:	To avoid this message on the next run, correct the entry in column 53.

RG461–INVALID ENTRY IN COL 70, ASSUME BLANK

Code:	W–Warning
Specification Type:	F
Explanation:	Valid entries are R, U, or N.
System Action:	Tape rewind information specified at job execution time assumed.
User Response:	Verify that the execution time rewind information will be adequate. If not, correct column 70 and resubmit the job.

RG462–CONTINUATION, K IN COL 53, INVALID FOR MAIN FILE DESCRIPTION LINE. ASSUME BLANK.

Code:	W–Warning
Specification Type:	F
Explanation:	K is valid only on a continuation file description specification.
System Action:	Blank is assumed.
User Response:	To avoid this message on the next run, leave column 53 blank.

RG500–FROM NAME INVALID OR MISSING FROM RA FILE

Code:	T–Terminal
Specification Type:	E
Explanation:	The From Filename entry in columns 11-18 is missing or not specified properly for an RA file.
System Action:	The job is terminated.
User Response:	Enter the proper record address filename in columns 11-18 and resubmit the job.

RG502–FROM FILENAME IS AN RA FILE THAT IS USED MORE THAN ONCE

Code:	T–Terminal
Specification Type:	E
Explanation:	The RA file named in columns 11-18 is used more than once in the extension specifications.
System Action:	The job is terminated.
User Response:	Since only one RA file is allowed in a program, either remove the extension specifications or correct the entry in columns 11-18.

RG503–TO FILENAME FOR A RA FILE TYPE IS EITHER: 1–NOT A PRIMARY, SECONDARY OR DEMAND FILE OR 2–IS MISSING, INVALID OR NON DISK FILE

Code:	T–Terminal
Specification Type:	E
Explanation:	The To Filename entry in columns 19-26 must be a primary or secondary disk file to be processed by an RA file.
System Action:	The job is terminated.
User Response:	Make the proper To Filename entry in columns 19-26 and resubmit the job.

RG504–TO FILENAME IS INCORRECT FILE TYPE

Code:	T–Terminal
Specification Type:	E
Explanation:	The filename specified in columns 19-26 is not an input, output, or update file.
System Action:	The job is terminated.
User Response:	Make sure the file named in columns 19-26 is an input, output, or update file. Resubmit the job.

RG510–LENGTH GIVEN FOR BINARY FIELD IS NOT 2 OR 4, ASSUME 2

Code:	T–Terminal
Specification Type:	I, O
Explanation:	Binary field length specified is neither 2 nor 4 bytes.
System Action:	The job is terminated.
User Response:	Make the length of the binary field either 2 or 4 bytes. Resubmit the job.

RG511–PACKED LENGTH GREATER THAN 8 FOR A FIELD, TABLE, OR ARRAY

Code:	T–Terminal
Specification Type:	I, O
Explanation:	The length specified for a packed field, table, or array is greater than 8.
System Action:	The job is terminated.
User Response:	Specify a length of 8 or less for a packed field, table, or array. Resubmit the job.

RG516–MORE THAN 7 AN/OR LINES SPECIFIED

Code:	T–Terminal
Specification Type:	C
Explanation:	More than 7 consecutive AN/OR line specified in the calculation specifications.
System Action:	The job is terminated.
User Response:	Specify up to 7 consecutive AN, OR, or AN/OR lines to condition an operation. Resubmit the job.

RG517–AN/OR LINES OUT OF ORDER

Code:	T–Terminal
Specification Type:	C
Explanation:	The line immediately following a line with an operation code is an AN/OR line.
System Action:	The job is terminated.
User Response:	Remove the AN/OR entry in columns 7-8 from the first line in an AN/OR group and resubmit the job.

RG518–NO INDICATORS GIVEN WITH AN/OR LINES

Code:	T–Terminal
Specification Type:	C
Explanation:	At least one indicator must be given in an AN or OR line.
System Action:	The job is terminated.
User Response:	Remove the specification in error or specify an indicator on the AN or OR line.

RG519–COLUMNS 18-59 ARE INVALID WITH AN/OR LINES OR OP CODE IS MISSING WITH INDICATORS PRESENT, ASSUME BLANK

Code:	T–Terminal
Specification Type:	C
Explanation:	Only the last line of a group of AN/OR lines can have entries in columns 18-59 or indicators are specified in columns 7-17, but no operation is specified in columns 28-32.
System Action:	The job is terminated.
User Response:	Make sure that entries are made only in columns 18-59 of the last line of a group of AN/OR lines or make the proper operation code entry in columns 28-32. Resubmit the job.

RG520–THIS LINE IS NOT AN AN/OR LINE AND PREVIOUS LINE HAS NO OP CODE; OR THIS LINE HAS NO INDICATORS AND NO OP CODE

Code:	T–Terminal
Specification Type:	C
Explanation:	This line is not an AN/OR line and previous line has no operation code specified.
System Action:	The job is terminated.
User Response:	If this line should be an AN/OR line, enter an AN/OR entry in columns 7-8; if this line should have had an operation code (an operation code must be entered in the last line of a group of AN/OR lines), make the proper operation code entry in columns 28-32. Resubmit the job.

RG521–MINUS INDICATOR IS NOT ALLOWED FOR TEST BIT OPERATION OF ONLY 1 BIT

Code:	W–Warning
Specification Type:	C
Explanation:	Columns 56-57 (Minus) must be blank when only one bit is specified for a TESTB operation.
System Action:	Blank is assumed.
User Response:	To avoid the message the next time this job is run, leave columns 56-57 blank.

RG522–ALL THREE RESULTING INDICATORS ARE THE SAME

Code:	W–Warning
Specification Type:	C
Explanation:	Usually the same indicator is used for only one or two of the conditions.
System Action:	The indicator specified will be set on each time the calculation is executed.
User Response:	Make sure the proper resulting indicator entries have been made in columns 54-59. If the entires were incorrect, resubmit the job.

RG523–A NEGATIVE FACTOR FOR THE SQUARE ROOT OPERATION IS NOT ALLOWED

Code:	T–Terminal
Specification Type:	C
Explanation:	The entry specified in Factor 2 of a SQRT operation is negative.
System Action:	The job is terminated.
User Response:	Make the entry in Factor 2 of a SQRT operation a positive value. Resubmit the job.

RG524–WHOLE ARRAYS ARE NOT ALLOWED AS FACTOR 1 WITH DISPLAY OR CHAIN OP CODE

Code:	T–Terminal
Specification Type:	C
Explanation:	The entry in Factor 1 of a DSPLY or CHAIN operation cannot be a whole array.
System Action:	The job is terminated.
User Response:	Enter the array named and index in Factor 1 of a DSPLY or CHAIN operation. Resubmit the job.

RG525–OPERATION CODE IS INVALID FOR DEVICE TYPE OR MODE OF PROCESSING

Code:	T–Terminal
Specification Type:	C
Explanation:	The CHAIN operation can only be specified for disk files processed randomly.
System Action:	The job is terminated.
User Response:	Make sure that CHAIN is only specified for disk files processed randomly.

RG541–FILE DESIGNATION IS INVALID FOR ADDROUT FILE, ASSUME R

Code:	T–Terminal
Specification Type:	F
Explanation:	The file designation entry in column 16 is not R for an ADDROUT file.
System Action:	The job is terminated.
User Response:	Enter an R in column 16 for the ADDROUT file and resubmit the job.

RG543–LENGTH OF KEY COL 29-30, OR LENGTH OF KEY AND KEY START LOCATION GREATER THAN RECORD LENGTH

Code:	T–Terminal
Specification Type:	F
Explanation:	The key field entry in columns 29-30 must be less than 29 characters and must be less than the record length. The sum of the key field starting location plus the key length must not exceed the record length.
System Action:	Key field length of 03 is assumed; key field starting location of 01 is assumed. The job is terminated.
User Response:	Make the proper key field length (columns 29-30) and key field starting location (columns 35-38) entries. Resubmit the job.

RG544–LENGTH OF RA OR KEY FIELD, COLS 29-30 BLANK OR INVALID, ASSUME 03

Code:	T–Terminal
Specification Type:	F
Explanation:	Columns 29-30 are blank or the entry specified is invalid for files that contain limits or for ADDROUT files.
System Action:	03 is assumed; the job is terminated.
User Response:	Make the entry in columns 29-30 a number from 1 to 29 for files that contain limits and for ADDROUT files. Resubmit the job.

RG545–RECORD LENGTH, COLS 24-27, NOT 18 FOR ADDROUT TAPE FILE. ASSUME 18.

Code:	W–Warning
Specification Type:	F
Explanation:	The record length of an ADDROUT tape file must be 18.
System Action:	18 is assumed.
User Response:	To avoid this message on the next run, correct the entry in columns 24-27 and resubmit the job.

RG548–FILE ADDITION IS INVALID FOR FILE OR DEVICE, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F, O
Explanation:	File addition (A in column 66) can be specified for sequential and indexed output files on disk only.
System Action:	The job is terminated.
User Response:	Make the proper file addition entry in column 66 and resubmit the job.

RG549—KEY FIELD START LOCATION IS BLANK OR EXCEEDS RECORD LENGTH

Code:	T—Terminal
Specification Type:	F
Explanation:	Columns 35-38 are blank or the entry specified exceeds the record length in your file description specifications.
System Action:	The job is terminated.
User Response:	Make the key field starting location entry (1-4096) in columns 35-38 equal to or less than the record length. Resubmit the job.

RG550—NO MORE THAN 20 FILE DESCRIPTION SPECS ALLOWED

Code:	T—Terminal
Specification Type:	F
Explanation:	More than 20 file description lines were specified.
System Action:	The job is terminated.
User Response:	Specify a maximum of 20 file description lines per program. Resubmit the job.

RG551—RECORD LENGTH MISSING OR INVALID FOR DISK FILE, ASSUME 256

Code:	T—Terminal
Specification Type:	F
Explanation:	The record length entry in columns 24-27 is missing.
System Action:	The job is terminated.
User Response:	Make the proper record length entry in columns 24-27; it can be a number from 1 to 4096. Resubmit the job.

RG552—FACTOR 1 AND RESULT FIELD MUST NOT BOTH BE BLANK WITH DSPLY OP CODE

Code:	T—Terminal
Specification Type:	C
Explanation:	Both the Result Field and Factor 1 were left blank on a DSPLY operation.
System Action:	The job is terminated.
User Response:	Make the proper entry under Factor 1 or the Result Field for the DSPLY operation and resubmit the job.

RG553—CORE INDEX IS INVALID FOR DEVICE TYPE OR MODE OF PROCESSING

Code:	W—Warning
Specification Type:	F
Explanation:	Core index can be specified in columns 60-65 for indexed disk files processed randomly. Core index should not be specified when using Shared I/O.
System Action:	Blank is assumed.
User Response:	Make the proper core index entry in columns 60-65.

RG554—ADD SPECIFIED ON THE FILE DESCRIPTION SPEC BUT ADD NOT REFERENCE ON OUTPUT

Code:	T—Terminal
Specification Type:	Not applicable
Explanation:	Column 66 contains an A, but record addition ADD in columns 16-18 is not specified in your output specifications.
System Action:	The job is terminated.
User Response:	Place ADD in columns 16-18 of your output specifications when A is specified in column 66 of file description. Resubmit the job.

RG555—NO ADD SPECIFIED ON FILE DESCRIPTION

Code:	T—Terminal
Specification Type:	Not applicable
Explanation:	ADD is specified in columns 16-18 of your output specifications, but the add function was not specified in file description specifications (column 66) for this file.
System Action:	The job is terminated.
User Response:	Place A in column 66 of your file description specifications when ADD is specified in columns 16-18 of the output specifications. Resubmit the job.

RG557–MASK FOR BIT OPERATION IS NOT 0-7

Code:	T–Terminal
Specification Type:	C
Explanation:	The mask specified for the bit operation is not 0-7.
System Action:	The job is terminated.
User Response:	Specify bits 0-7 as the mask for the bit operation and resubmit the job.

RG558–INVALID USE OF (OR MISSING) RESULTING INDICATORS WITH THIS OP CODE. ASSUME INVALID RESULTING INDICATORS BLANK.

Code:	W–Warning
Specification Type:	C
Explanation:	For CHAIN: Columns 56-59 must be blank. It is suggested that columns 54-55 contain an indicator that can be tested for record not found. For READ: Columns 54-57 must be blank. It is suggested that columns 58-59 contain an indicator that can be tested for end of file.
System Action:	Blank is assumed for columns that must be blank.
User Response:	To avoid this message the next time the job is run, make the necessary corrections, as mentioned above.

RG559–FACTOR 2 OR RESULT FIELD INVALID FOR SPECIFIED OPERATION

Code: T–Terminal

Specification Type: C

Explanation:

MOVEA operation:

(1) Either Factor 2 or the Result Field must contain the name of an array. (2) Both Factor 2 and the Result Field may contain the name of an array but not the same array.

XFOOT operation:

Factor 2 must be a whole array.

System Action:	The job is terminated.
User Response:	Make the proper entries in columns 33-42 and/or columns 43-48. Resubmit the job.

RG560–MODE OF PROCESSING GIVEN (COL 28) NOT ALLOWED, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F
Explanation:	The mode of processing entry specified in column 28 is invalid.
System Action:	The job is terminated.
User Response:	An entry of 'L' is allowed for limits or 'R' for random processing of disk files. Place the proper entry in column 28 and resubmit the job.

RG561–KEY FIELD START LOCATION (COLS 35-38) GIVEN BUT NOT ALLOWED, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F
Explanation:	The key field start location entry specified in columns 35-38 is invalid.
System Action:	The job is terminated.
User Response:	Place the proper entry in columns 35-38 of file description specifications for indexed files only. Resubmit the job.

RG562–FILE TYPE FOR FROM FILENAME AND/OR TO FILENAME INVALID WITH TABLE/ARRAY

Code:	T–Terminal
Specification Type:	Not applicable
Explanation:	The From Filename and/or the To Filename specified is invalid.
System Action:	The job is terminated.
User Response:	Make sure the From Filename specified in columns 11-18 of extension specifications is an input file and that the To Filename in columns 19-26 is an output file. Resubmit the job.

RG564–RECORD LENGTH IS NOT AT LEAST TWICE THE KEY LENGTH

Code:	T–Terminal
Specification Type:	F
Explanation:	The record length must be at least twice the key length.
System Action:	The job is terminated.
User Response:	Specify the record length to be at least twice the key length, and resubmit the job.

RG565–COLUMN 31 INVALID FOR DEVICE TYPE

Code:	T–Terminal
Specification Type:	F
Explanation:	The entry in column 31 is valid for record address or index files.
System Action:	The job is terminated.
User Response:	Leave column 31 blank or change the file type entry. Resubmit the job.

RG566–INVALID USE OF DEVICE AS FROM FILENAME

Code:	T–Terminal
Specification Type:	E
Explanation:	The file named in columns 11-18 of extension specifications is not assigned to the disk, MFCU, or console.
System Action:	The job is terminated.
User Response:	Place the proper From Filename entry in columns 11-18 and resubmit the job.

RG567–TABLE RECORD SIZE GREATER THAN FROM FILENAME DEVICE RECORD SIZE

Code:	T–Terminal
Specification Type:	E
Explanation:	Table or array record length specified exceeds the maximum record allowed for the device.
System Action:	The job is terminated.
User Response:	Make the table or array record length equal to or less than the maximum record length for the device. Resubmit the job.

RG568–LENGTH OF KEY FIELD OR RA LENGTH COLS 29-30 GIVEN BUT NOT ALLOWED, ASSUME BLANK

Code:	T–Terminal
Specification Type:	F
Explanation:	Length of key field or RA length specified in columns 29-30 is invalid for this file type.
System Action:	The job is terminated.
User Response:	Leave columns 29-30 blank, and resubmit the job.

RG569–ENTRY OF I COL 32 NOT GIVEN FOR AN INDEXED FILE, ASSUME I

Code:	T–Terminal
Specification Type:	F
Explanation:	The entry specified in column 32 for an indexed file is not I.
System Action:	I is assumed, the job is terminated.
User Response:	Enter I in column 32 for an indexed file and resubmit the job.

RG570–LOOK AHEAD WITH NUMERIC SEQUENCE OR LOOK AHEAD FOLLOWS A NUMERIC RECORD

Code:	T–Terminal
Specification Type:	I
Explanation:	A look ahead record type (** in columns 19-20) cannot be specified on the same line as a numeric sequence entry in columns 15-16.
System Action:	The job is terminated.
User Response:	Specify look ahead record types (** in columns 19-20) on the same line with an alphabetic entry in columns 15-16. Resubmit the job.

RG571–MORE THAN ONE LOOK AHEAD RECORD IN A FILE

Code:	T–Terminal
Specification Type:	I
Explanation:	Look ahead is specified more than once for this file.
System Action:	The job is terminated.
User Response:	Make only one look ahead specification for a file. Resubmit the job.

RG572–LOOK AHEAD CANNOT BE THE ONLY RECORD IN A FILE

Code:	T–Terminal
Specification Type:	I
Explanation:	Look ahead records specified do not follow other file or record type specifications.
System Action:	The job is terminated.
User Response:	Specify look ahead records following other file or record type specifications. Resubmit the job.

RG573–MULTI RA FILES DEFINED

Code:	T–Terminal
Specification Type:	F
Explanation:	More than one record address file is defined in this program.
System Action:	The job is terminated.
User Response:	Specify only one record address file per program. Resubmit the job.

RG574–EXTERNAL INDICATOR COLS 71-72 NOT THE SAME AS RA FILES

Code:	T–Terminal
Specification Type:	F
Explanation:	The record address file and the file it is used to process are not conditioned by the same external indicator.
System Action:	The job is terminated.
User Response:	When external indicators are used, specify the same external indicator for both the record address file and the file it is used to process. Resubmit the job.

