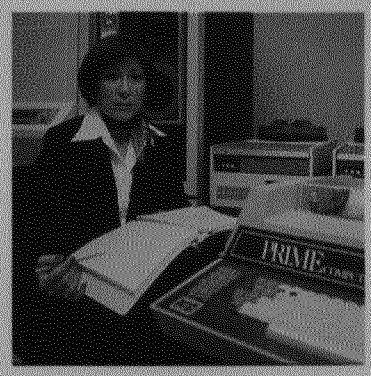
Prime Computer, Inc.
Programmer's Guide PDR3031

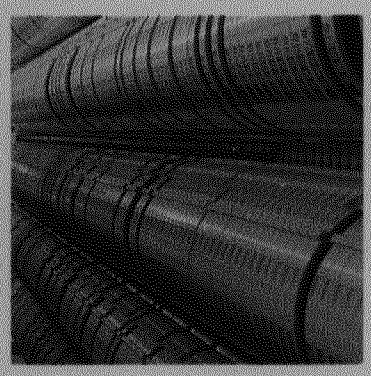
PRIME

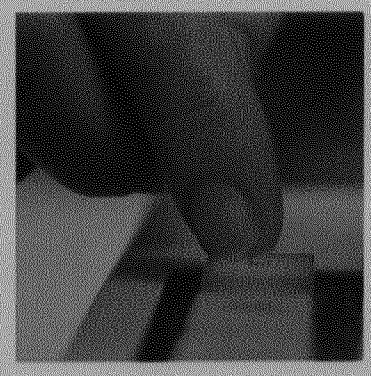
RPG II












PRIME'S RPG II PROGRAMMER'S GUIDE

This guide documents Prime RPG II and all supporting PRIMOS operating system features as implemented at Master Disk Revision Level 16. Information in the Rev. 14 and Rev. 15 PTUs have been incorporated into this guide as well as corrections, and additions to the Rev. 13 IDR.

This guide is organized to make life easier for you, the RPG II application programmer.

We assume you know RPG II, and will easily adapt to Prime's implementation and extensions, which are fully defined in the reference sections of this guide.

PRIMOS, on the other hand, is a large and versatile operating system. It is no small task to sift through all the reference documentation for PRIMOS and its file system, libraries, utilities, and supporting software to find what you need to get a RPG II application running.

To save you trouble, we've done all that for you in the early sections of this guide, by:

- Selecting the PRIMOS capabilities that are of key importance to the RPG II programmer.
- Presenting these capabilities in the usual order of RPG II program development.
- Including all the details on the essential tools.
- Summarizing optional, convenience, and advanced features.
- Leaving out what is irrelevant.

The result is a single document containing everything you need to know to write, modify, compile, load, execute, and debug most RPG II application programs.

In exceptional cases, you may need to refer to supporting reference documents. For example, this guide gives enough information on Prime's MIDAS subsystem for you to use it for most RPG applications. To develop applications using MIDAS and FORMS subsystems, however, you need access to the complete details in the reference documents.

We hope you will find this to be a helpful guide to the particulars of RPG II programming within the PRIMOS operating system. We invite comments on the organization and philosophy of this guide, as well as its contents, accuracy, and clarity.

All correspondence on suggested changes to this document should be directed to:

Anthony R. Lewis, Technical Writer
Technical Publications Department
Prime Computer, Inc.
145 Pennsylvania Avenue
Framingham, Massachusetts 01701

Acknowledgements:

We wish to thank the members of the RPG II PROGRAMMER'S GUIDE team and also the non-team members, both customer and Prime, who contributed to and reviewed this PDR.

First Printing February 1979

CONTENTS

PART I - OVERVIEW

PART II - USING RPG II UNDER PRIMOS

PART III - ADVANCED TOPICS

PART IV - REFERENCE

APPENDICES

ILLUSTRATIONS

TABLES

SECTION 1

INTRODUCTION

PRIME RPG II

This guide describes the capabilities available in Prime RPG II. The language is industry-standard, designed for interactive data processing, and highly compatible with IBM System/3 Model 1Ø RPG II. Prime's RPG II operates on Prime 3ØØ and higher systems. At Revision level 16, the compiler and interpreter run in 64R mode.

Under PRIMOS, RPG II is concurrently available to as many as 63 interactive users. It is supported by the extensive editing, file handling, input/output, and job control capabilities of PRIMOS. This environment obviates the need for extensive input/output supports internal to RPG logic. Prime RPG II is a disk to disk operation. Optimal software and hardware capability is directed toward data manipulation, calculation, and output file construction during program execution. Figure 1-1 presents a general flowchart of Prime RPG II program cycle logic.

In the total Prime environment, a variety of input/output methods can be managed. For example, reports can be spooled to the common system line printer; card or magnetic tape output files can be read and converted to disk. Such operations can be independent of, but concurrent with, RPG program execution for maximum efficiency in typical business applications.

SCOPE OF THIS DOCUMENT

This document is a comprehensive guide for the Prime RPG II programmer. It contains everything normally necessary for writing, compiling, loading, and executing RPG II programs. The user is assumed to be familiar with the RPG II language but not with its implementation and use on a Prime computer. Users unfamiliar with the language should read one of the commercially available instruction books, such as:

> IBM SYSTEM/3 RPG II REFERENCE MANUAL SC21-75Ø4-6 (File No.S3-28)
> available from: IBM Corporation, Publications, Dept. 245,
> Rochester, Minnesota 559Ø1, U.S.A.

This Preliminary Documentation Release documents the operation of RPGII under PRIMOS at software revision level 16. It is more complete than the previous documentation, especially in the area of utilities supporting the RPGII language. This guide replaces the following documents:

> PRIME RPG II, IDR3Ø31
> RPG II (Rev.14), PTU44
> RPG II (Rev.15), PTU49

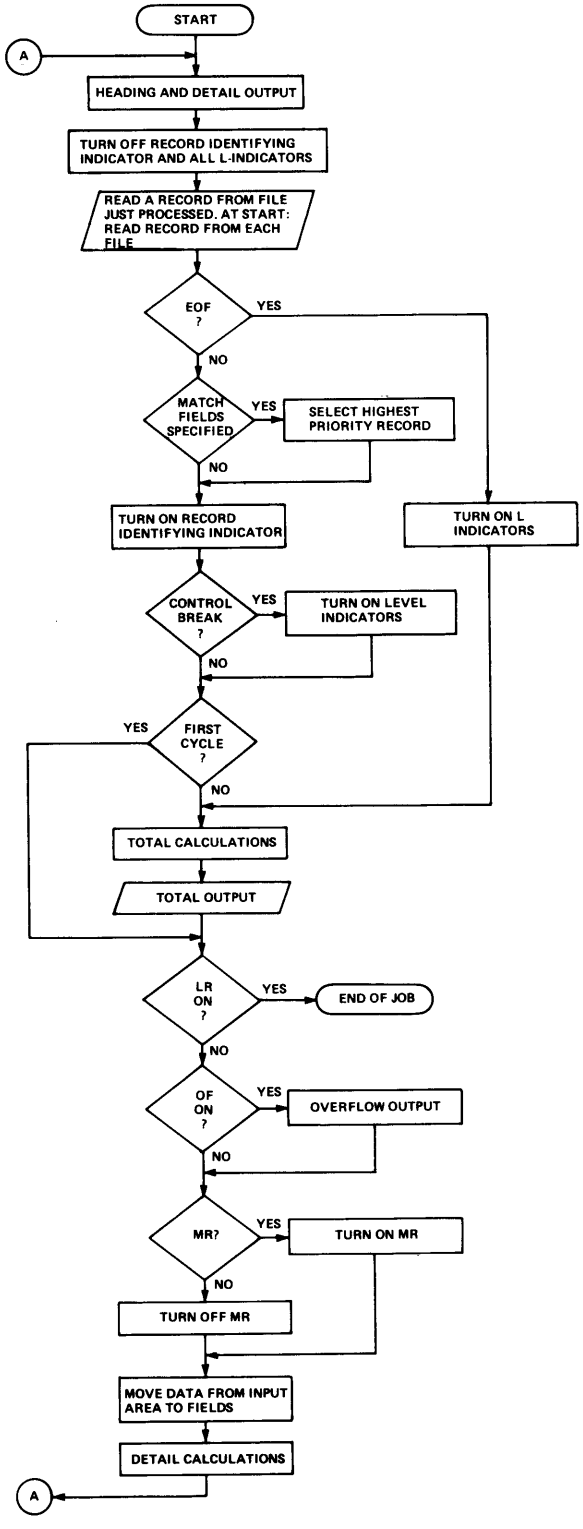


Figure 1-1. Prime RPG Logic Flow

This document is not yet in its final form. Certain sections, less central to the major purposes of this guide, have not been included. They will be added to the next version of this guide, which will be a Final Documentation Release.