RG575–NO INPUT SPECIFICATIONS FOUND FOR THIS FILE

Code:	T–Terminal
Specification Type:	Not applicable
Explanation:	Input specifications required for this file, but none were supplied.
System action:	The job is terminated.
User Response:	Supply input specifications for all input files (except record address and tables) and for update files. Resubmit the job.

RG576–COMPILE TIME TABLE DATA FOUND. COMPILE TIME TABLE OR ARRAY NOT SPECIFIED IN EXTENSION

Code:	W–Warning
Specification Type:	E
Explanation:	No extension specifications were supplied for compile time table.
System Action:	Table data is not processed.
User Response:	Supply the proper extension specifications and resubmit the job.

RG577–ONLY ONE FILE ASSOCIATED WITH AN RA FILE IS ALLOWED IN A PROGRAM

Code:	T–Terminal
Specification Type:	F, E
Explanation:	More than one record address file or more than one file associated with a record address file is defined in this program.
System Action:	The job is terminated.
User Response:	Specify only one record address file per program or associate only one file with a record address file.

RG578–A RECORD ADDRESS FILE OR A FILE ASSOCIATED WITH THE RECORD ADDRESS FILE IS REQUIRED BUT NOT DEFINED

Code:	T–Terminal
Specification Type:	C
Explanation:	A record address file or a file associated with a record address file is required for this job but was not defined.
System Action:	The job is terminated.
User Response:	Supply the proper record address file or file associated with the record address file and resubmit the job.

RG579–FIRST 1P LINE NOT FOR PRINTER, ASSUME COL 41 IN CONTROL CARD BLANK

Code:	W–Warning
Specification Type:	O
Explanation:	Forms alignment is requested but the first 1P line is not specified for a printer file.
System Action:	Column 41 of the control card specifications is assumed to be blank; therefore, no forms alignment is done.
User Response:	For forms alignment, specify the first 1P line for a printer file.

RG580–REFERENCED A MATCH LEVEL WHICH IS NOT VALID, OR DEFINED A LEVEL MORE THAN ONCE

Code:	T–Terminal
Specification Type:	I
Explanation:	Either an invalid match level is used or a match level is defined more than once.
System Action:	The job is terminated.
User Response:	Be sure that each record group contains the same match levels, and that each match level is defined only once. Resubmit the job.

RG581–MISSING OR INVALID AN/OR ENTRY IN COL 7-8

Code:	T–Terminal
Specification Type:	C
Explanation:	An AN/OR entry in columns 7-8 is missing or the entry specified is not AN or OR.
System Action:	The job is terminated.
User Response:	Make the proper AN/OR entry in column 7-8 and resubmit the job.

RG582–THE RELATIVE RECORD NUMBER FOR THE CHAIN OPERATION MUST BE NUMERIC WITH 0 DECIMAL

Code:	W–Warning
Specification Type:	C
Explanation:	The relative record number specified for a CHAIN operation is not a numeric field with zero decimal positions.
System Action:	The decimal positions are ignored.
User Response:	To avoid this message the next time this job is run, make the relative record number for a CHAIN operation a numeric field with zero decimal positions.

RG583–BINARY LENGTH SPECIFIED GREATER THAN 9, ASSUME 9

Code:	T–Terminal
Specification Type:	O
Explanation:	The binary length specified is greater than 9.
System Action:	The job is terminated.
User Response:	Make the binary length entry 9 or less and resubmit the job.

RG584–THIS MATCH LEVEL WAS REFERENCED PREVIOUSLY IN THIS RECORD GROUP

Code:	T–Terminal
Specification Type:	I
Explanation:	A match level was referenced more than once within one record group.
System Action:	The job is terminated.
User Response:	Be sure that each match level is referenced only once within a record group. Resubmit the job.

RG585–DISPLAY, CHAIN, OR DEMAND FILE SPECIFIED, BUT APPROPRIATE OPERATION CODE NOT FOUND IN CALCULATION SPECIFICATIONS

Code:	T–Terminal
Specification Type:	C
Explanation:	Display, chain, or demand files are specified but the appropriate operation codes are not specified in calculation specifications.
System Action:	The job is terminated.
User Response:	Specify the appropriate operation code and resubmit the job.

RG586–MORE THAN ALLOWABLE TABLE/ARRAY NAMES USED IN THE PROGRAM

Code:	T–Terminal
Specification Type:	E
Explanation:	More than 60 compile-time tables and/or arrays were defined or a total of more than 63 tables and/or arrays were defined in this program.
System Action:	The job is terminated.
User Response:	Reduce the number of compile-time tables and/or arrays to 60 or less and the total number tables and/or arrays to 63 or less.

RG587–IF FACTOR 1 OR FACTOR 2 IS A WHOLE ARRAY, RESULT FIELD MUST BE WHOLE ARRAY

Code:	T–Terminal
Specification Type:	C
Explanation:	The entry in Factor 1 or Factor 2 is a whole array, but the Result Field does not refer to a whole array.
System Action:	The job is terminated.
User Response:	When the entry in Factor 1 or Factor 2 is a whole array, place an array name in the Result Field. Resubmit the job.

RG588–TESTB, BITON, AND BITOF MAY NOT REFERENCE AN ENTIRE ARRAY

Code:	T–Terminal
Specification Type:	C
Explanation:	An entire array must not be referenced in a TESTB, BITON, or BITOF operation.
System Action:	The job is terminated.
User Response:	When using arrays with TESTB, BITON, or BITOF operations, specify array elements not the whole array. Resubmit the job.

RG589–RESULT FIELD MUST BE A ONE-POSITION ALPHAMERIC FIELD. IF FACTOR 2 IS A FIELD NAME, IT MUST BE A ONE-POSITION ALPHAMERIC FIELD

Code:	T–Terminal
Specification Type:	C
Explanation:	The Result Field is not a one-byte alphameric field for TESTB, BITON, and BITOF, or Factor 2 is a field name but is not a one-byte alphameric entry.
System Action:	The job is terminated.
User Response:	Make the Result Field a one-byte alphameric field for TESTB, BITON, or BITOF. If Factor 2 contains a field name, make it a one-byte alphameric field. Resubmit the job.

RG590–WHENEVER HIGH IS USED IN A MOVE ZONE OPERATION, IT MUST REFERENCE AN ALPHAMERIC FIELD

Code:	T–Terminal
Specification Type:	C
Explanation:	The high portion of a move zone instruction does not reference an alphameric field.
System Action:	The job is terminated.
User Response:	Make the high portion of a move zone instruction reference an alphameric field and resubmit the job.

RG591–LENGTH OF FIELD IN FACTOR 1 NOT EQUAL TO KEY LENGTH OF FILE SPECIFIED IN FACTOR 2

Code:	T–Terminal
Specification Type:	C
Explanation:	The length of the field in Factor 1 of a CHAIN operation is not equal to the key field length specified in Factor 2.
System Action:	The job is terminated.
User Response:	For a CHAIN operation, make the length of the chaining field (Factor 1) equal to the length of the key field (Factor 2). Resubmit the job.

RG592–FOR SEQUENTIALLY PROCESSED UPDATE FILE–T ENTRY IN COL 15 IS INVALID OR L0-L9 INDICATOR USED WITH E IN COL 15

Code:	T–Terminal
Specification Type:	O
Explanation:	Total output cannot be specified for update files processed sequentially.
System Action:	The job is terminated.
User Response:	Remove the T or E entry from column 15 and resubmit the job.

RG593–TABLE/ARRAY NAME MISSING FOR 'TO' AND/OR 'FROM' FILENAME

Code:	W–Warning
Specification Type:	E
Explanation:	No table name was specified in columns 27-32 for a table load operation (From Filename in columns 11-18) or for a table output operation (To Filename in columns 19-26).
System Action:	No action taken.
User Response:	To avoid the message when this job is run again, specify the proper table name in columns 27-32.

RG594–TO FILENAME MAY NOT BE USED WITH EXECUTION TIME TABLE/ARRAY

Code:	T–Terminal
Specification Type:	E
Explanation:	An array output operation (To Filename in columns 19-26) must not be specified for execution time arrays.
System Action:	The job is terminated.
User Response:	Remove the To Filename entry in columns 19-26 for execution time arrays. Resubmit the job.

RG595–COLS 27-32 AND 46-51 MUST BE BOTH TABLE OR BOTH ARRAY NAMES

Code:	T–Terminal
Specification Type:	E
Explanation:	For alternating tables, columns 27-32 and 46-51 do not both contain table names; or columns 27-32 and 46-51 do not both contain array names for alternating arrays.
System Action:	The job is terminated.
User Response:	For alternating tables or arrays, specify either table names or array names in both columns 27-32 and 46-51. Resubmit the job.

RG597–END POSITION SPECIFIED FOR *PLACE LESS THAN TWICE THAT OF HIGHEST PREVIOUSLY SPECIFIED FIELD END POSITION OR END POSITION GREATER THAN 256

Code:	T–Terminal
Specification Type:	O
Explanation:	The end position specified for *PLACE is lower than end position specified for the preceding field or greater than 256.
System Action:	The job is terminated.
User Response:	Make the proper end position entry for *PLACE, and resubmit the job.

RG598–ALPHA TABLE/ARRAY SPECIFIED AS PACKED, ASSUME NUMERIC

Code:	T–Terminal
Specification Type:	E
Explanation:	An alphameric table or array was specified as packed.
System Action:	The job is terminated.
User Response:	Specify the table or array as numeric, and resubmit the job.

RG599–LENGTH OF ELEMENT FOR BINARY TABLE/ARRAY NOT SPECIFIED AS 4 OR 9, DEFAULT TO 4 IF LENGTH SPECIFIED IS LESS THAN 4, OTHERWISE DEFAULT TO 9

Code:	T–Terminal
Specification Type:	E
Explanation:	The binary length was not specified as 4 or 9.
System Action:	The job is terminated.
User Response:	Make the proper binary length entry and resubmit the job.

RG621–TRAILER RECORD OVERLAPS HEADER RECORD

Code:	T–Terminal
Specification Type:	I
Explanation:	The trailer field overlaps the header field in a spread card.
System Action:	The job is terminated.
User Response:	Make the first trailer field start after the last position in the header field. Resubmit the job.

RG622–NO TRAILER FIELDS FOR SPREAD CARD

Code:	T–Terminal
Specification Type:	I
Explanation:	No trailer fields are specified for the spread card.
System Action:	The job is terminated.
User Response:	Make the proper trailer field entries for the spread card (TR in columns 19-20). Resubmit the job.

RG623–ENTRIES IN COLUMNS 7-18 AND 21-74 INVALID FOR TR SPECIFICATION, ASSUME NO TR

Code:	T–Terminal
Specification Type:	I
Explanation:	Entries specified in columns 7-18 and 21-74 of a TR line.
System Action:	Columns 19-20 are assumed blank; no spread cards are accepted. The job is terminated.
User Response:	If spread cards are to be used, leave columns 7-18 and 21-74 blank for the TR line (TR in columns 19-20). Resubmit the job.

RG624–TR SPECIFICATION OUT OF ORDER

Code:	T–Terminal
Specification Type:	I
Explanation:	The TR specification line is not preceded by a definition of a header record.
System Action:	The job is terminated.
User Response:	Place the TR specification line immediately after a definition of a header record. Resubmit the job.

RG625–FACTOR 1 MUST BE NUMERIC FOR CHAIN OPERATION WHEN FACTOR 2 FILENAME HAS PACKED KEYS

Code:	T–Terminal
Specification Type:	C
Explanation:	The entry specified in Factor 1 of a CHAIN operation is not numeric even though the file named in Factor 2 has packed keys.
System Action:	The job is terminated.
User Response:	Make the entry in Factor 1 of a CHAIN operation numeric when the file named in Factor 2 has packed keys. Resubmit the job.

RG626–MORE THAN 128 TR SPECIFICATIONS GIVEN

Code:	T–Terminal
Specification Type:	I
Explanation:	More than 128 valid TR lines are specified in this program.
System Action:	The job is terminated.
User Response:	Make the number of valid TR lines in this program 128 or less. Resubmit the job.

RG628–INVALID FILE TYPE FOR SPREAD CARD

Code:	T–Terminal
Specification Type:	I
Explanation:	The file containing spread cards is not a card input file designated as primary or secondary.
System Action:	The job is terminated.
User Response:	Make sure the file containing spread cards is a card input file designated as primary or secondary. Resubmit the job.

RG631–FACTOR 1 MUST HAVE SAME LENGTH WHEN PACKED AS LENGTH OF PACKED KEYS FOR FACTOR 2 FILENAME

Code:	T–Terminal
Specification Type:	C
Explanation:	The entry in Factor 1 of a CHAIN operation is not the same length when packed as the record keys in the file named in Factor 2.
System Action:	The job is terminated.
User Response:	Make sure the entry in Factor 1 of a CHAIN operation is the same length when packed as the record key in file named in Factor 2. Resubmit the job.

RG635–NUMERIC SEQUENCE CHECKING SPECIFIED FOR A SPREAD RECORD, BUT N NOT SPECIFIED FOR NUMBER, ASSUME N

Code:	W–Warning
Specification Type:	I
Explanation:	An N entry was not made in column 17 even though sequence checking was specified (numeric entry in columns 15-16).
System Action:	N is assumed.
User Response:	To avoid this message when this job is run again, enter N in column 17.

RG644–SHARED I/O NOT ALLOWED WITH DISK40 OR DISK45 IN PROGRAM, ASSUME NO SHARED I/O

Code:	T–Terminal
Specification Type:	F
Explanation:	Shared I/O cannot be used with DISK40 or DISK45.
System Action:	Job is terminated.
User Response:	To avoid this message on the next run, remove the shared I/O entry (column 48) from the header card.

RG645–IMPROPER USE OF THE PACK/UNPACK FEATURE FOR LIMITS FILE PROCESSING

Code:	T–Terminal
Specification Type:	E
Explanation:	The unpacked key length must be either 1 or 2 less than twice the packed key length.
System Action:	The job is terminated.
User Response:	Correct the unpacked key length and resubmit the job.

RG646–WHOLE ARRAY IN RESULT FIELD INVALID FOR SPECIFIED OP CODE

Code:	T–Terminal
Specification Type:	C
Explanation:	An entire array cannot be specified as the result field for the operation specified.
System Action:	The job is terminated.
User Response:	Make the result field a field (must be numeric for XFOOT), an array element or a table element. Resubmit the job.

RG647–UNEQUAL KEY LENGTHS SPECIFIED FOR KEYS OF IDENTICAL FORMAT

Code:	W–Warning
Specification Type:	E
Explanation:	The key length of a limits file (record address file) should be equal to the key length of the file to be processed by limits.
System Action:	The key length of the file processed by limits is assumed as the key length of the limits file.
User Response:	Specify the key lengths to be equal, to eliminate the message.

RG701–INVALID NUMBER OF SECTORS SPECIFIED

Code:	W–Warning
Specification Type:	F
Explanation:	The number of sectors specified on the file description specifications continuation statement for INDEX is invalid.
System Action:	A default of 1 is used.
User Response:	Specify correct number wanted.

RG702–KEY WORD REPEATED FOR A FILE, SECOND IGNORED

Code:	W–Warning
Specification Type:	F
Explanation:	Continuation entry INDEX may appear only once for any file.
System Action:	The second entry is ignored.
User Response:	Remove second entry.

RG703–CONTINUATION KEYWORD SPECIFIED IS INVALID

Code:	T–Terminal
Specification Type:	F
Explanation:	The only entry in columns 54-59 is INDEX.
System Action:	The continuation specification is ignored, and the job is terminated.
User Response:	Make correct entry in columns 54-59 or remove the continuation specification.

RG704–BOTH MFCU AND MFCM DEVICES ARE SPECIFIED IN THE SAME PROGRAM

Code:	T–Terminal
Specification Type:	F
Explanation:	The MFCU and MFCM may not be specified in the same program.
System Action:	The job is terminated and the entire specification is ignored. This condition may cause other errors to be generated.
User Response:	Make necessary corrections and resubmit the job.

RG705–INPUT OR OUTPUT CRT77 FILE SPECIFIED IN THE SAME PROGRAM WITH CRT77 UPDATE FILE

Code:	W–Warning
Specification Type:	F
Explanation:	If a CRT77 update file is specified there should not be any input or output CRT77 files specified.
System Action:	None.
User Response:	Correct the program and recompile.

RG706–MORE THAN ONE DISPLAY FILE SPECIFIED IN PROGRAM

Code:	T–Terminal
Specification Type:	F
Explanation:	Only one display file is allowed per program.
System Action:	The job is terminated.
User Response:	Correct the program and resubmit the job.

RG707–RESULT FIELD FOR TIME OPERATION CODE MUST BE 6 OR 12 DIGIT NUMERIC WITH NO DECIMAL POSITIONS

Code:	T–Terminal
Specification Type:	C
Explanation:	The result field specified for the TIME operation code has been defined with a length other than 6 or 12 or is not numeric with zero decimals.
System Action:	The job is terminated.
User Response:	Correct the error and resubmit the job.

RG708–CONTINUATION KEYWORD 'INDEX' IS NOT ALLOWED FOR THE TYPE OF FILE PROCESSING SPECIFIED

Code:	W–Warning
Specification Type:	C
Explanation:	'INDEX' keyword is not allowed for: – Index random input (with NO ADD specified) – Index random update (with NO ADD specified) – Index output (with NO ADD specified)
System Action:	The continuation entry is ignored.
User Response:	Make sure 'INDEX' continuation is specified with an allowed type of file processing.