Organization

The guide is composed of four major parts:

Part 1. An introductory section including basic concepts and conventions used in this guide (Section 1). This is followed by a section on considerations to be taken into account when converting existing RPG II programs to run on a Prime computer (Section 2).

Part 2. Using the Prime computer for RPG II programming. This is a tutorial, arranged to follow the normal sequence of program development. A single pass through this part will enable the user to perform all the usual RPG II programming functions. The order of information presented is:

Accessing the system (Section 3).

Creating a program (Section 4).

Compiling (Section 5).

Loading (Section 6).

Executing the program (Section 7).

System utilities are introduced and all concepts and PRIMOS-level commands necessary for the large majority of users are discussed with examples. A user wishing to go beyond these for special programming concepts, etc. will find references to the information (either in this document or another reference document) at the appropriate place. In most cases, it is unnecessary to use any document other than this one.

Part 3. Advanced topics. Section 8 covers necessary details of RPG's interface to Prime's Multiple Index Data Access System (MIDAS). MIDAS is used in RPG for non-sequential files.

Part 4. Reference. Section 9 forms a convenient reference to the coding specifications for sheets H, F, E, L, I, C, and O.

Section 1Ø contains complete details on the Compiler and A-register settings. Section 11 gives details on packed decimal and binary files and their usage.

Appendices:

A. Sample programs

B. Sample program development

C. ASCII character set

D. Summary of differences between Prime and IBM System/3 RPG

E. Compiler messages

F. Run-time messages

Related Documents

The following documents contain detailed reference information useful to the RPG II programmer.

PRIME RPG II Debugging Template

Reference Guide, PRIMOS Commands
Subroutine Reference Guide
New User's Guide to EDITOR and RUNOFF
Reference Guide, MIDAS
FORMS Programmer's Guide

GLOSSARY OF PRIME CONCEPTS AND CONVENTIONS

The following is a glossary of basic concepts and conventions of Prime computers, the PRIMOS operating system, and the file system.

abbreviation of PRIMOS commands

Only internal PRIMOS commands may be abbreviated.

binary file

A translation of source file generated by a language translator (PMA, COBOL, FTN, RPG). Such files are in the format required as input to the loaders. Also called "object file".

byte

8 bits; 1 ASCII character.

CPU

Central Processor Unit (the Prime computer proper as distinct from peripheral devices or main memory).

current directory

A temporary working directory explained in the discussion on Home vs Current Directory later in this section.

directory

A file directory; a special kind of file containing a list of files and/or other directories, along with information on their characteristics and location. MFDs, UFDs, and subdirectories (sub-UFDs) are all directories. (Also see segment directory.)

directory name

The file name of a directory.

external command

A PRIMOS command existing as a runfile in the command directory (CMDNCØ). It is invoked by name, and executes in user address space. External commands print GO when starting, and cannot be abbreviated.

file

An organized collection of information stored on a disk (or a peripheral storage medium such as tape). Each file has an identifying label called a filename.

filename

A sequence of 32 or fewer characters which names a file or a directory. Within any directory, each filename is unique. Directory names and a filename may be combined into a pathname. Most commands accept a pathname wherever a filename is required.

Filenames may contain only the following characters:

A-Z, Ø-9, _ # $ - . * &

The first character of a filename must not be numeric. On some devices underscore (_) prints as backarrow (<-).

filename conventions

Prefixes indicate various types of files. These conventions are established by the compilers and loaders, or by common use, and not by PRIMOS itself.

B_filename	Binary (Object) file
C_filename	Command input file
L_filename	Listing file
M_filename	Load map file
O_filename	Command output file
filename	Source file or text file
*filename	SAVED (Executable) R-mode runfile
#filename	SAVED (Executable) V-mode runfile

file-unit

A number between 1 and 63 ('77) assigned as a pseudonym to each open file by PRIMOS. This number may be given in place of a filename in certain commands, such as CLOSE. PRIMOS-level internal commands require octal values. Each user may have up to 16 file units open at the same time. Certain commands or activities use particular unit numbers by default:

PRIMOS assigned units	Octal	Decimal
INPUT, SLIST	1	1
LISTING	2	2
BINARY	3	3
AVAIL	5	5
COMINPUT	6	6
SEG's loadmap	13	11
COMOUTPUT	77	63
EDITOR	1,2	1,2
SORT	1-4	1-4
RUNOFF	1-3	1-3

file protection keys

See keys, file protection.

home directory

The user's main working directory, initially the login directory. A different directory may be selected with the ATTACH command. See the discussion on Home vs Current Directory later in this section.

identity

The addressing mode plus its associated repertoire of computer instructions. Programs compiled in 32R or 64R mode execute in the R-identity; programs compiled in 64V mode execute in the V-identity. R-identity and V-identity are also called R-mode and V-mode.

internal command

A command that executes in PRIMOS address space. Does not overwrite the user memory image. Internal commands can be abbreviated. See "abbreviation of PRIMOS commands".

keys, file protection

Specify file protection, as in the PROTEC command.

Ø	No access
1	Read
2	Write
3	Read/Write
4	Delete and truncate
5	Delete, truncate and read
6	Delete, truncate and write
7	All rights

LDEV

Logical disk device number as printed by the command STATUS DISKS. (See ldisk.)

ldisk

A parameter to be replaced by the logical unit number (octal) of a disk volume. It is determined when the disk is brought up by a STARTUP or ADDISK command. Printed as LDEV by STATUS DISKS.

logical disk

A disk volume that has been assigned a logical disk number by the operator or during system startup.

MFD

The Master File Directory. A special directory that contains the names of the UFDs on a particular disk or partition. There is one MFD for each logical disk.

mode

An addressing scheme. The mode used determines the construction of the computer instructions by a compiler or assembler. (See identity.)

nodename

Name of system on a network; assigned when local PRIMOS system is built or configured.

number representations

xxxxx	Decimal
'xxxxx	Octal
$xxxxx	Hexadecimal

object file

See binary file

open

Active state of a file-unit. A command or program opens a file-unit in order to read or write it.

output stream

Output from the computer that would usually be printed at a terminal during command execution, but which is written to a file if COMOUTPUT command was given.

packname

See volume-name.

page

A block of 1024 16-bit words within a segment (512 words on Prime 300).

partition

A portion [or all] of a multihead disk pack. Each partition is treated by PRIMOS as a separate physical device. Partitions are an integral number of heads in size, offset an even number of heads from the first head. A volume occupies a partition, and a "partition of a disk" and a "volume of files" are actually the same thing.

pathname

A multi-part name which uniquely specifies a particular file (or directory) within a file system tree. A pathname (also called treename) gives a path from the disk volume, through directory and subdirectories, to a particular file or directory. See the discussion on Pathnames in this section.

PDEV

Physical disk unit number as printed by STATUS DISKS. (See pdisk.)

pdisk

A parameter to be replaced by a physical disk unit number. Needed only for operator commands.

phantom user

A process running independently of a terminal, under the control of a command file.

runfile

Executable version of a program, consisting of the loaded binary file, subroutines and library entries used by the program, COMMON areas, initial settings, etc. (Created using LOAD or SEG.)

SEG

Prime's segmentation utility.

segment

A 65,536-word block of address space.

segment directory

A special form of directory used in direct-access file operations. Not to be confused with directory, which means "file directory".

segno

Segment number.

source file

A file containing programming language statements in the format required by the appropriate compiler or assembler.

subdirectory

A directory that is in a UFD or another subdirectory.

sub-UFD

Same as subdirectory.

treename

A synonym for pathname.