RG799–ERROR FILE FULL

Code:	T–Terminal
Specification Type:	Not applicable
Explanation:	Too many errors were made in this program.
System Action:	Job is terminated.
User Response:	Correct the errors diagnosed in this program; resubmit the job.

RG999–PROGRAM EXCEEDS CORE IN COL 12-14 OF HEADER CARD

Code:	W–Warning
Specification Type:	H
Explanation:	The program requires more core storage for execution than specified in columns 12-14 of the control card specifications.
System Action:	No action taken.
User Response:	To avoid this message when this job is run again, make the proper entry in columns 12-14.

Assembler subroutines may be linked to an RPG II program by the following methods:

- An EXIT operation is used to pass control to a subroutine to be used during calculations. The subroutine is named in the EXIT operation, and a field, table or array, or indicator may be passed to the subroutine by an RLABL operation following the EXIT operation.
- For the IBM System/3 Card System and Disk System, a SPECIAL device may be named in the RPG II file description specifications. The file description specifications are used to name a subroutine to control input/output functions for the special device.

EXIT and RLABL Operations

Linkage from RPG II to an assembler language subroutine is accomplished through the EXIT and RLABL RPG II operations. Control *cannot* be transferred from one assembler subroutine to another. All EXIT and SUBR type SPECIAL subroutines will be a part of the root segment and will not be put into overlays.

EXIT Operation

The EXIT operation code is used to designate a point in the RPG II calculation specifications at which control is to be passed to a previously assembled, external subroutine.

The rules for use of the EXIT operation in RPG II calculation specifications are as follows:

Columns	*Entry*
Operation (28-32)	EXIT
Factor 1 (18-27)	Blank
Factor 2 (33-42)	The name of the subroutine to which control is to be passed. The name must consist of five or six characters, the first four of which are SUBR. The remaining characters must be alphabetic for user written subroutines. (Numeric characters are reserved for IBM supplied subroutines.) The module name and entry point name must be the same.
Result Field (43-48)	Blank
Resulting Indicators (54-59)	Blank

The EXIT operation can be conditioned by control level entries (columns 7-8) and indicators entries (columns 9-17). If not conditioned by control level entries, the EXIT operation occurs at detail calculation time.

The position of the EXIT operation in the calculation specifications of the RPG program determines at what point the actual subroutine execution will occur. See Figure F-1.

Position	Execution of Subroutine
First Detail line in calculation specifications	Immediately following data routine (that is, after data is extracted from input record.)
Last Detail line in calculation specifications	Immediately before heading records output time.
First Total line in calculation specifications	Immediately following input routine (after determination of record type and testing for control level break).
Last Total line in calculation specifications	Immediately before total records output time.
Any other Detail/Total line in calculation specifications	Immediately following the previous calculation operation.

Figure F-1. Relationship Between Position of EXIT Operation and Execution of Subroutine

RLABL Specification

Through the RLABL operation, a field, table or array, or indicator defined in the RPG II program can be referenced by the subroutine to which the EXIT operation gives control. The rules for use of RLABL in RPG II calculation specifications are as follows:

Columns	*Entry*
Operation (28-32)	RLABL
Result Field (43-48)	Field, table or array name, or indicator
Field Length (49-51)	Length of field (optional)
Decimal Positions (52)	Decimal indication (optional)

The RLABL specifications must immediately follow the EXIT specifications for the subroutine which references the RPG II field. A name defined by a TAG, BEGSR, or ENDSR specification cannot be used in an RLABL specification.

An assembler subroutine may reference indicators in the RPG II program to which it is linked. This is done by entering INxx in the result field of an RLABL specification. The xx represents the indicator to be referenced. For example, if MR is to be tested, INMR must be entered in the result field of the RLABL specification.

Using RLABL Fields in the EXIT Routine

When linkage is effected from RPG II to an assembler subroutine, there are three possible entries in the Result Field of the RLABL specification: field table or array, and indicator. See Figure F2. Figures also show the RPG II coding for the linkages, and the compiled parameters representing the RLABL fields. See *Sample Programs* later in this section for further examples.

Table

RPG CALCULATION SPECIFICATIONS (IBM International Business Machine Corporation; Form GX21-9093, Printed in U.S.A.)

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators	Comments
01	C				EXIT	SUBRXX						
02	C				RLABL		TABLE	2				
03	C											
04	C											

(Table name or Array name)

Field

RPG CALCULATION SPECIFICATIONS (IBM International Business Machine Corporation; Form GX21-9093, Printed in U.S.A.)

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators	Comments
01	C				EXIT	SUBRXX						
02	C				RLABL		FIELD	1				
03	C											
04	C											

(Field name)

Indicator

RPG CALCULATION SPECIFICATIONS (IBM International Business Machine Corporation; Form GX21-9093, Printed in U.S.A.)

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators	Comments
01	C				EXIT	SUBRXX						
02	C				RLABL		INYY					
03	C											
04	C											

(Indicator)

Figure F-2. RPG II Coding for RLABL Field Entries

SAMPLE PROGRAMS

RPG Linkage Sample Program 1

In this sample program, the RPG II coding uses the EXIT operation to effect linkage to the assembler language subroutine SUBRxx (Figures F-3 and F-4). The RLABL specification names a field called HERE, into which SUBRxx moves a character A. When control is returned to the RPG II program, a compare operation is performed to determine which character was placed in the field HERE.

RPG Linkage Sample Program 2

In this sample program, the RPG II coding uses the EXIT specification to effect linkage to the assembler subroutine SUBRB (Figure F-4). The first RLABL specification names a table, TABB, and the second names an indicator, IN44; The subroutine refers to both RLABL entries. It first tests the indicator. If the indicator is off, control is returned to the RPG II program. If the indicator is on, a character C is moved into the last looked up entry in the table TABB. When control is returned to the RPG II program, a compare operation is performed to see whether or not the subroutine placed a C in TABB.

Special

A System/3 Model 10 Disk System RPG II, Model 12 RPG II, or Model 15 RPG II user can link to a subroutine to control input/output for a SPECIAL device. This is done by providing a link to a user-written routine that performs data transfer for the special device. Control cannot be transferred from one user assembler subroutine to another.

The following specifications are for the RPG II file description specifications in which the SPECIAL device file is defined and the I/O subroutine is named.

File description entries for SPECIAL device:

Columns	*Entries*
6	F
7-14	Valid RPG II file name
15	I, O, C, or U
16	Blank, P, S, D
17	Blank, E
18	Blank, A, D
19	Blank, F
20-23	Blank, block length
24-27	Record length
28-31	Must be blank
32	Blank, 1-9 (dual I/O areas are allowed).
33-39	Must be blank
40-46	The word SPECIAL
47-52	Must be blank
53	Continuation line
54-59	Name of the user's subroutine which will perform I/O function. The subroutine name will be SUBRxx, where x is any valid alphabetic character.
60-70	Must be blank
71-72	Blank, U1-U8
73-74	Must be blank

The following can be used with SPECIAL files:

- FORCE operation code.
- READ operation code.
- File translation.
- *PLACE on output.

The following cannot be used with SPECIAL files:

- CHAIN operation code.
- Stacker select.
- Spacing and skipping.
- *PRINT.
- *(asterisk) in column 40 on Output Sheet to print constants on cards.

On Models 10 and 12 systems, care must be taken when using the Dual Programming Feature with a special device. If SPECIAL for the same device is used in both levels, it is the programmer's responsibility to see that the device is ready.

SPECIAL files can only be processed consecutively. See Figure F-5 for possible file description entries for SPECIAL files.

The *IBM System/3 Basic Assembler Reference Manual,* SC21-7509, describes the operation codes passed to data management and the completion codes passed back by data management.

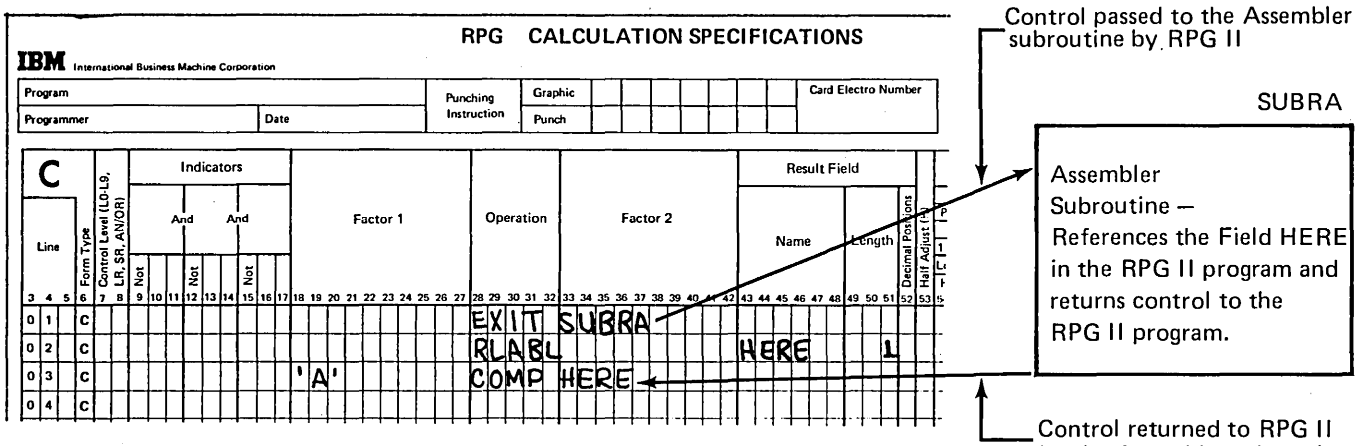


Figure F-3. RPG II Coding for Sample Program 1

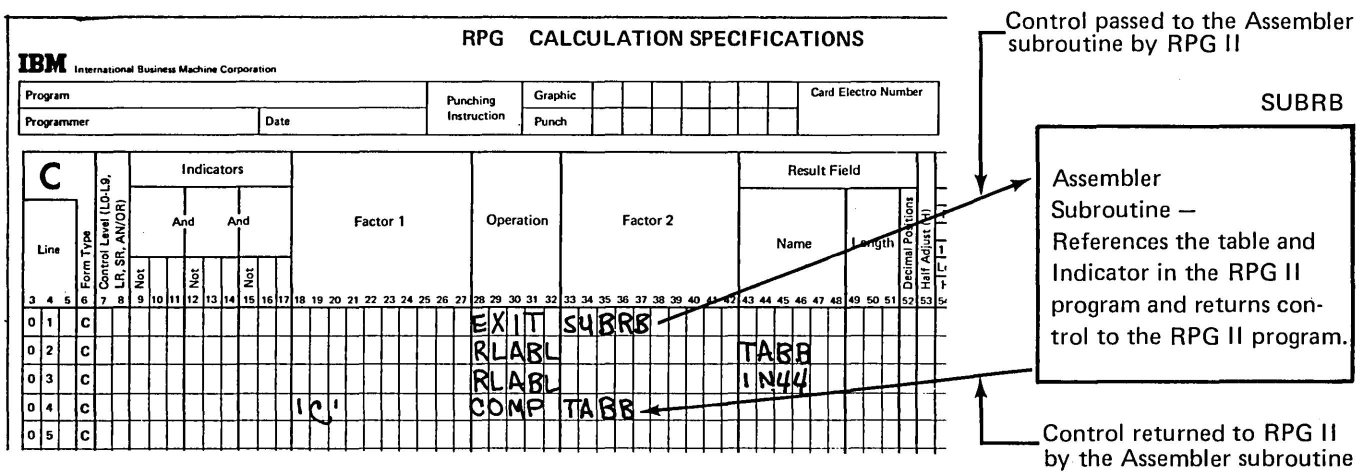


Figure F-4. RPG II Coding for Sample Program 2

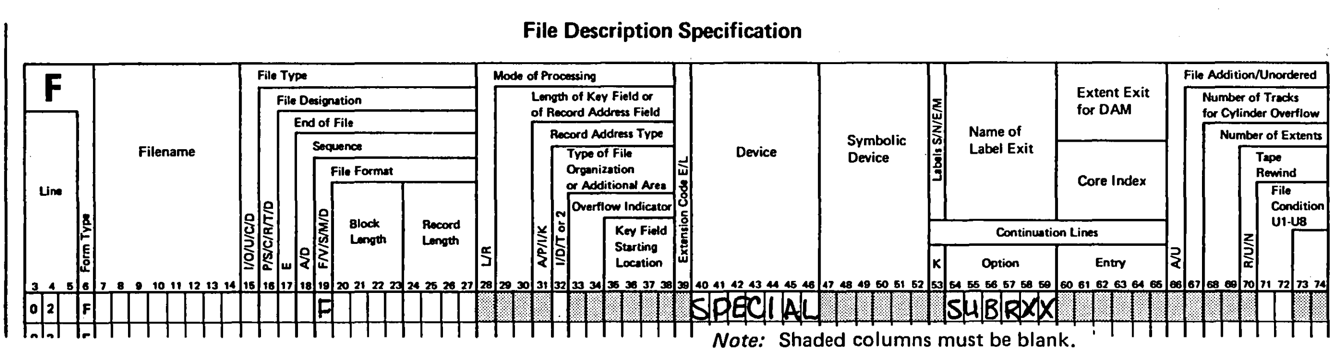


Figure F-5. File Description Entries for Special Device Support

Appendix G. Summary of RPG II Specifications

This appendix contains a brief column-by-column description of each of the RPG II specification sheets. It is intended as a quick reference by programmers who are acquainted with RPG II for the IBM System/3 Disk System. For a complete description of each entry, refer to the applicable section of this manual. For a complete description of telecommunications entries see *IBM System/3 RPG II Telecommunication Programming Reference Manual,* SC21-7507.

INFORMATION COMMON TO ALL FORMS

RPG II source cards should be in ascending numeric sequence by columns 1 through 5. Cards that are out of sequence are flagged. Adjacent cards with duplicate sequence numbers are not flagged.

Columns 1-2 (Page)

Arrange the specifications sheets in the following order and number them in ascending sequence:

1. Control Card and File Description.
2. Extension and Line Counter.
3. Telecommunications.
4. Input.
5. Calculation.
6. Output.

Columns 3-5 (Line)

The first two digits of the line number are pre-printed. Use the unnumbered lines on the sheet for additional specifications or, along with column 5, to insert a line between two other completed lines. For example, line 025 would be inserted between lines 02 and 03.

Column 6 (Form Type)

This column contains a pre-printed code (H, F, E, L, T, I, C, or O) which must be punched into all RPG II specifications cards.

Column 7 (Comments)

Enter an asterisk in each line used as a comment line. The control card specification line (line 01) cannot be used as a comment line.

Columns 75-80 (Program Identification)

Insert any valid characters in columns 75-80 of the control card to identify the program. This name is used in a program directory which contains the location of your program on disk. If these columns are left blank, RPGOBJ is assumed. Columns 75-80 on all other specifications cards can contain any entries.

CONTROL CARD SPECIFICATIONS

Columns 7-9 (Core Size to Compile)

Leave these positions blank.

Column 10 (Object Output)

Entry		*Object Program is:*
Blank	–	Written temporarily in object library.
D	–	Written temporarily in object library.
C	–	Written permanently in object library.
P	–	Punched into cards.
R	–	Place non-link-edited object program in the library as a permanent entry (Model 15 only).
T	–	Place non-link-edited object program in the library as a temporary entry (Model 15 only).
B	–	Punch non-link-edited object programs (Model 15 only).

Column 11 (Listing Options)

Blank	–	Program listing produced
B	–	No program listing produced
P	–	Partial program listing produced

Columns 12-14 (Core Size to Execute)

Column 12

Blank, 0	–	No additional 256-byte increments are needed.
Q	–	One additional 256-byte increment is needed.
H	–	Two additional 256-byte increments are needed.
T	–	Three additional 256-byte increments are needed.

Columns 13-14

Blank	–	Core size available for execution is same as core size used for compilation.
01-61	–	Core size available for execution if different from core size used for compilation. Entry is the number of K (1K=1,024 bytes) available.

Column 15 (Debug)

Blank	–	DEBUG operation not used.
1	–	DEBUG operation used.

Column 16-20

Leave these positions blank.

Column 21 (Inverted Print)

Blank	–	United States format.
I	–	World Trade format.
J	–	World Trade format (leading zero remains for zero balances).
D	–	United Kingdom format.

Columns 22-25

Leave these positions blank.

Column 26 (Alternate Collating Sequence)

Blank	–	Normal collating sequence used.
S	–	Alternate collating sequence used.

Columns 27-36

Leave these positions blank.

Column 37 (Inquiry)

Blank	–	Program not interruptable.
B	–	Program recognizes inquiry requests.
I	–	Inquiry program.

Columns 38-40

Leave these positions blank.

Column 41 (1P Forms Position)

Blank – First 1P line printed only once.

1 – First 1P line can be printed repeatedly to allow forms positioning.

Column 42

Leave this position blank.

Column 43 (File Translation)

Blank – No file translation needed.

F – Input, output, update, or combined files are to be translated.

Column 44 (Punch MFCU Zeros)

Blank – Leading zeros are removed.

1 – Leading zeros are used (applies to MFCU only).

Column 45 (Nonprint Characters)

Blank Program halts if an unprintable character is encountered.

1 – No halt for unprintable character.

Columns 46-47

Leave these positions blank.

Column 48 (Shared I/O on Model 10; External Buffers on Program Number 5704-RG2)

Blank– All 5444 disk files use a separate input/output area.

1 – All 5444 disk files share a single input/output area (Model 10 only).

E – External disk I/O buffers (Program Number 5704-RG2 only).

Columns 49-74

Leave these positions blank.

FILE DESCRIPTION SPECIFICATIONS

Columns 7-14 (Filename)

Enter a name for each file. The filename can be from one to eight characters long, must begin in column 7, and must be a valid RPG II name (see *Definition of Terms* in Chapter 1).

Column 15 (File Type)

I – Input

O – Output

U – Update

C – Combined

D – Display

Column 16 (File Designation)

P – Primary

S – Secondary

C – Chained

R – Record Address

T – Table or Array

D – Demand

Leave blank for display files and all output files except chained output files.

Column 17 (End of File)

E – All records from the file must be processed before the program can end.

Blank – The program can end whether or not all records from this file have been processed.

If column 17 is blank or *E* for all files, all records from every file must be processed before the program can end. An *E* can only be specified here if column 15 contains *I, U,* or *C* and column 16 contains a *P, S,* or *R.*

Column 18 (Sequence)

Blank – No sequence checking is to be done.

A – Sequence checking is done. Records are in ascending sequence.

D – Sequence checking is done. Records are in descending sequence.

Sequence checking is required when matching fields are used. Column 18 applies to update and combined files and all input files except table, array, chained, demand, and record address files.

Column 19 (File Format)

F – Must be entered for fixed length records.

V – Must be entered for tape files with variable length records.

Columns 20-23 (Block Length)

Disk: 1-9999 (multiple of record length)

MFCM: 1-80

MFCU: 1-96

DISKET: 1-128

1442: 1-80

2501: 1-80

Printer-keyboard: 1-125

CRT/keyboard: 1-120, 1-39, 1-279 (depending on file type).

Printer: 1-96, 1-120, or 1-132 (depending on number of print positions).

Tape: 18-9999 (multiple of record length plus the size of the buffer offset). For variable length tape records see *Columns 20-23 (Block Length)* under *File Description Specifications.*

Device independent input and output files: 1-9999 (multiple of record length).

Block length entry for files other than disk or tape must be equal to record length.

Columns 24-27 (Record Length)

Disk: 1-9999

MFCM: 1-80

MFCU: 1-96

1442: 1-80

2501: 1-80

Printer-keyboard: 1-125

CRT/keyboard: 1-120, 1-39, 1-279 (depending on file type).

Printer: 1-96, 1-120, or 1-132 (depending on number of print positions).

Tape: 18-9999 (equal or less than block length).

Device independent input and output files: 1-9999

DISKET: 1-128

Column 28 (Mode of Processing)

Blank –
1. Sequential by key.
2. Consecutive

L – Sequential within limits.

R –
1. Random by relative record number.
2. Random by key.
3. By ADDROUT file.
4. Direct file load (random load).

This column must be blank for non-disk files.

Column 29-30 (Length of Key Field or Record Address Field)

Indexed file: Length of record key.

Record address file containing limits: Length of record key.

ADDROUT file: Length of record (always 3).

Maximum length of a record key is 29 characters.

Column 31 (Record Address Type)

A – Indexed file.

P – Indexed file with packed keys.

I – ADDROUT file or processed by ADDROUT file.

Blank – Sequential or direct file.

Column 31 applies to disk files specified as input, update, or chained output files.

Column 32 (File Organization or Additional I/O Area)

I – Indexed organization.

T – ADDROUT file.

1-9 – Sequential or direct file, use two I/O areas for the file.

Blank – Sequential or direct file, use one I/O area for the file.

Columns 33-34 (Overflow Indicator)

OA-OG, OV – Overflow indicator used to condition records in the file.

Blank – No overflow indicator is used.

Columns 35-38 (Key Field Starting Location)

1-9999 Record position in which the key field begins.

Column 39 (Extension Code)

E – The file described on this line is a table file, array file, or record address file further described on extension specifications.

L – The file described on this line is a printer file futher described on line counter specifications.

Columns 40-46 (Device)

Enter the device code for the input/output unit used by the file specified in columns 7-14, as follows:

Input/Output Unit	*Device Code*
IBM 5424 Multi-Function Card Unit	MFCU1 (Primary Hopper) MFCU2 (Secondary Hopper)
IBM 2560 Multi-Function Card Machine	MFCM1 (Primary Hopper) MFCM2 (Secondary Hopper)
IBM 1442 Card Read Punch	READ42
IBM 2501 Card Reader	READ01
IBM 1403 Printer	PRINTER
IBM 3284 Printer	PRINT84
IBM 5203 Printer (dual carriage)	PRINTER (Left Carriage) PRINTR2 (Right Carriage)
IBM 5471 Printer-Keyboard	CONSOLE
IBM 3277 Display Station	CRT77
IBM 5444 Disk Storage	DISK
IBM 5445 Disk Storage Drive	DISK45
IBM 3410 or 3411 Magnetic Tape Unit	TAPE
Binary Synchronous Communications Adapter	BSCA
IBM 3741 Data Station or Programmable Work Station	DISKET
IBM 3340 Direct Access Storage Facility	DISK40 (Models 12 and 15 only)
IBM 3344 Direct Access Storage Facility	DISK40 (Program Number 5704-RG2 only)
Device independent input and output files	No entry in Device Code

Columns 47-52

Leave these positions blank.

Column 53

Labels

Leave this position blank unless using continuation lines.

Continuation Lines

K – Continuation record specified for tape, SPECIAL, DISK (Model 15), DISK45 (Model 15), or DISK40 (Model 15).

Columns 54-59

Name of Label Exit

Entry	*Explanation*
Blank	No SPECIAL device used.
SUBRxx	Name of the user-written subroutine which will perform the I/O operation for a SPECIAL device
SRyzzz	Name of the IBM-written subroutine which will perform the I/O operation for a SPECIAL device.

Continuation Line Option

Entry	*Explanation*
ASCII	ASCII tape file specified.
BUFOFF	Tape input file contains a block prefix (used only if ASCII file is specified).
Array	Name of array to be used by user.
Name	Written IOS subroutine.
INDEX	User specifies amount of index buffer RPG should allocate.

Columns 60-65

Core Index

6-9999 – Number of bytes reserved for core index.

Blank – No core index will be built.

Continuation Line Entry

0-99 – Length of the block prefix in an ASCII tape input file that specifies BUFOFF.

1-9 – Amount of storage, in 256 byte increments, to be provided for the index buffer.

Column 66 (File Addition/Unordered)

A – New records will be added to the file.

U – Records are to be loaded into an indexed file in unordered sequence.

This column applies to sequential and indexed disk files.

Column 67

Leave this position blank.

Columns 68-69 (Number of Extents)

Blank	–	Single volume file.
1-50	–	Number of volumes that contain the file.

For consecutive processing, if any volumes are offline during processing, then all volumes must be on removable packs. For sequential or random processing, all volumes must be on line.

Column 70 (Tape Rewind)

R	–	Rewind tape at end of file
U	–	Unload tape at end of file
N	–	Leave tape at end of file

Columns 71-72 (File Condition U1-U8)

U1-U8	–	File is conditioned by the specified external indicator.
Blank	–	File is not conditioned by an external indicator.

These columns apply to output files and primary and secondary input (except table or array input files), update, and combined files. A record address file may be conditioned by an external indicator if its associated primary or secondary file is conditioned either by the same indicator or by no indicator.

Columns 73-74

Leave these positions blank.

EXTENSION SPECIFICATIONS

Columns 7-10

Leave these positions blank.

Columns 11-18 (From Filename)

Enter, left justified, the name of the table or array input file loaded at pre-execution time or the name of the record address file defined on the File Description Sheet.

Columns 19-26 (To Filename)

If the file named in From Filename is a record address file, enter the name of the primary or secondary input or update file containing the data records to be processed. If From Filename is a table or array file, enter the name of the output file to which the table or array is written at end of job. Leave this entry blank if the table or array is not written out.

Columns 27-32 (Table or Array Name)

Enter the name of a table or array used in the program. If alternating tables or arrays are described, enter the name of the table or array whose entry is first on the input record. Entries are left-justified and must be valid RPG II names (see *Definition of Terms* in Chapter 1). Table names must begin with TAB; array names must not begin with TAB.

Columns 33-35 (Number of Entries Per Record)

Enter, right-justified, the number of entries on each table or array input record. These columns must contain an entry for compile and pre-execution time tables and arrays. These columns must be blank for execution time arrays.

Columns 36-39 (Number of Entries Per Table or Array)

Enter, right-justified, the maximum number of entries in the table or array named in columns 27-32. For alternating tables or arrays, corresponding items are considered one entry.

Columns 40-42 (Length of Entry)

Enter, right-justified, the length of each table or array entry. The maximum length is 256 for alphameric entries and 15 for numeric entries. For packed or binary tables and arrays, enter the number of bytes of storage required to represent the data in unpacked format.

Column 43 (Packed or Binary Field)

Blank	–	Alphameric or unpacked numeric data.
P	–	Packed numeric data.
B	–	Binary numeric data.

Column 44 (Decimal Positions)

Blank	–	Alphameric table or array.
0-9	–	Number of positions to the right of the decimal.

Column 45 (Sequence)

Blank	–	No particular sequence.
A	–	Ascending sequence.
D	–	Descending sequence.

This column describes the sequence of data in a table or array. Column 45 must contain an entry if high or low look-up is to be used.

Columns 46-57

Use these columns when describing a second table or array entered in alternating format with the table or array named in columns 27-32. These entries have the same significance as the corresponding entries in columns 27-45.

Columns 58-74 (Comments)

Enter any information you wish to help you understand or remember what you are doing in each specification line.

LINE COUNTER SPECIFICATIONS

Columns 7-14 (Filename)

Enter the name of a printer file for which you wish to specify a form size and overflow line.

Columns 15-17 (Line Number - Number of Lines Per Page)

12-112	–	Number of lines available for printing on the printer form.

Columns 18-19 (Form Length)

Enter *FL* to indicate the previous entry is the form length.

Columns 20-22 (Line Number - Overflow Line)

1-112	–	Number of the overflow line.

Columns 23-24 (Overflow Line)

Enter *OL* to indicate the previous entry is the overflow line.

Columns 25-74

Leave these positions blank.

TELECOMMUNICATIONS SPECIFICATIONS

Columns 7-14 (Filename)

Enter a valid filename for every BSC file your program uses.

Column 15 (Configuration)

P or blank – Point-to-point, non-switched network.

M – Multipoint network, where the control station selects the tributary station through polling or addressing. System/3 cannot be the control station.

S – Switched network.

Column 16 (Type of Station)

T – This station will transmit messages from this file (transmit only or transmit with reception of conversational reply). The file must be designated as an output or combined file on the File Description Sheet and must appear on the Output Sheet.

R – This station will receive messages into this file (receive only or receive with transmittal of conversational reply). The file must be designated as an input or combined file on the File Description Sheet and must appear on the Input Sheet.

Column 17 (Type of Control)

T – This is a tributary station on a multipoint network. System/3 cannot be the control station and transmit the polling supervisory sequence.

Blank – Polling is not used; non-tributary station.

Column 17 must contain a T if column 15 contains an M (multipoint network).

Column 18 (Type of Code)

A, U – ASCII data link control characters will be used. When ASCII is used, each station must provide file translation when it is required.

E or blank – EBCDIC data link control characters will be used.

Column 19 (Transparency)

Y – This entry is valid only for EBCDIC. The transparency feature must be installed. The data being transferred may contain data link control characters.

N or blank – The transparency feature is not used. Unpacked numeric or alphameric data will be transmitted and received. The data being transferred may not contain data link control characters.

Column 20 (Switched)

M – The computer operator makes the connection between stations by dialing the number (manual dial).

E – Autocall is to be used. The dial number is listed in columns 21-31.

S – Autocall is to be used. The entry in columns 21-31 is the symbolic location of the dial number.

A – Autoanswer is used by the called station.

B – Manual answer is used by the called station.

Blank – This is not a switched network.

Columns 21-31 (Dial Number)

Numeric – This is the number to be dialed when column 20 contains an E.

Alphameric – Columns 21-31 must contain a symbolic name, other than an array name, referencing the location of the dial number when column 20 contains an S. If the BSC file is an input file other than a demand or conversational receive file, this name must refer to the first (or only) element of a table.

Column 32 (Location of Identification—This Station)

S – Switched network. This station's identification is located at the position referenced by the symbolic name specified in columns 33-39.

E – Switched network. The entry in columns 33-39 is this station's identification.

Blank – This is a non-switched network or a switched network where no ID is desired for this station.

Columns 33-39 (Identification—This Station)

Alphameric – When column 32 contains an E, this entry is the actual identification sequence of this station (from 2 to 15 characters). The station identification must not contain a control character sequence.
When column 32 contains an S, this entry is the symbolic name of the location of this station's identification. The symbolic name must not be an array name. If the BSC file is primary or secondary, this symbolic name must refer to the first element of a table.

Column 40 (Location of Identification—Remote Station)

S – Switched network. The remote station's identification is located at the position referenced by the symbolic name specified in columns 41-47.

E – Switched network. The entry in columns 41-47 is the remote station's identification.

Blank – This is a non-switched network or a switched network where no ID is desired for the remote station.

Columns 41-47 (Identification—Remote Station)

Alphameric – When column 40 contains an E, this entry is the actual identification sequence of the remote station (from 2 to 15 characters). A station identification must not contain a control character sequence.
When column 32 contains an S, this entry is the symbolic name of the location of the remote station's identification. The symbolic name must not be an array name. If the BSC file is primary or secondary, this symbolic name must refer to the first element of a table.

Columns 48-51 (Remote Terminal)

Blank – System/3 is not used to communicate with the IBM 2770 Data Communication System or the IBM 2780 Data Transmission Terminal.

2770 – The remote terminal is an IBM 2770. If System/3 is transmitting, the output channel on the IBM 2770 is, by default, output channel 1.

2771 – The remote terminal is an IBM 2770, output channel 1.

2772 – The remote terminal is an IBM 2770, output channel 2.

2773 – The remote terminal is an IBM 2770, output channel 3.

2774 – The remote terminal is an IBM 2770, output channel 4.

2780 – The remote terminal is an IBM 2780.

Column 52 (ITB)

I – Intermediate block check (ITB) is used.

Columns 53-54 (Permanent Error Indicator)

01-99, L1-L9, LR, H1-H9 – A permanent error indicator should be used with every BSC file. If you are using more than one BSC file, each should have a permanent error indicator. BSC input/output operations must be conditioned on all permanent error indicators being off.

Columns 55-57 (Wait Time)

Numeric – The length of time in seconds, 1-999, that BSC will wait with no messages being sent or received before a permanent error condition occurs.

Blank – The system convention for timeout, 180 seconds, is used.

Columns 58-59 (Record Available Indicator)

01-99, L1-L9, LR, H1-H9 – A record available indicator is used only when System/3 transmits interspersed with receive (no conversational reply) to System/360-System/370. The record available indicator is set on when System/360-System/370 wishes to transmit to System/3.

Column 60 (Last File)

L – This BSC input file is processed only after all other primary and secondary input files have been processed.

Blank – This BSC input file does not have to be the last input file processed.

Columns 61-62 (Polling Characters)

Alphameric – The polling identification of this station is needed if this station is part of a multipoint network and the BSC is a transmit (output) file.

Blank – This station is not transmitting on a multipoint network.

Columns 65-70 (Remote Device)

Blank – System/3 is not used to communicate with the IBM 2770 Data Communication System or the IBM 2780 Data Transmission Terminal.

1442-1 – The IBM 1442 Card Read Punch (card read) is a remote device used with the IBM 2780 remote terminal.

1442-2 – The IBM 1442 Card Read Punch (card punch) is a remote device used with the IBM 2780 remote terminal.

1443 – The IBM 1443 Printer is a remote device used with the IBM 2780 remote terminal.

0545-3 – The IBM 0545 Card Punch, Model 3, is a remote device used with the IBM 2770 remote terminal.

0545-4 – The IBM 0545 Card Punch, Model 4, is a remote device used with the IBM 2770 remote terminal.

2213-1 – The IBM 2213 Printer, Model 1, is a remote device used with the IBM 2770 remote terminal.

2213-2 – The IBM 2213 Printer, Model 2, is a remote device used with the IBM 2770 remote terminal.

2502-1 – The IBM 2502 Card Reader, Model 1, is a remote device used with the IBM 2770 remote terminal.

2502-2 – The IBM 2502 Card Reader, Model 2, is a remote device used with the IBM 2770 remote terminal.

5496-1 – The 5496 Data Recorder (card read) is a remote device used with the IBM 2770 remote terminal.

5496-2 – The 5496 Data Recorder (card punch) is a remote device used with the IBM 2770 remote terminal.

INPUT SPECIFICATIONS

Columns 7-14 (Filename)

Enter a valid RPG II filename for every input, update, and combined file your program uses.

Columns 15-16 (Sequence)

Enter a 2-digit number to assign a special sequence to record types in a file and to request that the record type sequence be checked by the program. Enter two alphabetic characters to indicate that record type sequence is not checked. Alphabetic characters must be used for a chained file. Within a file, record types with an alphabetic sequence entry must be described before record types with a numeric sequence entry.

Column 17 (Number)

Blank	–	Columns 15-16 contain alphabetic characters (record type sequence is not being checked).
1	–	Columns 15-16 contain numeric characters; only one record of this type is present in each sequenced group.
N	–	Columns 15-16 contain numeric characters; one or more records of this type can be present in the sequenced group.

Column 18 (Option)

Blank	–	Record type must be present.
O	–	Optional. Record type may or may not be present.

Column 18 is used when record types are being sequence checked (columns 15-16 contain numeric characters).