UFD

A User File Directory, one of the Directories listed in the MFD of a volume. It may be used as a LOGIN name.

unit

See file-unit.

volume

A self-sufficient unit of disk storage, including an MFD, a disk record availability table, and associated files and directories. A volume may occupy a complete disk pack or be a partition within a multi-head disk pack.

volume-name

A sequence of 6 or fewer characters labeling a volume. The name is assigned during formatting (by MAKE). The STATUS DISKS command uses this name in its DISK column to identify the disk.

word

As a unit of address space, two bytes or 16 bits.

Conventions

The conventions for PRIMOS command documentation are:

WORDS-IN-UPPER-CASE

Capital letters identify command words or keywords. They are to be entered literally. If a portion of an upper-case word is underlined, the underlined letters indicate the minimum legal abbreviation.

Words-in-lower-case

Lower case letters identify parameters. The user substitutes an appropriate numerical or text value.

Braces { }

Braces indicate a choice of parameters and/or keywords. Unless the braces are enclosed by brackets, at least one choice must be selected.

Brackets [ ]

Brackets indicate that the word or parameter enclosed is optional.

Hyphen -

A hyphen identifies a command line option, as in: SPOOL -LIST

Parentheses ( )

When parentheses appear in a command format, they must be included literally.

Ellipsis ...

The preceding parameter may be repeated.

Angle brackets < >

Used literally to separate the elements of a pathname. For example: <FOREST>BEECH>BRANCH537>TWIG43>LEAF4.

option

The word option indicates one or more keywords or parameters can be given, and that a list of options for the particular command follows.

Spaces

Command words, arguments and parameters are separated in command lines by one or more spaces. In order to contain a literal space, a parameter must be enclosed in single quotes. For example, a pathname may contain a directory having a password:

'<FOREST>BEECH SECRET>BRANCH6'.

The quotes ensure that the pathname is not interpreted as two items separated by a space.

User input usually may be either in lower case or in UPPER CASE. The rare exceptions will be specified in the commands where they occur.

SPECIAL TERMINAL KEYS

CONTROL

The key labeled CONTROL (or CTRL) changes the meaning of alphabetic keys. Holding down CONTROL while pressing an alphabetic key generates a control character. Control characters do not print. Some of them have special meanings to the computer. (See CONTROL-P, CONTROL-Q and CONTROL-S, below.) Others are ignored.

RUBOUT

The key labeled RUBOUT has a special use in RUNOFF. It is not generally meaningful to other standard Prime software. On some terminals it is labeled DELETE or DEL.

RETURN

The RETURN key ends a line. PRIMOS edits the line according to any erase (") or kill (?) characters, and either processes the line as a PRIMOS command, or passes it to a utility such as the editor. RETURN is also called CR or CARRIAGE-RETURN.

BREAK
ATTN
INTRPT

See CONTROL-P.

Special Characters

Caret (^)

Used in EDITOR to enter octal numbers and for literal insertion of Erase and Kill characters. On some terminals and printers, prints as up-arrow (↑).

Backslash (\)

Default EDITOR tab character.

Double-quote (")

Default erase character for PRIMOS, EDITOR, and RUNOFF Command Mode. Each double-quote erases a character from the current line. Erasure is from right (the most recent character) to left. Two double-quotes erase two characters, three erase three, and so forth. You cannot erase beyond the beginning of a line. The PRIMOS command TERM (Section 1Ø of this guide) allows the user to choose a different erase character.

Question mark (?)

Default kill character for PRIMOS, EDITOR, and RUNOFF Command Mode. Each question mark deletes all previous characters on the line. The PRIMOS command TERM (Section 1Ø of this guide) allows the user to choose a different kill character.

CONTROL-P

QUIT immediately (interrupt/terminate) from execution of current command and return to PRIMOS level. Echoes as QUIT. Used to escape from undesired processes. Will leave used files open in certain circumstances. Equivalent to hitting BREAK key.

CONTROL-S

Halt output to terminal, for inspection. No commands other than CONTROL-P (QUIT) or CONTROL-Q (Continue) may be given. This special function is activated by the command TERM -XOFF.

CONTROL-Q

Continue output to terminal following a CONTROL-S (if TERM -XOFF is in effect).

UNDERSCORE (_)

On some devices, prints as a backarrow (<-).

SYSTEM PROMPTS

The OK Prompt

The OK prompt indicates that the most recent command to PRIMOS has been successfully executed, and that PRIMOS is ready to accept another command from the user. The punctuation mark following the "OK" indicates to the user whether he is interfacing with a single-user level of PRIMOS. The prompt "OK:" indicates single-user PRIMOS (a version of PRIMOS II); the prompt "OK," indicates multi-user PRIMOS (PRIMOS III, IV or V).

PRIMOS III, IV, and V support type-ahead. The user need not wait for the "OK," after one command before beginning to type the next command. However, since each character echoes as the user types it, output from the previous command may appear on the terminal to be jumbled with the command being typed ahead. Type ahead is limited to 192 characters.

PRIMOS II does not support type-ahead. The user must wait for "OK:" before beginning to enter the next command.

The ER! Prompt

The ER! prompt indicates that PRIMOS was unable to execute the most recent command, for one reason or another, and that PRIMOS is ready to accept another command from the user. The ER! prompt usually is preceded by one or more error messages indicating what PRIMOS thought the trouble was.

Common errors include:

- Typographical errors
- Omitting a password
- Being in the wrong directory
- Forgetting a parameter or argument

USING THE FILE SYSTEM

File and Directory Structures

A PRIMOS file is an organized collection of information identified by a filename. The file contents may represent a source program, an object program, a run-time memory image, a set of data, a program listing, text of an on-line document, or anything the user can define and express in the available symbols.

Files are normally stored on the disks attached to the computer system. No detailed knowledge of the physical location of a file is required because the user, through PRIMOS commands, refers to files by name. On some systems, files may also be stored on magnetic tape for backup or for archiving.

PRIMOS maintains a separate user file directory (UFD) for each user to avoid conflicts that might arise in assignment of filenames. A master file directory (MFD) is maintained by PRIMOS for each logical disk connected to the system. The MFD contains information about the location of each User File Directory (UFD) on the disk. In turn, each UFD contains information about the location and content of each file or sub-UFD in that directory.

The types of files most often encountered are shown in Table 1-1. For a description of the PRIMOS file system and a description of the ordering of information within files, refer to the Reference Guide, PRIMOS Subroutines.

Pathnames

The PRIMOS file directory system is arranged as a tree. At the root are the disk volumes (also called partitions, or logical disks). Each disk volume has a Master File Directory (MFD) containing the names of several User File Directories (UFDs). Each UFD may contain not only files, but subdirectories (sub-UFDs), and they may contain subdirectories as well. Directories may have subdirectories to any reasonable level.

A pathname (also called a treename) is a name used to specify uniquely any particular file or directory within PRIMOS. It consists of the names of the disk volume, the UFD, a chain of subdirectories, and the target file or directory. For example,

```
<FOREST>BEECH>BRANCH5>SQUIRREL
```

specifies a file on the disk volume FOREST, under the UFD BEECH and the sub-UFD BRANCH5. The file's name is SQUIRREL. Figure 1-1 illustrates how pathnames show paths through a tree of directories and files.