Columns 19-20 (Record Identifying Indicator, **)

01-99	–	Record identifying indicator.
L1-L9	–	Control level indicator used as a record identifying indicator when record type rather than control field signals start of a new control group.
LR	–	Last record indicator.
H1-H9	–	Halt indicator used as a record identifying indicator when checking for a record type that causes an error condition.
**	–	Look-ahead fields.
TR	–	Spread card.

Columns 21-41 (Record Identification Codes)

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

Blank	–	No record identification code is needed.
1-9999	–	Record position of the record identification code.

Not

Blank	–	Either the record identification code is present in the specified record position, or no record identification code is needed.
N	–	Record identification is being used, but the identification code is not present in the specified record position.

C/Z/D

C	–	Entire character.
Z	–	Zone portion of character.
D	–	Digit portion of character.

Remember that many characters have either the same zone or the same digit portion.

AND and OR Relationships

Enter AND in columns 14-16 on the next line of the Input Sheet if more than three record identification code subfields are needed to identify the record. Enter OR in columns 14-15 if either one of the codes may be present to identify the record. A maximum of 20 AND or OR lines in any combination may be used to describe the record identifying code.

Column 42 (Stacker Select)

Blank	–	Cards automatically fall into a predetermined stacker
1-2	–	1442 stacker into which the card type is stacked.
1-4	–	MFCU or MFCM (Model A2) stacker into which the card type is stacked.
1-5	–	MFCM (Model A1) stacker into which the card type is stacked.

Only card from input files and combined files can be stacker selected on input. If this column is blank, cards from the MFCU, MFCM, or 1442 primary hopper are placed in stacker 1, cards from the MFCU secondary hopper are placed in stacker 4 and cards from the secondary MFCM hopper are placed in stacker 4 (MFCM Model A2) stacker 5 (MFCM Model A1).

Column 43 (Packed or Binary Field)

Blank	–	Input field in upacked decimal format.
P	–	Input field in packed decimal format.
B	–	Input field in binary format.

Columns 44-51 (Field Location)

Enter two 1-4 digit numbers to identify the beginning of a field (From) and the end of a field (To) in the input record. These entries are identical for a 1-position field.

Column 52 (Decimal Position)

Blank	–	Alphameric field.
0-9	–	The number of decimal positions in the numeric field named in columns 53-58.

This column must contain an entry for numeric fields.

Columns 53-58 (Field Name)

These columns can contain:

- A valid RPG II field name (see *Definition of Terms* in Chapter 1) for each field defined in Field Location.
- An array name or array element.
- PAGE, PAGE1, or PAGE2 special words.

Columns 59-60 (Control Level)

L1-L9	–	Field described on this line is a control field.
Blank	–	Field described is not a control field.

These columns must be blank for chained or demand files.

Columns 61-62 (Matching Fields)

Enter a matching level identifier (M1-M9) to indicate matching fields and sequence checking when you have two or more input, update, or combined files with match fields. When you have just one input, update, or combined file with match fields this entry causes only sequence checking.

Columns 63-64 (Field Record Relation)

01-99	–	Record identifying indicator assigned to a record type.
L1-L9	–	Control level indicator previously used.
MR	–	Matching record indicator.
U1-U8	–	External indicator previously set.
H1-H9	–	Halt indicator previously used.

The following general rules apply to this entry:

1. All fields without field record relation should be specified before fields with field record relation.

2. All fields with the same field record relation entry should be entered on consecutive lines.

3. All parts of a split control field must have the same field record relation entry and must be described on consecutive specification lines.

Columns 65-70 (Field Indicators)

01-99	–	Field indicator.
H1-H9	–	Halt indicator (when checking for an error condition in the data).

An indicator used in these columns is turned on if the condition tested for is true. For numeric fields, more than one condition may be tested at a time, but only the indicator which reflects the result of the test is turned on, the others are turned off. If a field is alphameric, an indicator can only be specified in Zero or Blank (columns 69-70).

Columns 71-74

These positions should be blank.

Columns 75-80 (Program Identification)

See Chapter 2.

CALCULATION SPECIFICATIONS

Columns 7-8 (Control Level)

Blank	–	Operation done at detail time.
L0	–	Calculation is performed at total time (always on).
L1-L9	–	Calculation operation is done when the appropriate control break occurs or an indicator is set on.
LR	–	Calculation operation is done after the last record has been processed or after LR has been set on.
SR	–	Calculation operation is part of a subroutine.

AN or OR can be entered in these columns to indicate that indicators on the line are in an AND or OR relationship with indicators on the preceding line. A maximum of seven AN, OR, or mixed AN and OR lines are allowed to condition an operation. Entries must be in the order listed.

Columns 9-17 (Indicators)

Enter one to three indicators. Any indicators except 1P and L0 can be used. Columns 9, 12, and 15 may contain blank or *N*. An AND relationship exists between indicators on a line. Additional lines may be used containing indicators in columns 9-17 which are in an AND or OR relationship with those on the first line by entering AN or OR in columns 7-8.

Columns 18-27 (Factor 1) and Columns 33-42 Factor 2

Factor 1 and Factor 2 may contain the following entries:

1. Name of any field that has been defined.

2. Alphameric or numeric literal.

3. Subroutine, table or array name, or array element.

4. Date field name (UDATE, UMONTH, UDAY, UYEAR).

5. Special name, PAGE, PAGE1, or PAGE2.

6. Label for a TAG, BEGSR, or ENDSR operation (Factor 1) or a label for a GOTO or EXSR operation (Factor 2).

7. Filename for a CHAIN, DEBUG, DSPLY, READ, SETLL, or FORCE operation (Factor 2).

Columns 28-32 (Operation)

Enter an operation code, left justified.

Columns 43-48 (Result Field)

Enter the name of the field, table, array, or array element that holds the result of the operation specified in columns 28-32. If the field named in Result Field has not been defined in extension, input, or previous calculation specifications, it must be defined by making entries in columns 49-52.

Columns 49-51 (Field Length)

Blank	–	Field defined elsewhere.
1-256	–	Result field length.

Maximum length of a numeric field is 15 digits; maximum length of an alphameric field is 256 characters. Entry must be right justified.

Column 52 (Decimal Position)

Blank	–	Alphameric field or numeric field described elsewhere.
0-9	–	Number of decimal places in a numeric result field.

Column 53 (Half Adjust)

Blank	–	Do not half adjust the Result Field.
H	–	Half adjust the Result Field.

Half adjust is allowed only with arithmetic operations.

Columns 54-59 (Resulting Indicators)

Enter any of the following indicators: 01-99, H1-H9, L1-L9, LR, OA-OG, and OV. Columns 54-59 are used for four purposes:

1. To test the value of the result field after an arithmetic operation.

2. To check the outcome of a CHAIN, LOKUP, COMP, TESTB, or TESTZ operation.

3. To specify which indicators to SETON or SETOF.

4. To indicate end of file for the READ operation code.

Arithmetic Operations: Enter up to three indicators to be turned on whenever the result is positive (indicator in columns 54-55), 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 is greater than Factor 2 (indicator in columns 54-55), Factor 1 is less than Factor 2 (indicator in columns 56-57), or Factor 1 is equal to Factor 2 (indicator in columns 58-59).

LOKUP Operation: Enter one or two indicators in High, Low, Equal, High and Equal, or Low and Equal. If there is an entry in the High or Low columns, the table name in Factor 2 should be specified as ascending or descending on the Extension Sheet.

TESTB Operation: Resulting indicators have the following meaning for this operation:

- *Columns 54-55:* An indicator in these columns is turned on if each bit specified in Factor 2 is off in the Result Field.
- *Columns 56-57:* An indicator in these columns is turned on if two or more bits were tested and of mixed status (some bits on and some bits off).
- *Columns 58-59:* An indicator in these columns is turned on if each bit specified in Factor 2 is on in the Result Field.

TESTZ Operation: Enter one to three indicators to reflect the zone of the leftmost character in the Result Field, as follows:

- *Columns 54-55:* Turned on by the zone portion of the characters & and A-I.
- *Columns 56-57:* Turned on by the zone portion of the characters } (bracket), – (minus), and J-R.
- *Columns 58-59:* Turned on by the zone portion of any character not listed above.

CHAIN Operation: Enter an indicator (optional) in columns 54-55 to be turned on in the case of a record-not-found condition.

SETON and SETOF Operations: Enter up to three indicators in columns 54-59 to be turned on (SETON) or turned off (SETOF).

READ Operation: Enter an indicator in columns 58-59 to be turned on after each read operation if an end-of-file condition is reached. Once end-of-file is reached, a halt occurs after each read operation if no indicator is entered.

Columns 60-74 (Comments)

Enter any meaningful information you wish to help you understand or remember what you are doing in each specification line.

OUTPUT-FORMAT SPECIFICATIONS

Columns 7-14 (Filename)

Enter a valid RPG II filename for each output, combined, and update file used by your program. Each filename need be specified only once, on the first line describing that file.

Columns 14-16 (AND/OR Relationship)

Enter AND in columns 14-16 or OR in columns 14-15 if output records are in an AND or OR relationship.

Column 15 (Type)

H	–	Heading records.
D	–	Detail records.
T	–	Total records.
E	–	Exception records.

Columns 16-18 (Add a Record)

Enter ADD in these columns if records are added to an input, update, or output disk file. An *A* must also be coded in column 66 of the File Description sheet for the file to which a record is added.

Column 16 (Stacker Select/Fetch Overflow)

Blank	–	Cards automatically fall into certain stackers as follows: Primary hopper – stacker 1 (MFCU and MFCM) Secondary hopper – stacker 4 (MFCU and MFCM Model A2) stacker 5 (MFCM Model A1)
1-5	–	Indicates the stacker you wish.
F	–	Fetch overflow.

Only combined or output files can be stacker selected on Output specifications. Stacker selection on output overrides stacker selection on input.

If *F* is entered, the overflow routine is fetched when overflow occurs, before the usual time in the cycle.

Columns 17-22 (Space/Skip)

If these columns are blank, single spacing occurs after each line is printed. Spacing and skipping are not allowed on the printer keyboard.

Columns 17-18 (Space)

Enter a number (0-3) under the appropriate column to indicate the number of lines spaced before or after a line is printed.

Columns 19-22 (Skip)

Blank	–	No skipping.
01-99	–	Lines 1-99.
A0-A9	–	Lines 100-109.
B0-B2	–	Lines 110-112.

Enter one of the 2-digit numbers listed above to indicate the next line printed. All line numbers between are bypassed. Enter the number in the Before or After columns, depending on whether you want skipping to occur before or after the line is printed.

Columns 23-31 (Output Indicators)

Enter one to three indicators. Any indicator may be used. Columns 23, 26, and 29 may contain blank or *N*. *N* preceding an indicator means the output operation will be done only if the indicator is not on. An AND relationship exists between indicators on a line. Additional lines of indicators in an AND or OR relationship may be used by entering AND in columns 14-16 or OR in columns 14-15 of each additional line (up to 20).

Columns 32-37 (Field Name)

Enter one of the following to name every field written out:

- Any field name previously defined in this program.
- The special words, PAGE, PAGE1, PAGE2, *PLACE, *PRINT, UDATE, UDAY, UMONTH, and UYEAR.
- A previously defined table name, array name, or array element.

These columns must be blank if a constant is entered on columns 45-70 of the line. If an entry is made under Field Name, columns 7-22 must be blank.

Column 38 (Edit Codes)

Enter an edit code in column 38 when you want to:

1. Suppress leading zeros for a numeric field.
2. Omit a sign from the low order position of a numeric field.
3. Punctuate a numeric field without setting up your own edit word.

A table summarizing the edit codes that can be used is printed above columns 45-70 on the Output Sheet.

Column 39 (Blank After)

B	–	Field is reset to blank or zero after writing.
Blank	–	Field is not reset after writing.

Numeric fields are set to zero and alphameric fields are set to blanks. This column must be blank for look-ahead and update fields. If the field name specified with Blank After is a table name, the element of the table looked up last will be blanked or zeroed.

Column 40-43 (End Position in Output Record)

Columns 40-43 indicate the location on the output record of the field or constant written. Enter the number of the position occupied by the rightmost character of the output field. The End Position entry must not be greater than the record length.

Printing on Cards (MFCU)

If you want to print fields on cards in *other* than the positions which correspond to the punch positions, you must:

1. Name the field in columns 32-37.
2. Place an * (asterisk) in column 40.
3. Specify an end position for that field in columns 40-43. The maximum end position entry is 128.

Printing on Cards (MFCM)

If you want to print fields on cards in print positions other than those provided by *PRINT, you must:

1. Name the fields, in columns 32-37.
2. Specify a print head number (1-6) in column 41.
3. Specify a print end position (01-64) in columns 42 and 43 (the leading zero in column 42 is mandatory).

Column 44 (Packed or Binary Field)

Blank	–	Field is unpacked numeric or alphameric data.
P	–	Field is packed decimal numeric data.
B	–	Field is in binary format.

Packed and binary fields can be output to disk, tape, 1442, or MFCM, but should not be printed. Column 44 must be blank with *PLACE fields, *PRINT fields, and asterisk in column 40.

Columns 45-70 (Constant or Edit Word)

Constant: The following rules apply to constants:

1. Field Name (columns 32-37) must be blank.
2. A constant must be enclosed in apostrophes. Enter the leading apostrophe in column 45.
3. An apostrophe in a constant must be represented by two apostrophes.
4. Up to 24 characters of constant information can be placed in one line. Additional lines may be used, but each line must be treated as a separate line of constants. The end position of each line must appear in columns 40-43.

Edit Word: Enter any edit word to specify editing of numeric fields. Edit words must be enclosed by apostrophes. Constants are allowed within edit words.

Edit words are not used with edit codes. However, when edit codes 1-4, A-D, and J-M are used, columns 45-47 may contain an * (to denote asterisk fill) or a $ (to denote a floating dollar sign).

Appendix H. Programming Tips

MAIN STORAGE SAVING TECHNIQUES

When your program is too large to fit into the execution main storage size, you may want to use some main storage saving techniques to help reduce the program size. Before you can use these techniques effectively, however, you need to understand (1) how the RPG II Compiler creates overlays to make a program fit into the main storage available for execution and (2) how the compiler determines when a program is too large to fit into the main storage available for execution. This section will discuss the overlay process and then give you some suggestions for saving main storage.

Overlay Process for Models 10 and 12

When your program exceeds the available storage for program execution, the RPG II compiler places some RPG II object program routines on disk. These routines are then called into main storage as they are needed by your program. This is known as the overlay process.

When the overlay process is used, main storage is divided into two main parts: the Root segment and the Overlay area.

The Root segment contains constants and data used more than once during program execution. For this reason, the Root segment always remains in main storage. The Root segment may be used by routines in the Overlay area. The Root segment can call a routine in the Overlay area by using a branch instruction.

The Main Overlay area contains the major routines of the RPG II object program. Routines in this area may be called by the Root segment or by other routines in the same main overlay.

Some large programs require that storage be divided into two additional parts: the Secondary Root segment and the Suboverlay area. The Secondary Root segment is used to supplement the Root segment. If the Root segment and the Overlay area fill main storage, the Secondary Root segment is not created. The Suboverlay area, created by the RPG II compiler, contains subroutines and other RPG II code needed to support a routine in the Main Overlay area. Figure H-1 shows the location of the main storage areas.

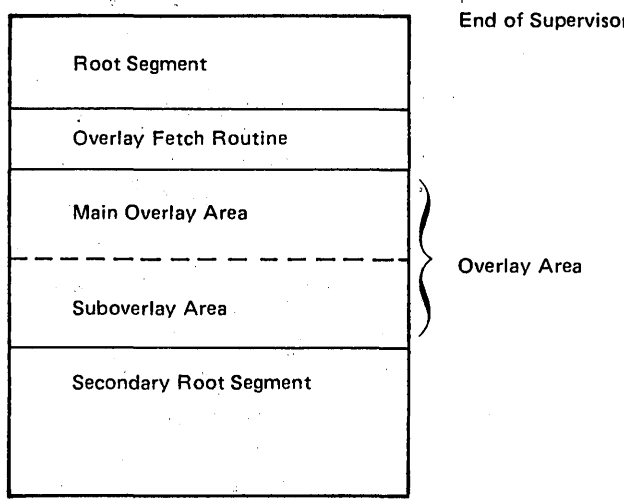


Figure H-1. Models 10 and 12 RPG II Storage Map

Creating the Overlays

In order to create overlays, the compiler must first determine which routines will go into the Main Overlay areas and which routines will go into the Suboverlay areas. Then it calculates the size of the largest Main Overlay and the size of the largest Suboverlay. These sizes are rounded off upwards in increments of 256 bytes (1 sector). The compiler then adds the lengths of the Root segment, the largest Main Overlay, and the largest Suboverlay. If the sum is larger than the available storage, your program is too large, and main storage core saving techniques must be used if the program is to be run.

Special Open/Close

Special Open/Close is used when the overlay requirements for Open and Close exceed the overlay requirements for the rest of the program.

Special Open/Close can be easily identified because overlay $##002 is the first overlay identified in the main storage usage map (see Figure H-2).

The first load will bring in the Root, the Overlay Fetch Routine, the Overlay Fetch Area, and a special transfer vector to call the Open overlay. Open is completely self-contained and does not need any of the non-overlay code. When Open is complete, Overlay 1 is loaded. Overlay 1 consists of all code that is identified as non-overlay and was not loaded during the first load. The program then executes as a normal overlay program until Close is needed. At this time, Close is brought into main storage starting at the Overlay Fetch Area and using as much main storage as is needed.