Disk volume names, and the associated logical disk numbers, may be found with the STATUS DISKS command, described later. A pathname can be made with the logical disk number, instead of the disk volume name. For example, if FOREST is mounted as logical disk 3,

```
<3>BEECH>BRANCH5>SQUIRREL
```

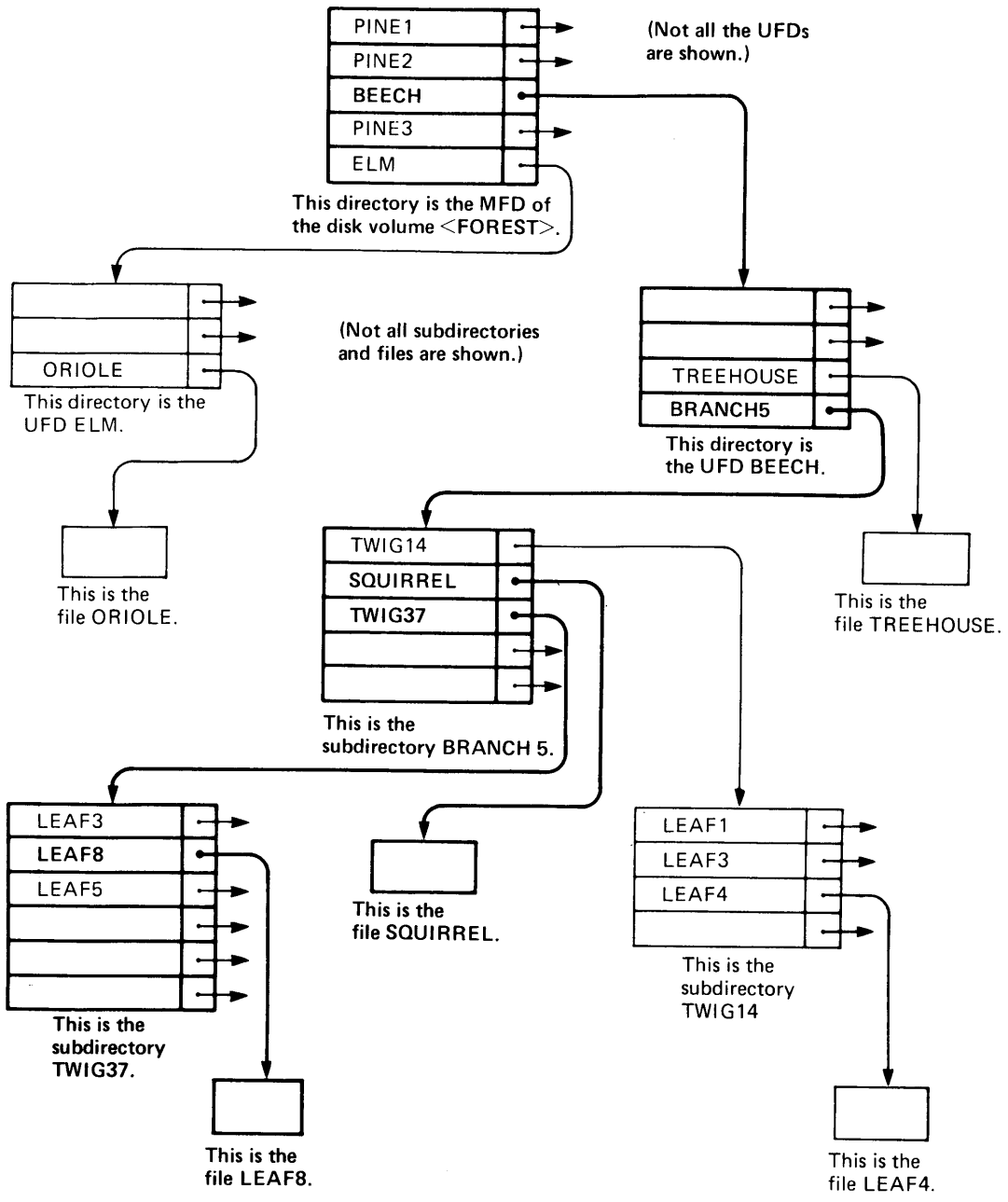
specifies the same file as the previous example.

Usually each UFD name is unique throughout all the logical disks. In our example that would mean that there would be only one UFD named BEECH in all the logical disks, Ø through 17. When that is the case, the volume or logical disk name may be omitted, and PRIMOS will search all the logical disks, starting from Ø, until the UFD is found. For example, if there is no UFD named BEECH on disks Ø, 1, or 2, then

```
BEECH>BRANCH5>SQUIRREL
```

Table 1-1.
Types of Files in PRIMOS

File Type	How Created	How Accessed	How Deleted	Use
ASCII, uncompressed	Programs SORT COMOUTPUT	Programs ED (examine only) SLIST SPOOL FTN READ/WRITE	DELETE FUTIL DELETE	Source files, text, data records for sequential access
ASCII, Compressed	CMPRES Some COBOL programs, ED	EXPAND to ASCII SPOOLer with EXPAND option ED	DELETE FUTIL DELETE	Same as compressed ASCII
Object (Binary)	Translators:RPG, FTN, PMA, COBOL, Binary Editor	LOAD or SEG Binary Editor (EDB)	DELETE FUTIL DELETE Binary Editor	Input to SEG or LOAD, Libraries
Saved Memory Image	LOAD Applications programs	TAP, PSD Control panel	DELETE FUTIL DELETE	Runfiles
Segmented runfile	SEG	SEG, VPSD Control panel	SEG DELETE FUTIL TREDEL	Runfiles
Segmented data file	SGDR$$ subroutine MIDAS, DBMS	SGDR$$ subroutine MIDAS DBMS	FUTIL TREDEL MIDAS KIDDEL	Data records for direct access
UFD Sub-UFD	CREATE	Contents: LISTF	DELETE FUTIL DELETE FUTIL TREDEL FUTIL UFDDEL	Used by PRIMOS
MFD	MAKE	Contents: LISTF	NO	Used by PRIMOS
Disk record availability table DSKRAT file	MAKE	NO	NO	Used by PRIMOS
BOOT	MAKE	NO	NO	Used by PRIMOS
CMDNCØ	MAKE	Contents: LISTF	NO	Used by PRIMOS



Pathnames identify files in a tree-structured file system. The pathnames:
<FOREST>BEECH>BRANCH5>SQUIRREL
<FOREST>BEECH>BRANCH5>TWIG37>LEAF8
are illustrated.

Figure 1-2. Examples of Files and Directories in PRIMOS Tree-structured File System

will specify the same file as the previous two examples. This last form of pathname, in which the disk specifier is omitted, is called an ordinary pathname because it is very frequently used.

Pathnames vs Filenames

Most commands accept a pathname to specify a file or a directory. So the terms "filename" and "pathname" may be used almost interchangeably. A few commands, however, require a filename, not a pathname. It is easy to tell a filename from a pathname. A pathname always contains a ">", while a filename or directory name never does.

Home vs Current Directories

PRIMOS has the ability to remember two working directories for each user: the "home" directory, and the "current" directory. With few exceptions, the home and current directories are the same. All work can be accomplished while treating them both under the single concept of "working directory".

When the user logs in to a UFD, that UFD becomes the working directory. The ATTACH command changes the working directory to any other directory to which the user has access rights. A working directory may be an MFD, UFD, or sub-UFD.

The ATTACH command has a home-key option which allows the current directory to change while the home directory remains the same. See Reference Guide, PRIMOS Commands for details of this operation.

Relative pathnames

It is often more convenient to specify a file or directory pathname relative to the home directory, rather than via a UFD. For example, when the home directory is

```
BEECH>BRANCH5
```

the commands

```
OK, SLIST BEECH>BRANCH5>TWIG9>LEAF3
```

and

```
OK, SLIST *>TWIG9>LEAF3
```

have the same meaning. The symbol "*" as the first directory in a pathname means "home directory".

Current disk

Occasionally it will be necessary to specify a UFD on the disk volume you are currently using; that is, where your home directory is. For example, when developing a new disk volume with UFD names identical to those on another disk, it is necessary to carefully specify which disk is to be used, each time a pathname is given. The current disk is specified by

```
<*>BEECH>BRANCH5
```

for example. Do not confuse "<*>", meaning current disk, with the asterisk alone, which means home directory.

SECTION 2

CONVERSION CONSIDERATIONS

CHARACTER SET AND FILE FORMAT

The standard character set used by Prime is the ANSI, ASCII, 7-bit set shown in Appendix C. Standard RPG II specification sheets are supported; column by column coding details are summarized in Section 9. The source program order of the seven standard RPG specifications utilized by Prime is:

1. Header Specifications
2. File Description Specifications
3. Extension Specifications
4. Line Counter Specifications
5. Input Specifications
6. Calculation Specifications
7. Output Specifications

Tables or arrays included at compile time should follow Output Specifications. The RPG compilation is controlled by command line options (see Section 5).

If position 21 in the Header Specification is blank (denoting United States format), the entire Header Specification may be omitted.

The system will accept sequential, keyed index or direct access files. File conversion, where required, may be aided by Pulse library programs. This is a group of user programs available through Prime. For more information contact:

User Group Co-ordinator
Prime Computer, Inc.
Post Office Box 2600
Framingham, Massachusetts 01701

The new user may be particularly interested in the MAGNET utility, which will convert fixed-length, EBCDIC tape files to uncompressed ASCII disk files. (See Section 4 of this document and Reference Guide, PRIMOS Commands for details of MAGNET usage).

PROGRAMS

Source program specifications and sequence are essentially compatible with IBM's System/3 RPG II. (See the column by column coding review presented in Section 9 of this document.) Where incorrect coding does not affect the integrity of Prime RPG II, the system will accept and ignore such coding, or convert it at execution time. Both error and warning messages are printed by the compiler at compile-time.

Note

During conversion, warning messages will not inhibit execution in most instances. Error message indications, however, must be corrected. Accuracy in old or new source programs will maximize compilation time and efficiency.

The user should be aware that certain situations require special attention during conversion:

- A 1P indicator in column 41 of the Control Specification is ignored by Prime RPG. Repeated printing for forms positioning is not available. The system will accept the indicator without flagging it. Forms positioning is not currently implemented as Prime systems are oriented towards queuing of user print requests. Forms positioning may be simulated by using halt indicators in the program.

- An overflow indicator should be assigned to a listing file; the form will not automatically advance to a new page through default. In addition to the overflow indicator, a skip-to-a-new-page must be specified or the listing will be continuous. This is usually done on the first heading line. If there is no indicator assignment, a spooler-generated title line and spooler pagination are printed at the top of each page.

- EBCDIC fields are flagged and treated as unpacked ASCII. The results of operations performed on such fields will be inaccurate. The MOVE operations are also affected.

- Sequential RPG Update files must be in uncompressed ASCII format for accurate program execution. Physical space requirements differ for compressed and uncompressed ASCII, and the writing of disk Update files is an overlaying process. All output written via Prime RPG II will be in uncompressed ASCII; therefore, all input should be in uncompressed ASCII to avoid record mutilation.

Note

For files other than Update files, output records are not written in the same physical location as input records, so that record size variance is not a factor. For non-Update files, Prime RPG II will read and convert compressed ASCII with no record loss.

- The DEBUG operation is controlled through A-register setting at run-time, rather than through control card entry. See Table 7-1, A-register Bit Definitions Governing External User Indicators.

- Files are referenced by PRIMOS through unique filenames. If a file enters a Prime system with a duplicate name, the original is deleted. For this reason, if RPG program output files are to be preserved, the user should issue a change name system command after program execution (See Section 4).

- If the display operation, DSPLY, is used, communication is possible between an RPG II program and the user at program run-time. First the prompt 'DSPLY' is output to the user terminal, then one or two values or character strings. The user entry format to change values or character strings consists of:

xxxxxxx...xxxxxx (CR)	Alpha item up to 256 characters.
ddddddddddddddd (CR)	Numeric item up to 15 characters.
ddddddddddddddd-(CR)	Negative numeric item (the length of the field need not accommodate the minus sign.)
?(CR)	Set item to zero or blank.
(CR)	Make no change to current value.

Any time an error on input is discovered by the operator, it may be corrected by entering a ? character which causes all the previous characters to be ignored, or by entering a " character which will delete the previous character.

Leading zeros need not be entered on numeric items. Trailing blanks need not be entered on alpha items. Decimal points should not be included in numeric items.

Character strings that are larger than the item's field size are truncated on the right. Numeric values containing more digits than the item's field size are truncated on the left.

- A complete list of IBM RPG II features not supported in Prime RPG II at Rev. 16 is in Appendix D.

SECTION 3

ACCESSING PRIMOS

INTRODUCTION

Purpose of this Section

This section is a brief overview of some of the fundamental features of the PRIMOS operating system for the RPGII programmer. It assumes that you are a RPGII programmer who has previous experience with an interactive computer system, although possibly not on a Prime computer. If you are not familiar with interactive computers, you may prefer to start with the New User's Guide to EDITOR and RUNOFF. This section also assumes you have read the concepts and definitions in Section 1.

Using the PRIMOS Programmer's Companion

In this section we introduce the essential PRIMOS commands so that you can begin working on the system. We recommend that you keep a Programmer's Companion handy as a summary of the commands explained in this section plus other PRIMOS commands. In this user's guide we have selected only those PRIMOS commands we know will be of use to the RPGII programmer. Depending upon your application, there are many other PRIMOS commands that may simplify your task or increase efficiency.

Using PRIMOS

PRIMOS recognizes more than one hundred commands, some of which invoke subsystems which themselves respond to subcommands or extensive dialogs. However, most RPGII users can do 99% of their program development using about a dozen commands. This section introduces the essential commands needed by all users. These commands allow you to:

- Gain admittance to the computer system (LOGIN).
- Change the working directory (ATTACH).
- Create new directories for work organization (CREATE).
- Secure directories against intrusion (PASSWD).
- Remove directories which are no longer needed (DELETE).
- Examine the location of the working directory and its contents (LISTF).
- Look at the availability and current usage of system resources - space, users, etc. (AVAIL, STATUS, USERS).

- Create files at the terminal or enter them from tape, etc. (MAGNET, CRSER, ED. See Section 4).
- Rename files (CNAME).
- Determine file size (SIZE).
- Examine files (SLIST).
- Print files (SPOOL).
- Remove unneeded files (DELETE).
- Allow controlled access to files (PROTEC).
- Complete a work session (LOGOUT).

ACCESSING THE SYSTEM

In order to access or work in the system, the user must first follow a procedure known as 'login'. 'Logging in' identifies the user to the system and establishes the initial contact between system and user (via a terminal). Once logged in, the user has access to a working directory (work area), to files and to other system resources. The format of the LOGIN command is:

```
LOGIN ufd-name [password] [-ON nodename]
```

ufd-name	The name of your login directory.
password	Must be included if the directory has a password.
-ON nodename	Used for remote login across PRIMENET network.

Example:

```
LOGIN DOUROS NIX
DOUROS (21) LOGGED IN AT 1Ø'33 112878
```

The number in parentheses is the PRIMOS-assigned user number (also called 'job' number). The time is expressed in 24-hour format. The date is expressed as mmddyy (Month Day Year). The word NIX, in this example, is the password on the login directory.

When logging into the system, typing errors, incorrect passwords, or similar errors may cause error messages to be displayed. Most are self-explanatory. For a detailed discussion, see the New User's Guide to EDITOR and RUNOFF.

DIRECTORY OPERATIONS

Changing the Working Directory

After logging in, the user's working directory is set to the login UFD by PRIMOS. The user can move to another directory in the PRIMOS tree structure (i.e., attach) with the ATTACH command. The format is:

ATTACH new-directory

new-directory is the pathname of the new working directory.

Note

If any of the directories in the pathname have passwords, the entire pathname must be enclosed in single quotes, as in:

A 'BEECH SECRET>BRANCH5'

To set the MFD of a disk as the working directory, the format is slightly different:

ATTACH '<volume>MFD mfd-password'

volume is either the literal volume name or the logical disk number, and mfd-password is the password of the MFD. A password is always required for a MFD.

Recovering from Errors While Attaching: If an error message is returned following an ATTACH command (for example, if a UFD is not found), the user remains attached to the previous working directory.

Creating New Directories

To organize tasks and work efficiently, it is often advantageous to create new sub-UFDs. These sub-UFDs can be created within UFDs or other sub-UFDs with the CREATE command. They can contain files and/or other sub-directories. The format is:

CREATE pathname

The pathname specifies the directory in which the sub-UFD is being created, as well as the name of the new directory.

Example:

CREATE <1>TOPS>MIDDLE>BOTTOM

The sub-UFD BOTTOM is created in the sub-UFD MIDDLE, which in turn is found in the UFD TOPS, which is in the MFD of disk volume 1.

Two files or sub-UFDs of the same name are not permitted in a directory. If this is inadvertently attempted, PRIMOS will return the message: ALREADY EXISTS.