The Overlay Fetch Area size for the rest of the program can be found by subtracting the start of the Overlay Fetch Area from the lowest start address of the non-overlay code that was not included in the first load. For example, INPUT CTRL RTN starts at 15E1, so 15E1 minus 13E1 equals X'200' – the Overlay Fetch Area size.

Saving Core

When the compiler finds that your program is too large, an error message is written. You can reduce the storage needed for your program either by using some general main storage saving techniques or by reducing the size of the overlays.

General Main Storage Saving Techniques

Some of the techniques you can use are:

1. Divide your program into separate tasks, creating a separate program for each task. For example, suppose you want to update a file and print a listing of the updated file. You can save main storage by updating the file with one program and printing the listing with another program.

2. Eliminate unreferenced indicators. Eliminating unreferenced indicators can eliminate the instructions required to set the indicators on and off.

3 Eliminate unnecessary conditioning indicators. Two possible forms of unnecessary indicator tests are:

   a. If only one type of input record is to be processed, the indicator associated with that record will always be on except during the first detail output time. It is, therefore, not necessary for any calculation to be conditioned with this indicator.

   b. When two subsequent operations on the same result field are conditioned on opposite indicator conditions, one of the conditions is not necessary. For instance, the N09 conditioning is not required in this example:

```
N09   Z-ADD   FLD    FLDB
 09   Z-ADD   FLDC   FLDB
```

   Note: This technique may not work for certain operations if the same field is used as the result field and as factor 1 or factor 2.

4. Reuse calculation work areas and temporary hold areas. Once the data stored in these areas is used for the last time in a given cycle, the area is available. Reusing these areas can eliminate the need for two or more additional areas to be defined.

   Note: Be sure you do not mix alphameric and numeric fields.

5. Reuse input field name areas. In some instances, two or more input files may have fields that always contain identical information. These fields can be given the same field name in order to use the same main storage area.

   Another way to reuse input field areas is to use the same names for fields in two files. This can be done only if both fields have the same attributes (length, alphameric/numeric, packed binary) and each field is only used in the cycle in which the record is processed. Both files cannot be used during the same cycle.

6. Reduce calculation work area sizes. Be sure that no work area has been defined as larger than it needs to be. This may cause a warning that the result field may not be large enough, but if you know that the largest possible number will fit into the areas specified, you may continue.

7. Include the necessary intervening blanks when describing alphameric fields and constants for output. This will make the fields adjacent. The output optimization phase will move all adjacent fields and constants with one instruction instead of using one instruction to move each line.

Not Optimized	*Optimized*
5 'DAILY'	18 'DAILY TRANSACTIONb̸'
17 'TRANSACTION'	26 'REGISTER'
26 'REGISTER'	

Note: This programming tip is valid only when the alphameric fields and constants are unique; in this case the fields are stored in consecutive storage locations and can be moved to a print line with one instruction. When using identical output constants, a move instruction is required for each print line moved.

8. Design files to contain record lengths that are an even multiple of 256 bytes or that will divide into 256 bytes an even number of times.

9. Design files so that match fields and control fields are assigned the same position within all record types.

10. Do not designate a field as numeric unless the field is to be used in a numeric operation in the program. This can save on the amount of storage required to store the field and can allow the input and output fields transfer routine to be optimized.

11. Use only one type of file organization in a program (indexed, direct, or sequential). Also, use the same method of processing where possible. This can reduce the disk data management main storage requirements. Some unit record data management can also be eliminated by transferring unit record files to disk.

12. On Models 10 and 12 systems, the shared input/output access method (SIAM) can be used to process disk files. This will reduce the storage required even on programs with only one disk file.

    Note: Using SIAM may decrease program throughput.

13. Group calculation statements together that are conditioned by the same indicators. When a large number of indicators are required, try to use GOTO or EXSR to reduce the number of indicator tests required on each statement.

14. When using TESTB, BITON, or BITOF, use the actual bit pattern in factor 2.

15. Do not use half adjust unless absolutely necessary.

16. Try to use either factor 1 or factor 2 as the result field whenever possible.

17. Try to use numeric fields of the same length and with the same number of decimal positions. If the fields cannot be the same length, try to have the number of decimal positions the same. (see Appendix I. for an example.)

18. Do not sequence check your records unless absolutely necessary.

19. Use OR lines rather than multiple record lines because OR lines require less code.

20. Specify the fields in a record in ascending order by record position.

21. Do not use halt indicators unless absolutely necessary.

Reduce the Overlay Size (Models 10 and 12)

To reduce the size of the overlay, you can reduce the size of the Root segment or the Overlay areas. First, however, you must identify the contents of the Root segment and the largest overlays in main storage. Then you can determine if the contents of these areas can be reduced to fit into the main storage available for execution.

The contents of the Root segment, main Overlay area, and Suboverlay area can be found by using the program listing.

Two sections of the program listing are used to determine the contents of the main Overlay and Suboverlay areas. The section shown in Figure H-3 tells the:

1. Overlay name

2. Number of sectors in the overlay

3. Start address of the overlay

The start address separates main overlays and suboverlays. Two start addresses appear in the Start Address column. The lower address (1A97) identifies a main overlay; the higher address (1C97) identifies a suboverlay.

The Text Sectors column indicates the largest overlays. In Figure H-3, overlays 002 and 005 are the largest suboverlays; overlays 007 and 008 are the largest main overlays.

Relate the name given in the Overlay Name column shown in Figure H-3 to the Core Usage of RPG II Code section shown in Figure H-4. The Name and Title columns in this section identify the routines or subroutines in the overlay.

Note: If overlay 001 does not appear in the Overlay Name column, a special Open/Close overlay construction has taken place. When this occurs, overlay 001 is not treated as an overlay, but remains in main storage.

After identifying the Root segment and the largest main overlays and suboverlays, you can determine whether they contain routines that can be manipulated to reduce the overlay size. The following routines can be controlled:

1. Input Records

2. Detail Calculations

3. Total Calculations

4. Detail Output

5. Total Output

Following are some main storage saving techniques that can be used for these routines. These techniques may not necessarily work for all programs.

Input Records: One or more of the input or update files can be processed as a demand or chained file, using the READ or CHAIN operation code. With a demand or chained file, the instructions to read the file can be moved into the Total or Detail Calculations routine.

Note: Total calculations will not be done on the first cycle.

```
                                                  CORE USAGE OF RPGII CODE
        START  NAME IF  CODE     NAME    TITLE
        ADDR   OVERLAY  LENGTH
        1000            0300     RGROOT  ROOT
        1300            00E1     RGSUBS  OVERLAY FETCH ROUTINE
        13E1            0400     RGSUBS  OVERLAY FETCH AREA
        163B            0091     RGMAIN  INPUT MAINLINE
Root    174C            000B     RGSUBS  TRANSFER VECTOR
        16CC            005A     RGSUBS  RECORD ID
        1726            0026     RGSUBS  CONTROL FIELDS
        15E1            005A     RGSUBS  INPUT CTRL RTN
        13E1   $##002   0008     RGSUBS  SUBSEG
        13E9   $##002   0145     $$MFRD  MFCU READ
        1757            006F     RGMAIN  INPUT FIELDS
        17CC            0069     RGMAIN  DETAIL CALCS
        18EA            000B     RGSUBS  TRANSFER VECTOR
Root    17C6            0006     RGSUBS  CONSTANTS
        18A7            0043     $$PGRI  RESET RESULTING INDR
        1835            0072     $$PGAA  TAG  FETCH
        13E1   $##003   0169     $$PGLC  LOKUP ROUTINE
        18F5            000B     RGMAIN  TOTAL OUTPUT
        199D            0024     RGMAIN  LR & OVERFLOW PROCESSING
Root    19C1            000B     RGSUBS  TRANSFER VECTOR
        1900            009D     RGSUBS  OUTPUT CTRL RTN
        13ED   $##004   001D     RGSUBS  OVERFLOW SUBSEGMENT
        13E1   $##004   000C     RGSUBS  SUBSEG
        140A   $##004   00FB     $$LPRT  5203 PRINT
        149F   $##005   0085     RGMAIN  OPEN
        161F   $##005   0021     RGSUBS  TRANSFER VECTOR
        13E1   $##005   009D     RGSUBS  OUTPUT CTRL RTN
        148A   $##005   0015     RGSUBS  CONSTANTS
        147E   $##005   000C     RGSUBS  SUBSEG
        1524   $##005   00FB     $$LPRT  5203 PRINT
        157C   $##006   0021     RGMAIN  CLOSE

        174C   $##006   0016     RGSUBS  TRANSFER VECTOR
        13E1   $##006   009D     RGSUBS  OUTPUT CTRL RTN
        148A   $##006   00F2     RGSUBS  CONSTANTS
        159D   $##006   00B4     RGSUBS  LR PROCESSING
        147E   $##006   000C     RGSUBS  SUBSEG
        1651   $##006   00FB     $$LPRT  5203 PRINT

                        02529    XRGE16  TOTAL CORE USAGE REQUIRED TO EXECUTE
                        03333            TOTAL CORE USAGE REQUIRED TO EXECUTE WITHOUT OVERLAYS
      OVERLAY NAME   RELATIVE START C/T/S  # TEXT SECTORS   START ADDRESS
         $##001               00  00  05            06           13E1
         $##002               00  00  0C            02           13E1
         $##003               00  00  0F            02           13E1
         $##004               00  00  12            02           13E1
         $##005               00  00  15            03           13E1
         $##006               00  01  01            04           13E1
    TOTAL NUMBER OF LIBRARY SECTORS REQUIRED    30
```

Figure H-2. RPG II Usage Map (Models 10 and 12)

Detail or Total Calculations: Use the following techniques:

1. Use subroutine calculations. In some instances this may increase, rather than decrease, the storage required due to the nature of the existing calculation routines.

   However, it may reduce the overall main storage requirements.

   Note: If one subroutine calls another subroutine, both subroutines must be in main storage at the same time. This may increase the size of the suboverlay area and the total storage required. To ensure the smallest requirement, do not call a subroutine from another subroutine.

2. Eliminate exception output if possible. This will move the logic for those output operations to either Total or Detail Output routines.

3. Eliminate read and/or chain operations by using matching records and processing consecutively. This will move the logic to Input Records routine.

4. Move part of the detail calculation logic to total calculations (or total calculation logic to detail calculations).

   Note: Total calculations will not be done on the first cycle.

Detail or Total Output: Use the following techniques:

1. Use exception output. This will move part of the output logic to Detail or Total Calculation routines.

2. Do some of the output at total (or detail) output time. This moves logic to the Total (or Detail) Output routine.

3. Do not specify blank after for fields. Instead, clear them at the beginning of detail or total calculations.

	OVERLAY NAME	RELATIVE START C/T/S			# TEXT SECTORS	START ADDRESS
	$##001	00	00	15	01	1C97
Largest Suboverlays	$##002	00	00	17	04	1C97
	$##003	00	01	04	01	1C97
	$##004	00	01	06	01	1A97
	$##005	00	01	08	04	1C97
	$##006	00	01	0D	02	1A97
Largest Overlays	$##007	00	01	10	03	1A97
	$##008	00	01	14	03	1A97

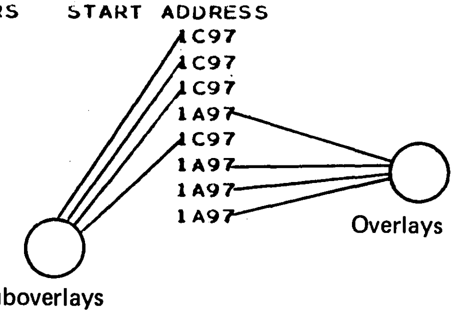


Figure H-3. Overlay Identification Area (Models 10 and 12)

	START ADDR	NAME IF OVERLAY	CODE LENGTH	NAME	CORE USAGE OF RPGII CODE TITLE
	1000		082C	RGROOT	ROOT
	1971		0126	RGSUBS	OVERLAY FETCH ROUTINE
	182C		0145	$$MFRD	MFCU READ
	1A97		0600	RGSUBS	OVERLAY FETCH AREA
	214B		00A0	RGMAIN	INPUT MAINLINE
	221F		002C	RGSUBS	TRANSFER VECTOR
	21EB		0034	RGSUBS	RECORD ID
	1C97	#$$001	0026	RGSUBS	CONTROL FIELDS
	2097		00B4	RGSUBS	INPUT CTRL RTN
	1C97	#$$002	0008	RGSUBS	SUBSEG
	1C9F	#$$002	0088	$$ISUL	DISK IDX SEQ UPDATE BY LIMITS
	1D27	#$$002	0019	$$SRCR	SYSTEM SUBR
	1D40	#$$002	0043	$$SRIC	SYSTEM SUBR
	1D83	#$$002	002B	$$SRIF	SYSTEM SUBR
	1DAE	#$$002	0046	$$SRIU	SYSTEM SUBR
Suboverlay 002	1DF4	#$$002	0028	$$SRLP	SYSTEM SUBR
	1E1C	#$$002	0081	$$SRMO	SYSTEM SUBR
	1E9D	#$$002	0015	$$SRPD	SYSTEM SUBR
	1EB2	#$$002	007F	$$SRRC	SYSTEM SUBR
	1F31	#$$002	0029	$$SRRI	SYSTEM SUBR
	1F5A	#$$002	001C	$$SRTC	SYSTEM SUBR
	1F76	#$$002	002F	$$SRBP	SYSTEM SUBR
	1FA5	#$$002	0015	$$SRRD	SYSTEM SUBR
	224B		003C	RGMAIN	INPUT FIELDS
	2287		0016	RGSUBS	TRANSFER VECTOR
	1C97	#$$003	00D4	$$PGBD	CONVERT TO DECIMAL
	1A97	#$$004	0016	RGMAIN	DETAIL CALCS
	229D		0001	RGSUBS	CONSTANTS
	234E		005F	RGMAIN	DETAIL OUTPUT
	23AD		0016	RGSUBS	TRANSFER VECTOR
	229E		00B0	RGSUBS	OUTPUT CTRL RTN
	1C97	#$$005	000C	RGSUBS	SUBSEG
	1CA3	#$$005	0088	$$ISUL	DISK IDX SEQ UPDATE BY LIMITS
	1D2B	#$$005	0019	$$SRCR	SYSTEM SUBR
	1D44	#$$005	0043	$$SRIC	SYSTEM SUBR
	1D87	#$$005	002B	$$SRIF	SYSTEM SUBR
	1DB2	#$$005	0046	$$SRIU	SYSTEM SUBR
Suboverlay 005	1DF8	#$$005	0028	$$SRLP	SYSTEM SUBR
	1E20	#$$005	0081	$$SRMO	SYSTEM SUBR
	1EA1	#$$005	0015	$$SRPD	SYSTEM SUBR
	1EB6	#$$005	007F	$$SRRC	SYSTEM SUBR
	1F35	#$$005	0029	$$SRRI	SYSTEM SUBR
	1F5E	#$$005	001C	$$SRTC	SYSTEM SUBR
	1F7A	#$$005	002F	$$SRBP	SYSTEM SUBR
	1FA9	#$$005	0015	$$SRRD	SYSTEM SUBR
	23C3		000B	RGMAIN	TOTAL OUTPUT
	1AC3	#$$006	0024	RGMAIN	LR & OVERFLOW PROCESSING
	23CE		0010	RGSUBS	CONSTANTS
	1AA3	#$$006	0020	RGSUBS	OVERFLOW SUBSEGMENT
	1A97	#$$006	000C	RGSUBS	SUBSEG
	1AE7	#$$006	000D	RGSUBS	SUBSEG
	1AF4	#$$006	0160	$$LPRT	5203 PRINT
Overlay 007	1B4F	#$$007	0018	RGMAIN	CLOSE
	1AA3	#$$007	00AC	RGSUBS	CONSTANTS
	1B67	#$$007	0047	RGSUBS	LR PROCESSING
	1A97	#$$007	000C	RGSUBS	SUBSEG
	1BAE	#$$007	0160	$$LPRT	5203 PRINT
	1AA3	#$$008	009A	RGMAIN	OPEN
Overlay 008	1A97	#$$008	000C	RGSUBS	SUBSEG
	1B4A	#$$008	0160	$$LPRT	5203 PRINT
	1B3D	#$$008	000D	RGSUBS	SUBSEG
			05086	RAFU06	TOTAL CORE USAGE REQUIRED TO EXECUTE

Figure H-4. RPG II Usage Map (Models 10 and 12)

PERFORMANCE IMPROVEMENT TECHNIQUES

Some relatively simple program changes may make significant improvements in your program's performance. However, these performance techniques will not improve performance in all programs. Therefore, study these techniques and determine if you think they will improve your program's performance before you use them. The performance improvement techniques are:

1. Unblock all randomly processed indexed files. Blocking gains nothing since each record has its own index entry with the direct address of the record.

2. Block all sequentially processed indexed files.

3. Use the core index. For a minimum cost in main storage this allows the system to read the single track of indexes it needs rather than reading the entire index to look for an extry.

4. Double buffer printer and card input files.

5. Reduce or eliminate blocking of sequential files and double the buffer instead. For example, instead of using a block of 1600 bytes with 80 byte records, use a block of 800 bytes and double buffer.

Dual Programming Feature (Models 10 and 12)

When using the dual programming feature (DPF), the following should be considered:

1. The compiler can be run in either program level; however, running compilations in both program levels simultaneously may produce erroneous results.

2. The printer must be available to the program level used by the compiler, unless No-List is requested (specified by a B in column 11 of the control card specifications).

3. Data management routines and user-written subroutines must be cataloged as permanent library entries. If a temporary program is being run in the opposite level, the link editor is prevented from cataloging any object modules in the library from which the temporary program came.