Assigning Directory Passwords

Directories may be secured against unauthorized users by assigning passwords with the PASSWD command. There are two levels of passwords: owner and non-owner. If you give the owner password in an ATTACH command, you have owner status; if you give the non-owner password in an ATTACH command, you have non-owner status. Files can be given different access rights for owners and non-owners with the PROTEC command (see Controlling File Access).

The PASSWD command replaces any existing password(s) on the working directory with one or two new passwords, or assigns passwords to this directory if there are none. The format is:

PASSWD owner-password [non-owner-password]

The owner-password is specified first; the non-owner-password, if given, follows. If a non-owner password is not specified, the default is null; then, any password (except the owner password) or none allows access to this directory as a non-owner.

Example:

```
OK, A DOUROS NIX
OK, PASSWD US THEM
```

The old password NIX is replaced by the owner password US, and the non-owner password THEM.

Deleting Directories

When directories are no longer needed they may be removed from the system to provide more room for other uses. The DELETE command can also delete empty subdirectories from a given directory. The format is:

DELETE pathname

Sub-UFDs that are not empty, i.e., that still contain files or subdirectories, cannot be deleted with this command. All entries in the directory must be deleted first. If an attempt is made to delete directories containing files, PRIMOS prints the message:

DIRECTORY NOT EMPTY

Examining Contents of a Directory

After logging in or attaching to a directory, the user can examine the contents of this directory with the LISTF command which generates a list of the files and sub-directories in the current directory. The format is:

LISTF

For example, the working directory is called LAURA. The following list will be generated when LISTF is entered at the terminal:

```
OK, LISTF

UFD=LAURA  6 OWNER

$QUERY  BOILER  EX       LETTER  QUERY   OLISTF  BASICPROGS
OUTLINE $OUTLINE         MQL     $MQL    $LETTER MQL.LETTER      FTN1Ø
EXAMPLES         FUTIL.1Ø         $FUTIL.1Ø

OK,
```

The number following the UFD-name is the logical device number, in this case, 6. The words OWNER or NONOWN follow this number, indicating the user status in this directory. (See Securing Directories).

If no files are contained in a directory, .NULL. is printed instead of a list of files.

SYSTEM INFORMATION

Table 3-1 summarizes useful information you may need about the system and how to obtain it.

Table 3-1
Useful System Information

Item	Use	PRIMOS commands
Number of users	Indicates system resource usage and expected performance.	STATUS USERS (user list) USERS (number of users)
User login UFD	Identifies user who spooled text file (printed on banner).	STATUS, STATUS UNITS
User number		STATUS
User line number		STATUS
User physical device		STATUS
Open file units	Avoids conflict when using files.	STATUS, STATUS UNITS
Disks in operation		STATUS, STATUS DISKS
Assigned peripheral devices	Tells if spool printer is working; if devices are available.	STATUS USERS
User priorities		STATUS USERS
Other user numbers		STATUS USERS
Your phantom user number	Logs out your phantoms.	STATUS USERS
Network information	Tells if network is available.	STATUS, STATUS NET
Login nodename		STATUS, STATUS UNITS
Records available	Tells if there is enough room for file building, sorting, etc.	AVAIL
System time and date	Performs time logging in audit files.	DATE (see section 10)
Computer time used since login	Measures program execution time.	TIME (see section 10)

Spool queue contents	Tells if job has been printed.	SPOOL -LIST
CX queue contents	Tells if job has been completed.	CX -ALL (see section 10)

Note

Information given by any STATUS command is also given by the STATUS ALL command.

FILE OPERATIONS

Creating and Modifying Files

Text files may be created and modified using the text editor (ED). Files may be transferred from other systems using magnetic tape (MAGNET command), paper tape (ED command), or punched cards (CRSER command). These commands are described in Section 3.

Changing File Names

It is often convenient or necessary to change the name of a file or a directory. This is done with the CNAME command. The format is:

```
CNAME old-name new-name
```

old-name is the pathname of the file to be renamed, and new-name is the new filename.

Example:

```
CN TOOLS>RPGII>TEST OLDTEST
```

The file named TEST in the sub-UFD RPGII in the UFD TOOLS is changed to OLDTEST. Since no disk was specified all MFDs (starting with logical disk Ø) are searched for the UFD TOOLS.

If new-name already exists, PRIMOS will display the message:

```
ALREADY EXISTS
```

An incorrect old-name prompts the message:

```
NOT FOUND
ER!
```

Determining File Size

The size (in decimal records) of a file is obtained with the SIZE command. This command returns the number of records in the file specified by the given pathname. The number of records in a file is defined as the total number of data words divided by 448. However, a zero-word length file always contains one record. The format is:

SIZE pathname

Example:

```
OK, SIZE GLOSSARY
GO
    14 RECORDS IN FILE

OK,
```

Examining File Contents

Contents of a program or any text file can be examined at the terminal with the SLIST command. The format is:

SLIST pathname

The file specified by the given pathname is displayed at the terminal. It is possible to suspend the terminal display as it is printing. This procedure is explained in Section 1Ø (Terminal operations).

Obtaining Copies of Files

Printed copies of files from a line printer are obtained with the SPOOL command. It has several options, some of which will not apply to all systems, as systems may be configured differently. The format is:

SPOOL pathname

PRIMOS makes a copy of pathname in the Spool Queue List for the line printer, and displays the message:

YOUR SPOOL FILE IS PRTxxx (length)

xxx is a 3-digit number which identifies the file in the Spool Queue List. The reason for a list, rather than just having each file spooled out as the request comes, is that some requests are very long - hundreds of pages. PRIMOS spools out the shorter files as soon as possible, rather than make the user wait while the long files are printed. The length (SHORT or LONG) which follows the SPOOL message is the category to which the file has been assigned. It is possible to check the status of a SPOOL request by giving the command:

SPOOL -LIST

Example:

```
OK, SPOOL $S2.3Ø57
GO
YOUR SPOOL FILE IS PRTØØ6  (LONG)  REV 15.2**

OK, SPOOL -LIST
GO

 USER    FILE    DATE/TIME    OPTS   SIZE  NAME          FORM    DEFER
------  ------  -----------  ------  ----  -----------  ------  -----
SOPHIE  PRTØØ5  1Ø/25 14:26  S          5  $UNFUNDED    W.WIBA
TEKMAN  PRTØØ6  1Ø/25 15:46  L         22  $S2.3Ø57

OK,
```

To cancel a spool request, the command format is:

SPOOL -CANCEL PRTxxx

xxx is the number of your Spool File.

For example:

```
OK, SPOOL -CANCEL PRTØ13
GO
PRTØ13 CANCELLED.

OK,
```

Deferring Printing: The -DEFER option tells the Spooler not to begin printing the indicated file until the system time matches the time specified with DEFER. This also permits you to enter SPOOL requests at your convenience, rather than waiting for the appropriate hour.

Specify the DEFER option by:

SPOOL filename -DEFER 'time'

The value for 'time' can be expressed either in 24-hour format (ØØ:ØØ = Midnight) or in 12-hour format followed by AM or PM (12:ØØ AM = Midnight). The format for 'time' is 'HH:MM', where HH is hours, ":" is any character, and MM is minutes. If you specify -DEFER but omit time you will get the prompt:

ENTER DEFERRED PRINT TIME:

If 'time' is not in the correct format, you will get the above prompt again, plus this informational message:

CORRECT FORMAT IS HH:MM AM/PM.

Printing on Special Forms: Line printers traditionally use one of two types of paper -- "wide" listing paper, on which most program listings appear, and 8-1/2x11-inch white paper, which is standard for memos and documentation. Computer rooms often stock a variety of special paper forms for special purposes, such as 5-copy sets, pre-printed forms (checks, orders, invoices), or odd sizes or colors of paper.

Request a specific form by:

SPOOL filename -FORM form-name

form-name is any six-character (or less) combination of letters. A list of available form names should be obtained from the System Administrator.

Deleting Files

When files or programs are no longer needed they may be removed from the system to provide more room for other uses. The DELETE command deletes files from the working directory. The format is:

DELETE pathname

Controlling File Access

Assigning passwords to directories allows users working in a directory to be classified as owners or non-owners, depending upon which password they use with the ATTACH command. Controlled access can be established for any file using the PROTEC command. This command sets the protection keys for users with owner and non-owner status in the directory (see Assigning Directory Passwords above). The format is:

PROTEC pathname [owner-rights] [non-owner-rights]

pathname	The name of the file to be protected.
owner-rights	A key specifying owner's access rights to file (original value=7).
non-owner-rights	A key specifying the non-owner's access rights (original value=Ø).

The values and meanings of the access keys are:

key	Rights
Ø	No access of any kind allowed
1	Read only
2	Write only
3	Read and Write
4	Delete and truncate
5	Delete, truncate and read
6	Delete, truncate and write
7	All access

Note

The default protection keys associated with any newly created file or UFD are: 7 Ø. The owner is given ALL rights and the non-owner is given none.

Example:

```
PROTEC <OLD>MYUFD>SECRET 7 1
```

In this example, protection rights are set on the file SECRET in the UFD MYUFD so that all rights are given to the owner and only read rights are given to the non-owner.

COMPLETING A WORK SESSION

When finished with a session at the terminal, give the LOGOUT command. The format is:

LOGOUT

PRIMOS acknowledges the command with the following message:

```
UFD-name (user-number) LOGGED OUT AT (time) (date)
TIME USED = terminal-time CPU-time I/O-time
```

user-number	The number assigned at LOGIN.
terminal-time	The amount of elapsed clock time between LOGIN and LOGOUT in hours and minutes.
CPU-time	Central Processing Unit time consumed in minutes and seconds.
I/O-time	The amount of input/output time used in minutes and seconds.

It is good practice to log out after every session. This closes all files and releases the PRIMOS process to another user. However, if you forget to log out, there is no serious harm done. The system will automatically log out an unused terminal after a time delay. This delay is set by the System Administrator (the default is 1000 minutes but most System Administrators will lower this value).

SECTION 4

ENTERING AND MANIPULATING SOURCE PROGRAMS

ENTRY FROM OTHER MEDIA

Existing source programs resident on punched cards, magnetic tape, or punched paper tape can easily be read into disk files using PRIMOS-level utilities. In addition, the punched card and magnetic tape transfer utilities will translate from BCD or EBCDIC representation into ASCII representation, saving considerable time and effort. The general order of operations for input from a peripheral device is:

1. Obtain exclusive use of the device (Assigning).
2. Transfer programs with appropriate utility.
3. Relinquish exclusive use of the device (Unassigning).

It may be necessary to alter existing programs to conform to Prime's RPG II standards. The Editor (described later in this section) is used for these modifications.

Assigning A Device

Assigning a device gives the user exclusive control over that peripheral device. The PRIMOS-level ASSIGN command is given from the terminal:

ASSIGN device [-WAIT]

device is a mnemonic for the appropriate peripheral

CR	Card Reader
MTn	Magnetic Tape Unit n (Ø-7)
PTR	Paper Tape Reader

-WAIT is an optional parameter. If included, it queues the ASSIGN command if the device is already in use. The assignment request remains in the queue until the device becomes available or the user types the BREAK key at the terminal; both occurrences return the user to PRIMOS. If the requested device is not available and the -WAIT parameter has not been included, the error message:

DEVICE IN USE

is printed at the terminal.

After all I/O operations are completed, exclusive use is relinquished by the command:

UNASSIGN device

device is the same mnemonic used in the ASSIGN command.

Reading Punched Cards

Assign use of the parallel interface card reader by:

AS CR -WAIT

To read cards from the card reader, load the card deck into the device and enter the command:

CRMPC deck-image

deck-image is the pathname of the file into which the card images are to be loaded.

Source deck header control cards are set up as follows:

Source deck representation	Columns 1 and 2 of deck header card
BCD	$6
EBCDIC	$9
ASCII	no header card

Reading continues until a card with $E in columns 1 and 2 is encountered (end of deck); control returns to PRIMOS and the file is closed. If the cards are exhausted (or the reader is halted by the user), control returns to PRIMOS but the file is not closed. If more cards are to be read into the file, the reader should be reloaded; reading is resumed by the command START at the terminal. The command:

CLOSE ALL

or

CLOSE deck-image

will close the file.

Example of card reading session:

```
OK, AS CR -WAIT
OK, CRMPC old-program-1
OK, UN CR
OK,
```

If a serial interface card reader is used, the process is similar, with slightly different reader commands.

```
OK, AS CARDR -WAIT
OK, CRSER old-program-2
OK, UN CARDR
OK,
```

CARDR may be abbreviated to CAR.

Reading Magnetic Tape

Assign use of the magnetic tape drive by:

```
AS MTx -WAIT
```

x is the tape drive unit number: Ø-7.

Mount the tape on the selected drive unit and read the tape with PRIMOS' MAGNET utility:

```
OK, MAGNET
GO

MAGNET 15.2 15-JULY-78

OPTION: READ

MTU# = unit-number [/tracks]
```

unit-number	The number of the magnetic tape drive unit which was previously assigned.
tracks	Either 7 or 9; if this parameter is omitted, 9-track tape is assumed.

MAGNET then asks a series of questions about the tape format:

```
MTFILE# = tape-file-number
```

tape-file-number	The file number of the tape. A positive integer causes the tape to be rewound and then positioned to the file number; a Ø causes no repositioning of the tape.

```
LOGICAL RECORD SIZE = 8Ø
```

This is the number of bytes/line image; normally this is 8Ø for a RPG II source program.

BLOCKING FACTOR = blocking-factor

blocking-factor is the number of logical records per tape record.

ASCII, BCD, BINARY, OR EBCDIC? data-representation

data-representation	action
ASCII	Transfer.
BCD	Translate to ASCII from 7-track tape.
BINARY	Transfer verbatim.
EBCDIC	Translate to ASCII.

FULL OR PARTIAL RECORD TRANSLATION? answer

answer is FULL or PARTIAL. The question is asked only for BCD or EBCDIC representations. Partial translation allows specified bytes in the record to be transferred to disk without translation to ASCII. The partial option is useful when transferring data files, but almost all source programs will be transferred with the full option.

OUTPUT FILENAME: filename

filename is the name of the file in the UFD into which the magnetic tape is to read. If the filename already exists in the UFD, the question:

OK TO DELETE OLD filename? answer

is asked. A NO will cause the request for an output filename to be repeated. A YES will cause the transfer to begin; upon completion, the following message is printed out:

DONE, tape-records RECORDS READ, disk-records DISK RECORDS OUTPUT OK,

Use of the tape drive unit should then be relinquished by UN MTx.

Reading Punched Paper Tape

First load tape into reader; then assign tape reader. Source programs punched on paper tape in ASCII representation can be read into a disk file with the Editor utility.

```
OK, AS PTR -WAIT      Assign tape reader
OK, ED                Invoke Editor
GO
INPUT
(CR)                  Switch to EDIT mode
EDIT
INPUT (PTR)           Input from tape reader
EDIT                  Tape is being read
FILE filename         File input under filename
OK, UN PTR            Unassign tape reader
```

ENTERING AND MODIFYING PROGRAMS - THE EDITOR

Programs are normally entered into the computer using Prime's Text Editor (ED). This editor is a line-oriented text processor whose line pointer is always located at the last line processed (whether the processing action is printing, locating, moving pointer, etc). The Editor operates in two modes, INPUT and EDIT.

Using the Editor

When creating a new file, the Editor is invoked by

```
ED
```

which places the Editor in the INPUT mode. When modifying an existing filename, the Editor is invoked by

```
ED filename
```

which places the Editor in the EDIT mode. A RETURN with no preceding characters on that line switches the Editor from one mode to another.

Input Mode

The INPUT mode is used when entering text information into a file (e.g., creating a program). The word INPUT is displayed at the user's terminal to indicate the Editor has entered that mode. The RETURN key terminates the current line and prepares the Editor to receive a new line. Tabulation is done with the backslash (\) character. Each backslash represents the first, second, etc., tab setting; the default tabs are at columns 6, 15, and 3Ø. These settings may be overridden and up to 8 tab settings may be specified by the user with the TABSET command (described later). A RETURN with no text preceding it puts the Editor into EDIT mode.

Edit Mode

The EDIT mode is used when the contents of the file are to be modified. More than 5Ø commands are available, although users will find that a small subset of these will suffice for most purposes. The commands are listed and described later in this section.