4. A program running in the opposite program level must not change the object library on the program pack.

5. During compilation, some logging is forced to the printer. If the other level attempts to print using Halt/Syslog during compilation, erroneous printing may result.

6. If the RPG compiler is run in program level 2 on a 64K system, a minimum partition size of 5-1/4K (5,376) must be assigned to level 2.

If an RPG II program is being compiled in program level 2, the addresses on the RPG II core usage map may indicate program level 1. This would be true if the program level 2 start address plus the length of the RPG II object program exceeded 64K; in such a case, program level 1 is assumed for the object program.

SAMPLE PROGRAM LISTINGS

Figures H-5 and H-6 are program listings for the Models 10 and 12 and Model 15 systems.

Figure H-7 shows the RPG routines used by the Model 15 system. This chart may be used if you are doing a separate link edit on your program.

System program compiled on — Compiler version — Date

```
SYSTEM/3 MODEL 10 DISK     RPGII   VERSION 09, MODIFICATION LEVEL 00        06/20/73

      0101 H     008                                                        SAMPL1

      0102 F*************************************************               SAMPL1
      0103 F*                                               *               SAMPL1
      0104 F* THIS PROGRAM -                                *               SAMPL1
      0105 F*                                               *               SAMPL1
      0106 F*   1. LOADS 100 RECORDS TO AN INDEXED FILE.    *               SAMPL1
      0107 F*                                               *               SAMPL1
      0108 F*   2. READS ONE RECORD FROM FILE $SOURCE FOR   *               SAMPL1
      0109 F*      INPUT. THE FILE $SOURCE IS BUILT WHEN    *               SAMPL1
      01091F*      SAMPLE PROGRAM SAMPL2 IS COMPILED BY     *               SAMPL1
      01092F*      GIVING A RETAIN-T PARAMETER TO THE       *               SAMPL1
      01093F*      FILE $SOURCE.                            *               SAMPL1
      01094F*                                               *               SAMPL1
      0110 F*   3. CREATES THE OUTPUT DATA USING A          *               SAMPL1
      0111 F*      LOOP IN THE CALCULATION SPECIFICATIONS.  *               SAMPL1
      0112 F*                                               *               SAMPL1
      0113 F*   4. USES KEYS FROM 000005 THROUGH 000500     *               SAMPL1
      0114 F*      IN INCREMENTS OF 5.                      *               SAMPL1
      0115 F*                                               *               SAMPL1
      0116 F*   5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2   *               SAMPL1
      0117 F*      TO VERIFY THAT THE FILE WAS PROPERLY     *               SAMPL1
      0118 F*      LOADED.                                  *               SAMPL1
      0119 F*                                               *               SAMPL1
      0120 F*************************************************               SAMPL1
0001  0121 F$SOURCE IP  F  96  96           DISK                            SAMPL1
0002  0122 FDISKOUT O   F 256 128 06AI     1 DISK                    01     SAMPL1
0003  0123 FPRINTER O   F  96  96           PRINTER                         SAMPL1

0004  0201 I$SOURCE NS  01                                                  SAMPL1
0005  0202 I                                      1    1 NUDATA             SAMPL1

0006  0301 C     01               Z-ADDO        COUNT   60                  SAMPL1
0007  0302 C     01               Z-ADDO        RECNBR  30                  SAMPL1
0008  0303 C           REPEAT     TAG                                       SAMPL1
0009  0304 C     01    COUNT      ADD  5        COUNT                       SAMPL1
0010  0305 C     01    RECNBR     ADD  1        RECNBR                      SAMPL1
0011  0306 C     01    COUNT      COMP 505                    02            SAMPL1
0012  0307 C     01N02            EXCPT                                     SAMPL1
0013  0308 C     01N02            GOTO REPEAT                               SAMPL1
0014  03081C                      SETON                    LR               SAMPL1
0015  0309 CLR         RECNBR     SUB  1        RECNBR                      SAMPL1

0016  0401 OPRINTER T  204    LR                                            SAMPL1
0017  0402 O                                  20 'SAMPLE PROGRAM 1 HAS'     SAMPL1
0018  0403 O                                  27 'LOADED'                   SAMPL1
0019  0404 O                        RECNBRZ   31                            SAMPL1
0020  0405 O                                  39 'RECORDS'                  SAMPL1
0021  0406 O                                  61 'INTO AN INDEXED FILE.'    SAMPL1
0022  0408 O           T  2       LR                                        SAMPL1
0023  0409 O                                  21 'KEYS ARE IN ASCENDING'    SAMPL1
0024  0410 O                                  42 'SEQUENCE STARTING AT'     SAMPL1
0025  0411 O                                  64 '000005 AND INCREASING'    SAMPL1
0026  0412 O                                  84 'IN INCREMENTS OF 5.'      SAMPL1
0027  0413 O           T      01  LR                                        SAMPL1
0028  0414 O                                  21 'SAMPLE PROGRAM 2 WILL'    SAMPL1
0029  0415 O                                  44 'PRINT FROM THE INDEXED'   SAMPL1
0030  0416 O                                  65 'FILE TO SHOW THAT IT'     SAMPL1
0031  0417 O                                  86 'WAS PROPERLY LOADED.'     SAMPL1
0032  0501 ODISKOUT E         01N02                                         SAMPL1
0033  0502 O                        COUNT      6                            SAMPL1
0034  0503 O                                  94 'RECORD NUMBER'            SAMPL1
0035  0504 O                        RECNBR   128                            SAMPL1
```

Source Statements

Statement Number

Figure H-5 (1 of 2). Models 10 and 12 Program Listing

```
 INDICATORS USED
       LR 01 02

RG 314 UNREFERENCED FIELD NAMES
 STMT#  NAME
  0005 NUDATA

  FIELD NAMES USED
 STMT#  NAME   DEC  LGTH  DISP
  0006 COUNT    0   006   0105
  0007 RECNBR   0   003  |0108|

  LABELS USED
 STMT#  NAME    TYPE
  0003 REPEAT   TAG

  ERROR SEVERITY                                  TEXT
RG  314 W        FIELD, TABLE OR ARRAY NAME DEFINED BUT NEVER USED.


                                       CORE USAGE OF RPGII CODE
     START  NAME IF  CODE    NAME     TITLE
     ADDR   OVERLAY  LENGTH
     1500            0642    RGROOT   ROOT
     1B9A            0091    RGMAIN   INPUT MAINLINE
     1C2B            0046    RGSUBS   RECORD ID
     1C71            0026    RGSUBS   CONTROL FIELDS
     1B42            0050    RGSUBS   INPUT CTRL RTN
     1B92            0008    RGSUBS   SUBSEG
     1C97            0027    $$CSIP   5444 CONSEC INPUT
     1CBE            0079    $$SRBR   SYSTEM SUBR
     1D37            0026    $$SRUA   SYSTEM SUBR
     1D5D            001C    $$SRTC   SYSTEM SUBR
     1D79            0081    $$SRMU   SYSTEM SUBR
     1DFA            0043    $$SRSB   SYSTEM SUBR
     1E3D            0038    $$SRUI   SYSTEM SUBR
     1E75            002F    $$SRBP   SYSTEM SUBR
     1EB2            000B    RGMAIN   TOTAL CALCS
     1EA4            000E    RGSUBS   CONSTANTS
     1E8D            001D    RGMAIN   INPUT FIELDS
     1F88            004B    RGMAIN   DETAIL CALCS
     1F83            0005    RGSUBS   CONSTANTS
     1EDA            009D    RGSUBS   OUTPUT CTRL RTN
     2003            0043    $$PGRI   RESET RESULTING INDR
     1FD3            0030    RGSUBS   EXCEPTION
     1F77            000C    RGSUBS   SUBSEG
     2046            0059    $$IOUT   5444 INDEXED OUTPUT
     2137            001C    $$SRDF   SYSTEM SUBR
     209F            0098    $$SRBI   SYSTEM SUBR
     2153            000B    RGMAIN   TOTAL OUTPUT
     2187            0024    RGMAIN   LR & OVERFLOW PROCESSING
     216A            001D    RGSUBS   OVERFLOW SUBSEGMENT
     215E            000C    RGSUBS   SUBSEG
     21AB            00FB    $$LPRT   5203 PRINT
     23A3            002D    RGMAIN   CLOSE
     22A6            00E9    RGSUBS   CONSTANTS
     23D0            0076    RGSUBS   LR PROCESSING
     238F            0014    RGSUBS   LR CALCS
     2446            0071    RGMAIN   OPEN

                    04023    SAMPL1   TOTAL CORE USAGE REQUIRED TO EXECUTE


TOTAL NUMBER OF LIBRARY SECTORS REQUIRED   18
```

Displacement to right end of field from start of Root.

Contains fields, buffers, constants, and work areas used by the RPG program.

Disk space required in the object library by the program.

Figure H-5 (2 of 2). Models 10 and 12 Program Listing

System program compiled on. → SYSTEM/3 MODEL 15

Compiler version → RPGII VER-xx, MOD-00

Size* → 12K

Date → xx/xx/xx

Source statements

Statement number

```
      0101 H      008                                                        SAMPL1

      0102 F***************************************************              SAMPL1
      0103 F*                                                 *              SAMPL1
      0104 F* THIS PROGRAM -                                  *              SAMPL1
      0105 F*                                                 *              SAMPL1
      0106 F*   1. LOADS 100 RECORDS TO AN INDEXED FILE.      *              SAMPL1
      0107 F*                                                 *              SAMPL1
      0108 F*   2. READS ONE RECORD FROM FILE $SOURCE FOR     *              SAMPL1
      0109 F*      INPUT. THE FILE $SOURCE IS BUILT WHEN      *              SAMPL1
      01091F*      SAMPLE PROGRAM SAMPL2 IS COMPILED BY       *              SAMPL1
      01092F*      GIVING A RETAIN-T PARAMETER TO THE         *              SAMPL1
      01093F*      FILE $SOURCE.                              *              SAMPL1
      01094F*                                                 *              SAMPL1
      0110 F*   3. CREATES THE OUTPUT DATA USING A            *              SAMPL1
      0111 F*      LOOP IN THE CALCULATION SPECIFICATIONS.    *              SAMPL1
      0112 F*                                                 *              SAMPL1
      0113 F*   4. USES KEYS FROM 000005 THROUGH 000500       *              SAMPL1
      0114 F*      IN INCREMENTS OF 5.                        *              SAMPL1
      0115 F*                                                 *              SAMPL1
      0116 F*   5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2     *              SAMPL1
      0117 F*      TO VERIFY THAT THE FILE WAS PROPERLY       *              SAMPL1
      0118 F*      LOADED.                                    *              SAMPL1
      0119 F*                                                 *              SAMPL1
      0120 F***************************************************              SAMPL1
0001  0121 F$SOURCE IP  F 256 256          DISK                              SAMPL1
0002  0122 FDISKOUT O   F 256 128 06AI    1 DISK                     01      SAMPL1
0003  0123 FPRINTER O   F  96  96          PRINTER                           SAMPL1

0004  0201 I$SOURCE NS  01                                                   SAMPL1
0005  0202 I                                      1    1 NODATA              SAMPL1

0006  0301 CL0                    Z-ADD0       COUNT   60                    SAMPL1
0007  0302 CL0                    Z-ADD0       RECNBR  30                    SAMPL1
0008  0303 CL0          REPEAT    TAG                                        SAMPL1
0009  0304 CL0          COUNT     ADD  5       COUNT                         SAMPL1
0010  0305 CL0          RECNBR    ADD  1       RECNBR                        SAMPL1
0011  0306 CL0          COUNT     COMP 505                  02               SAMPL1
0012  0307 CL0    N02             EXCPT                                      SAMPL1
0013  0308 CL0    N02             GOTO REPEAT                                SAMPL1
0014  03081CL0                    SETON                 LR                   SAMPL1
0015  0309 CLR          RECNBR    SUB  1       RECNBR                        SAMPL1

0016  0401 OPRINTER T  204    LR                                             SAMPL1
0017  0402 O                             20 'SAMPLE PROGRAM 1 HAS'           SAMPL1
0018  0403 O                             27 'LOADED'                         SAMPL1
0019  0404 O                   RECNBRZ   31                                  SAMPL1
0020  0405 O                             39 'RECORDS'                        SAMPL1
0021  0406 O                             61 'INTO AN INDEXED FILE.'          SAMPL1
0022  0408 O        T  2      LR                                             SAMPL1
0023  0409 O                             21 'KEYS ARE IN ASCENDING'          SAMPL1
0024  0410 O                             42 'SEQUENCE STARTING AT'           SAMPL1
0025  0411 O                             64 '000005 AND INCREASING'          SAMPL1
0026  0412 O                             84 'IN INCREMENTS OF 5.'            SAMPL1
0027  0413 O        T         LR                                             SAMPL1
0028  0414 O                             21 'SAMPLE PROGRAM 2 WILL'          SAMPL1
0029  0415 O                             44 'PRINT FROM THE INDEXED'         SAMPL1
0030  0416 O                             65 'FILE TO SHOW THAT IT'           SAMPL1
0031  0417 O                             86 'WAS PROPERLY LOADED.'           SAMPL1
0032  0501 ODISKOUT E         N02                                            SAMPL1
0033  0502 O                   COUNT      6                                  SAMPL1
0034  0503 O                             94 'RECORD NUMBER'                  SAMPL1
0035  0504 O                   RECNBR   128                                  SAMPL1
```

*This is the size that the link editor will attempt to put the object program into unless a different execute size is given on the header card (columns 13 and 14).

Figure H-6 (1 of 3). Model 15 Program Listing

```
INDICATORS USED
      LR LO 01 02

RG 305  INDICATORS UNREFERENCED
        01

RG 314 UNREFERENCED FIELD NAMES
 STMT#  NAME
 0005 NODATA

  FIELD NAMES USED
 STMT#  NAME   DEC  LGTH  DISP
 0006 COUNT    0    006   0005
 0007 RECNBR   0    003   0008

  LABELS USED
 STMT#  NAME   TYPE
 0008 REPEAT   TAG

  ERROR NUMBER  STATEMENT NUMBER
    RG 273          0032

  ERROR SEVERITY                                  TEXT
RG  273 W       OUTPUT INDICATORS IN COL 23-31 MISSING OR ALL NEGATIVE.
RG  305 W       INDICATOR ASSIGNED BUT NOT USED TO CONDITION OPERATIONS.
RG  314 W       FIELD, TABLE OR ARRAY NAME DEFINED BUT NEVER USED.
```

Displacement to right end of field from start of COMMON.

Overlay priority:
Low number = high priority
High number = low priority
Category 0 cannot be overlaid

```
        OVERLAY LINKAGE EDITOR CORE USAGE MAP     11/19/73

 START    CATEGORY   NAME AND       CODE LENGTH
ADDRESS               ENTRY    HEXADECIMAL DECIMAL

 4000                GLOBAL         07C8     1992
 47C8                COMMON         0009        9
 4800        0       SAMPL1         0100      256
 4900        0       $#RT02         0123      291
 4A23        0       $#IPCR         004F       79
 4A24                  $@0AC9
 4A72        0       $#OPCR         009C      156
 4A73                  $@0B18
 4B0E        0       $#CON0         00E9      233
 4BF7        0       $#CON1         0005        5
 4BFC        0       $#CON2         000E       14
 4C0A        2       $$CSIP         0027       39
 4C31        2       $$IOUT         005F       95
 4C90        2       $$SRBR         0082      130
 4D12        2       $$SRUA         0026       38
 4D38        2       $$SRTC         001C       28
 4D38                  DMSRLO
 4D49                  DMSRTC
 4D4C                  DMSRER
 4D54        2       $$SRBI         0105      261
 4E59        2       $$SRDF         001C       28
 4E75        2       $$SRMO         00A4      164
 4F19        2       $$SRSB         0046       70
 4F5F        2       $$SRDI         003E       62
 4F84                  DMSRPD
 4F7D                  DMSRRD
 4F9D        2       $$SRBP         002F       47
 4FCC        6       $$LPRT         00D1      209
 509D       93       $#OPEN         006D      109
 510A      126       $#INPT         0090      144
 515C                  $@0C25
 5163                  $@0C2C
 5190                  $@0C59
 5154                  $@0C1D
 5158                  $@0C21
 519A       28       $#IH01         0008        8
 51A2      126       $#TCAL         0052       82
 51F4       29       $#EXPT         002D       45
 5221       28       $#OH02         000C       12
 522D      126       $#IFLD         001D       29
 5246                  $@0CE8
 524A       93       $#CLOS         002B       43
 5251                  $@0EC2
 5262                  $@0ED3
 5266                  $@0ED7
 5275      107       $#LROT         0076      118
 52EB      126       $#TOUT         000B       11
 52F6      126       $#LROF         0024       36
 531A       71       $#OAOF         001D       29
 5337       28       $#OH03         000C       12
 5343      126       $#RCID         0046       70
 537A                  $@0C9A
 5389      126       $#CFLD         0026       38
 53AF       11       $$PGRI         0043       67
```

Contains buffers

Contains fields

Contains RPG work areas, and execution and pre-execution time DTTs.

Contains compile time tables, compile time DTTs and DTFs

Figure H-6 (2 of 3). Model 15 Program Listing

```
 START    CATEGORY  NAME AND       CODE LENGTH
ADDRESS              ENTRY     HEXADECIMAL DECIMAL

 53F2       107     $#LRC           0014       20

OL100 I   THE TOTAL CORE USED BY SAMPL1 IS  5126 DECIMAL.
OL101 I   THE START CONTROL ADDRESS OF THIS MODULE IS 4800.
OL104 I   TOTAL NUMBER OF LIBRARY SECTORS REQUIRED IS   13
          NAME-SAMPL1,PACK-SYSTEM,UNIT-R1,RETAIN-T,LIBRARY-O
```

Routine name. For a complete list of routine names see Figure H-7

Gives location of object program

Figure H-6 (3 of 3). Model 15 Program Listing

TITLE	MNEMONIC PROGRAM NAME
GLOBAL (contains buffers and IOBs)	
COMMON (contains fields) (see Note 1)	
ROCA (contains execution time and pre-execution time DTTs)	Name of program is printed as Program Name
IOCBs, DTFs, Compile Time Tables, compile time DTTs	$#RT02
Edit Code Assignments, Edit Code Patterns	$#MISC
Overlay Fetch Routine	OVLFRTN
File Translate Routine	$$PGAB
Detail Output	$#DOUT
Input Mainline	$#INPT
Total Calculations	$#TCAL
Total Output	$#TOUT
LR and Overflow Processing	$#LROF
Input Processing Control Routine (IPCR)	$#IPCR
Output Processing Control Routine (OPCR)	$#OPCR
Literals, Constants, Edit Patterns and Parameters	$#CON1-F
Detail Calculations	$#DCAL
Input Fields	$#IFLD
Record ID Routine	$#RCID
Multifile Logic	$#MFLG
Control Fields Processing	$#CFLD
Alternate Collating Sequence Routine	$$PGDI
Set Resulting Indicators Routine	$$PGRI
Array Index Routine	$$PGAA

Notes:

1. If COMMON is missing, $#RT02 and ROCA are combined.
2. Calculation subroutine names are assigned in ascending sequence with the first user defined subroutine having a category of 28. The first subroutine name may have any hexadecimal character assigned to it and gaps may exist between consecutive names.

TITLE	MNEMONIC PROGRAM NAME
Multiply Routine	$$PGMC
Input Hook (4F + file number)	$#IH01-19
Output Hook (69 + file number)	$#OH01-14
Exception Output Segment	$#EXPT
Calculation Subroutines (see Note 2)	$#SR01-FF
Move Fields (output fields for OR lines)	$#MF01-FF
LOKUP Subroutine	$$PGLC
Pack Routine	$$PGCO
Unpack Routine	$$PGCI
Divide Routine	$$PGIC
TESTZ Routine	$$PGTC
Convert to Binary Routine	$$PGBI
Convert to Decimal Routine	$$PGBO
Square Root Routine	$$PGAC
CHAIN Code Blocks	$#CHN1-F
DEBUG Routine	$$PGDC
RA File Process Routine	$#RAFL
Fetch Overflow Routine	$#FOVF
OA Overflow Routine	$#OAOF
OB Overflow Routine	$#OBOF
OC Overflow Routine	$#OCOF
OD Overflow Routine	$#ODOF
OE Overflow Routine	$#CEOF
OF Overflow Routine	$#OFOF
OG Overflow Routine	$#OGOF
OV Overflow Routine	$#OVOF
LR Output Segment	$#LROT
LR Calculations Segment	$#LRC
Open Mainline	$#OPEN
Close Mainline	$#CLOS
Load Object Tables	$$PGFI
Dump Object Tables	$$PGFO

Figure H-7. Model 15 Routine Names

This appendix contains the number of bytes of object code generated for RPG II operation codes. When used in conjunction with Appendix H, this information will help you determine the amount of main storage that may be saved by using certain coding practices.

For example, consider this main storage saving technique:

Try to use numeric fields of the same length and with the same number of decimal positions. If the fields cannot be the same length, try to have the number of decimal positions the same.

If the decimal position of Factor 1, Factor 2, and the Result Field are all different, an ADD operation will generate 27 bytes.

However, if all the fields were defined as having the same number of decimal positions, the same ADD operation would generate only 15 bytes.

Uniformity of fields will not only save main storage for ADD and SUB, but for most of the other arithmetic operations as well.

IBM International Business Machine Corporation — RPG CALCULATION SPECIFICATIONS — Form GX21-9093 Printed in U.S.A.

Line	Form Type	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Comments
01	C	* ASSUME FLD1 IS LENGTH 5, DECIMAL POSITIONS 2						
02	C	* ASSUME INPUT IS LENGTH 4, DECIMAL POSITION 1						
03	C	FLD1	ADD	INPUT	RFLD	9	3	27 BYTES
04	C							

IBM International Business Machine Corporation — RPG CALCULATION SPECIFICATIONS — Form GX21-9093 Printed in U.S.A.

Line	Form Type	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Comments
01	C	* ASSUME FLD1 AND INPUT ARE LENGTH 9, DECIMAL POSITIONS 3						
02	C	FLD1	ADD	INPUT	RFLD	9	3	15 BYTES
03	C							

The following abbreviations and symbols are used in discussing bytes used by calculation operations.

F1	–	Factor 1
F2	–	Factor 2
RF	–	Result Field
L1	–	Total length of Factor 1
L2	–	Total length of Factor 2
LR	–	Total length of Result Field
D1	–	Number of decimal positions in Factor 1
D2	–	Number of decimal positions in Factor 2
DR	–	Number of decimal positions in Result Field
H/A	–	Half adjust
=	–	equal
≠	–	not equal
–	–	minus
>	–	greater than
<	–	less than
+	–	plus

Operation	*Bytes*
SETON (each indicator set on)	3
SETOF (each indicator set off)	3
BITON	4
BITOF	4
TESTB	
test bit off	10
test bit mixed	17
test bit on	10
test bit off and mixed	23
test bit off and on	23
test bit mixed and on	23
test bit off, mixed, and on	29
SUB	
F1 = RF and D1 = D2 = DR	6
F1 ≠ RF and D1 = D2 = DR	15
F1 ≠ RF and D2 = DR	23
F1 ≠ RF and D2 = DR H/A	27
All other combinations	31
All other combinations H/A	39
Z-SUB	
D2 = DR	14
D2 ≠ DR	18
D2 ≠ DR H/A	22
TIME	
RF is a field or array with a constant index	15
RF is a variable indexed array	26
RF is a table	21

Operation	*Bytes*
ADD	
F1 = RF and D1 = D2 = DR	6
F2 = RF and D1 = D2 = DR	6
F1 ≠ F2 ≠ RF and D1 = D2 = DR	15
F1 = RF and D2 > DR	14
F2 = RF and D1 > DR	14
F1 = RF and D2 > DR H/A	18
F2 = RF and D1 > DR H/A	18
F1 = RF and D2 < DR H/A	18
F2 = RF and D1 < DR H/A	18
D1 = D2 < DR	23
All other combinations	27
All other combinations H/A	35
Z-ADD	
D2 = DR	6
D2 > DR	14
D2 > DR H/A	18
D2 < DR	18
COMP	
F1 and F2 are numeric and D1 = D2	10
F1 and F2 are numeric and D1 ≠ D2	18
F1 and F2 are alphameric and L1 = L2	6
F1 and F2 are alphameric and F1 is a field	22
F1 and F2 are alphameric and F1 is a table	26
alternate collating sequence (add these bytes to the appropriate compare operation listed previously)	10
TESTZ	
RF is a field	9
RF is a table	20
MULT	23
with H/A	27
DIV	
D1 - D2 = DR	23
D1 - D2 ≠ DR	27
D1 - D2 = DR + 1 H/A	31
D1 - D2 ≠ DR + 1 H/A	35
MVR	
D2 = DR	5
D2 ≠ DR	9
XFOOT	
D2 = DR	9
D2 ≠ DR	13
FORCE	13
with external indicator	13 + 7 = 20

Operation	*Bytes added to base*
CHAIN Base = 16	
with external indicator	6
when Factor 1 has a variable index	11
when key is not packed	14
when key is packed	23
when key is packed and Factor 1 is a table element	6
when key is a record number	8
when key is a record number and Factor 1 is a table element	6
when record not found indicator is given	12
when record not found indicator is not given	16

Operation	*Bytes added to base*
READ Base = 29	
with external indicator	6
with EOF indicator with BSCA	6
with EOF indicator without BSCA	12
with BSCA without EOF indicator	6
without BSCA without EOF indicator	19
with PAF limits	6
LOKUP Base = 15	
when Factor 1 is a table	6
when Factor 1 is a variable	11
with each resulting indicator	12
SETLL Base = 18	
when key is packed	12

MOVE, MOVEL, MHHZO, MHLZO, MLHZO, MLLZO — See the following table. The number of bytes specified includes all array control code lengths.

	MOVE alphameric/numeric	MOVEL LR<L2 numeric	MOVEL LR>L2 numeric	MOVEL LR=L2 alphameric/numeric	MOVEL LR<L2 alphameric	MOVEL LR>L2 alphameric	MLHZO	MHLZO	MHHZO	MLLZO
Field to Field	6	26	10	6	6	6	20	20	20	20
Array to Array	42	55	45	42	42	42	42	42	42	42
Field to Array	29	43	32	29	29	29	29	29	29	29
Table to Array	35	53	38	35	40	35	35	41	40	35
Array, Index to Array	40	66	43	40	52	40	40	52	52	40
Array to Array, Index	28	57	38	28	35	35	35	35	47	42
Field to Array, Index	17	34	27	17	17	24	24	31	24	31
Table to Array, Index	20	52	33	20	24	30	30	24	36	20
Array, Index to Table	20	46	27	20	30	24	24	24	36	20
Field to Table	9	23	16	9	9	13	13	9	13	9
Table to Table	15	41	22	15	19	19	19	19	25	15
Array, Index to Field	17	40	21	17	24	17	31	24	36	31
Table to Field	9	29	13	9	13	9	9	13	13	9

Operation	*Bytes*
DSPLY	
(for factor 1)	40
with variable index	51
with integer index	46
with alphameric field	73
with numeric field	74
(for result field)	10
with variable index	27
with integer index	16
with alphameric field	58
with numeric field	93
Conditioning indicators (does not apply to CHAIN, FORCE, LOKUP, and READ)	
each indicator	3
each AND type	3
Resulting indicators (does not apply to CHAIN, FORCE, LOKUP, and READ)	5
plus, for each resulting indicator	3
EXSR	4
GOTO	4
MOVEA	14
EXCPT	4
BEGSR	4
ENDSR	4

Array control code (initialization and processing) is generated for all calculations except DSPLY, LOKUP, CHAIN, READ, and FORCE.

Operation	*Bytes*
Array initialization	
F1 or F2 an array	6
F1 or F2 a table	4
F1 or F2 an array and tag	11
Array processing	
F1, F2, RF are arrays	30
F1-RF, F2-RF arrays	22
F2 and RF are arrays	16

Suppose, for example, that a SUB operation code was specified and has the following conditions:

1. F1 = RF

2. D1 = D2 = DR

3. F1, RF = full array

4. F2 = table

The length of object code generated would be as follows:

Array initialization	
F1 array	6 bytes
F2 table	4 bytes
RF array	6 bytes
SUB	6 bytes
Array processing	
F1-RF array	22 bytes

Thus, the total bytes of code generated for a SUB operation code is 44 bytes.

IN-LINE INQUIRY SUBROUTINE (SUBR95)

For Models 10 and 12 you can use SUBR95 to perform an inquiry type function without rolling your program out and rolling another program in (see *Control Card Specifications,* inquiry for a discussion of rollout/rollin). You can check at any point in your calculations to determine if an inquiry request has been made.

To use SUBR95 to check for an inquiry request, you must call SUBR95 by specifying the linkage shown in Figure J-1. The indicator specified in columns 45-46 can be an RPG II indicator. For a detailed discussion of this linkage, see *Appendix F. RPG II to Assembler Language Subroutine Linkage.*

When SUBR95 is called, it checks if an inquiry request has been made. If an inquiry request was made, the indicator specified in the RLABL operation is turned on and the inquiry request is reset. You can use this indicator to condition a GOTO or EXIT to another subroutine within your program. This subroutine can perform whatever function you wish.

You should not use SUBR95 in a B type program (one that performs normal rollout/rollin functions) as this can cause loss of interrupt requests.

Note: On the first call to SUBR95, the console typewriter is activated to accept an inquiry request.

PF KEY SUBROUTINE (SUBR89)

Model 15 RPG programs can use SUBR89 to allocate and test PF keys 1-9, on the keyboard. This allows you to control functions within an RPG program from an external source. SUBR89 can be used to perform inquiry type functions on the Model 15 in the same way SUBR95 is used for the Models 10 and 12; however, its use is not limited to that function.

To use SUBR89, you must call it using the linkage shown in Figure J-2. The first RLABL instruction passes a numeric field containing a value of 1-9. This specifies the PF key you want to allocate and test. You may allocate as many of the PF keys 1-9 that you want to but only two RLABL statements are allowed after each EXIT statement. Once a PF key is allocated to your program it remains allocated until your program terminates. The second RLABL instruction must specify any valid RPG II indicator. This indicator is turned on, by SUBR89, if the PF key specified in the first RLABL instruction is pressed.

The sign of the field in the RLABL instruction determines how SUBR89 will be used for that PF key.

If the field is a positive 1-9, on the first call to SUBR89, the PF key equal to the value in the field is allocated to the program. Subsequent calls to SUBR89 check if the PF key specified was pressed. If the PF key was pressed, the indicator specified in the second RLABL instruction is turned on and control is returned to the RPG program. If the PF key was not pressed the indicator is set off and control is returned to the RPG program.

If the field is a negative 1-9, on the first call to SUBR89, the PF key equal to the value in the field is allocated and SUBR89 waits until the PF key is pressed before turning on the indicator specified and returning control to the RPG program. Subsequent calls to SUBR89 check if the PF key specified was pressed. If the specified PF key hasn't been pressed, SUBR89 waits until the specified PF key is pressed before setting on the indicator and returning control to the RPG program.

In either case (field negative or positive), if the requested PF key cannot be allocated to the program a numeric zero (F0) is returned in the field specified in the first RLABL instruction. After the first call to SUBR89 you should check for a numeric zero (X'F0' is specified as a 0 in Factor 1 or Factor 2) in the field to determine if the requested PF key was allocated. A numeric zero (F0) is also returned if the field is not a digit 1-9. After control is returned to the RPG program it can use the indicator specified in the RLABL instruction to condition operations or program logic.

RPG CALCULATION SPECIFICATIONS

Punching Instruction	Graphic	Card Electro Number
	Punch	

Factor 1 (21–27)	Operation (28–32)	Factor 2 (33–42)	Result Field Name (43–48)	Length (49–51)	Decimal Positions (52)	Half Adjust (H) (53)	Plus / 1>2 / High (54–55)
	EXIT	SUBR95					
	RLABL		INXX				

Figure J-1. Linkage for SUBR95

RPG CALCULATION SPECIFICATIONS

Punching Instruction	Graphic	Card Electro Number
	Punch	

Factor 1 (21–27)	Operation (28–32)	Factor 2 (33–42)	Result Field Name (43–48)	Length (49–51)	Decimal Positions (52)	Half Adjust (H) (53)	Plus / 1>2 / High (54–55)
	EXIT	SUBR89					
	RLABL		FIELDA	(1)			
	RLABL		INXX	(2)			

(1) Field A is a one-position field containing a positive or negative 1-9.
(2) XX can be any valid RPG II indicator.

Figure J-2. Linkage to SUBR89

Index

SC21-7504-6

IBM System/3 RPG II Reference Manual (File No. S3-28) Printed in U.S.A. SC21-7504-6

IBM®

International Business Machines Corporation
General Systems Division
5775D Glenridge Drive N.E.
Atlanta, Georgia 30301
(USA Only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

Please use this form only to identify publication errors or request changes to publications. Technical questions about IBM systems, changes in IBM programming support, requests for additional publications, etc, should be directed to your IBM representative or to the IBM branch office nearest your location.

Error in publication (typographical, illustration, and so on). **No reply.**

Page Number *Error*

Inaccurate or misleading information in this publication. Please tell us about it by using this postage-paid form. We will correct or clarify the publication, or tell you why a change is not being made, provided you include your name and address.

Page Number *Comment*

☐ Check if reply is requested.

Note: All comments and suggestions become the property of IBM.

Name ______________________________

Address ______________________________

____

● No postage necessary if mailed in the U.S.A.

IBM System/3 RPG II
Reference Manual

SC21-7504-6

SC21-7504-6

Fold

Fold

FIRST CLASS
PERMIT NO. 387
ROCHESTER, MINN.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
General Systems Division
Development Laboratory
Publications, Dept. 245
Rochester, Minnesota 55901

Fold

Fold

IBM®

International Business Machines Corporation
General Systems Division
5775D Glenridge Drive N.E.
Atlanta, Georgia 30301
(USA Only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)

Please use this form only to identify publication errors or request changes to publications. Technical questions about IBM systems, changes in IBM programming support, requests for additional publications, etc, should be directed to your IBM representative or to the IBM branch office nearest your location.

Error in publication (typographical, illustration, and so on). **No reply.**

Page Number *Error*

Inaccurate or misleading information in this publication. Please tell us about it by using this postage-paid form. We will correct or clarify the publication, or tell you why a change is not being made, provided you include your name and address.

Page Number *Comment*

☐ Check if reply is requested.

Note: All comments and suggestions become the property of IBM.

Name ______________________________

Address ______________________________

____

● No postage necessary if mailed in the U.S.A.

IBM System/3 RPG II
Reference Manual

SC21-7504-6

Fold Fold

FIRST CLASS
PERMIT NO. 387
ROCHESTER, MINN.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY . . .

IBM Corporation
General Systems Division
Development Laboratory
Publications, Dept. 245
Rochester, Minnesota 55901

Fold Fold

IBM®

International Business Machines Corporation
General Systems Division
5775D Glenridge Drive N.E.
Atlanta, Georgia 30301
(USA Only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)