In EDIT mode, the Editor maintains an internal line pointer at the current line (the last line processed). Commands such as TOP, BOTTOM, FIND, and LOCATE, move this pointer. WHERE prints out the current line number; POINT moves the pointer to a specified line number. The MODE NUMBER command causes the line number to be printed out whenever a line of text is printed. All commands for location and modification begin processing with the current line.

A RETURN without any preceding characters puts the Editor into the INPUT mode.

Special Characters

In either mode, a single character can be erased with the erase character (default is "). For each " typed, a character is erased (from right to left). The entire current line may be deleted by typing the kill character (default is ?). A line followed by a ? is null, and a RETURN at that point will switch the Editor into the other mode.

In input mode, the semicolon (;) is equivalent to a CR (ends a line of input). In edit mode, semicolons in a character string are treated as a printing character, otherwise, semicolons separate multiple commands entered on the same line.

Saving Files

Orderly termination of an Editor session is done from EDIT mode. The command:

FILE filename

writes the current version of the edited file to the disk under the name filename. The specified file will be created if it did not previously exist or overwritten if it does exist. If an existing file is being modified, the command

FILE

writes the edited version to the disk with the old filename. After execution of the filing command, control is returned to PRIMOS.

Useful Techniques

The following will aid the user in adapting to Prime's Editor.

Tab Settings: When entering source code, much time can be saved using the TABSET command. In INPUT mode, each \ character is interpreted as one tab setting; the default values are columns 6, 15, and 3Ø. Tabs may be set to whatever values each programmer finds useful.

Moving Lines of Code: Any number of lines can be moved from one location to another using the DUNLOAD command. DUNLOAD deletes these lines as it writes them into an auxiliary file. A LOAD command loads the new file at the desired point. Any number of lines can be copied from one location in a program to another using the UNLOAD command. UNLOAD does not delete these lines as it writes them into an auxiliary file. A LOAD command loads the copy from the new file at the desired point.

Overlaying Existing Lines: A new line may be overlaid on an existing source program line with the OVERLAY command in conjunction with the TABSET command.

Finding a Line by Statement Number: The FIND command may be used to locate a statement number in a RPG II program.

Modifying a Line Without Changing Character Positions: The MODIFY command is used when a line must be modified but the absolute column alignment must remain the same.

Displaying Column Numbers: The MODE COLUMN command displays column numbers across the terminal screen, assisting in the alignment essential to RPG specifications. This command is given in EDIT mode; it should be followed by two consecutive carriage returns. The system will respond with INPUT and a string of column numbers. Source program lines may then be input.

Sample Editing Session

A few commands having particular significance to the RPG programmer are covered in the following example below. The user may choose to work free form, or with the TABSET or MODE COLUMN commands to facilitate alignment of source program specifications and file data.

See the list following the example for an explanation of the commands.

```
OK, ED
```

The editor begins in Input mode when the ED command is given.

```
GO
INPUT
(CR)
EDIT
C"MODE COLUMN
```

A carriage return invokes Edit mode.

The MODE COLUMN command is given in Edit mode. With a return to Input mode, it will automatically cause a column number display. The delete character, ", erases the previous character, C, to correct an error.

```
(CR)
INPUT
        1         2         3         4         5         6         7         8
12346789012345678901234567890123456789012345678901234567890123456789012345678901234567890

0001   *SAMPLE PROGRAM?0001 *SAMPLE PROGRAM
(CR)
EDIT
```

Specification entries are keyed in, aligned by column. The kill character, (?), deletes all characters preceding the ?. It is followed immediately by the corrected entry. To verify the correction, a PRINT command may be given in Edit mode. The correction is displayed, properly aligned.

```
P
0001 *SAMPLE PROGRAM
(CR)
INPUT
        1         2         3         4         5         6         7         8
12346789012345678901234567890123456789012345678901234567890123456789012345678901234567890
0002 FSAM2    IP  F  96  96              DISK
0003 FDISKOUT O   F 256 128 06AI      1 DISK
0004 FPRINT1  O   F  96  96              PRINTER
    .
    .
    .
0042 OPRINT   T  204   LR
```

Each time Input mode is invoked, there is an automatic column number display. This convention may be used at terminal overflow.

```
(CR)
EDIT
```

```
MODE NCOLUMN
```

The MODE NCOLUMN command, given in EDIT mode, turns off the column display. The FILE command, issued in Edit mode, writes the contents of the edited file to the filename specified (in this case, RPGSII). A FILE command also causes a return from the Editor to PRIMOS.

```
FILE RPGSII
OK,
ED
```

Invoking the Editor once again, a data file can be keyed in for a source program.

```
GO
INPUT
(CR)
EDIT
TA  1 9 24
```

The TABSET command given in EDIT mode, can be used for field alignment.

```
(CR)
INPUT
KEY\DESCRIPTION\PAGE
000005\RECORD NUMBER\ 1
000010\RECORD NUMBER\ 2
```

Data are keyed in using the tab character (\) to accurately position fields.

```
EDIT
```

To display the data for tabulation verification, invoke EDIT mode.

```
N-2
```

Position the line pointer to the first line to be verified (current line-2).

```
P3
KEY     DESCRIPTION    PAGE
000005  RECORD NUMBER    1
000010  RECORD NUMBER    2
```

This PRINT command will display 3 lines, beginning with the one designated by the line pointer. (If Verify Mode is in effect, the result of each command given in EDIT Mode is automatically displayed, obviating the need for PRINT.) Data entries have been properly tabulated.

Editor Command Summary

The following is an alphabetic list of each Editor command and its function. Acceptable command abbreviations are underlined. Especially useful commands are indicated with a bullet (•). For a detailed description of all commands, see the Editor Reference Section of The New User's Guide To EDITOR and RUNOFF.

Note

The string parameter in a command is any series of ASCII characters including leading, trailing, or embedded blanks.

Command — Function

• APPEND string

Appends string to the end of the current line.

• BOTTOM

Moves the pointer beyond the last line of the file.

BRIEF

Speeds editing by suppressing the (default) verification responses to certain Editor commands.

• CHANGE/string-1/string-2/[G] [n]

Replaces string-1 with string-2 for n lines. If G is omitted, only the first occurrence of string-1 on each line is changed; if G is present, all occurrences on n lines are changed.

• DELETE [n]

Deletes n lines, including the current line (default n=1).

DELETE TO string

Deletes all lines up to, but not including line containing string.

• DUNLOAD filename [n]

Deletes n lines from the current file and writes them into filename. (Default n=1.)

DUNLOAD filename TO string

Same as DELETE...TO, but writes deleted lines into filename.

ERASE character

Sets erase character to character.

- FILE [filename]

Writes the contents of the current file into filename and QUITs to PRIMOS.

FIND string

Moves the pointer down to the first line beginning with string.

- FIND(n) string

Moves the pointer down to the first line with string beginning in column n.

GMODIFY

Allows the user to enter a string of subcommands which modify characters within a line.

INPUT { (ASR) / (PTR) / (TTY) }

Reads text from the specified input device: ASR (Teletype paper-tape reader), PTR (high-speed paper tape reader) or TTY (terminal). Default is TTY.

- INSERT string

Inserts string after current line.

KILL character

Sets kill character to character.

LINESZ [n]

Changes maximum line length.

- LOAD filename

Loads filename into text following the current line.

- LOCATE string

Moves pointer forward to the first line containing string, which may contain leading and trailing blanks.

MODE COLUMN

Displays column numbers whenever INPUT mode is entered.

MODE COUNT start increment width{PRINT / BLANK / SUPRESS}

Turns on the automatic incremented counter.

MODE NCOLUMN

Turns off the column display (default).

MODE NCOUNT

Suspends counter incrementing (default).

MODE NUMBER

Displays line numbers in front of printed line.

MODE NNUMBER

Turns off the line number display (default).

MODE PRALL

Prints lower case characters if device has that capability.

MODE PRUPPER

Prints all characters as upper case. Precedes lower case characters with an ^L and precedes upper case characters with an ^U if the device is upper case only.

MODE PROMPT

Prints prompt characters for INPUT and EDIT modes.

MODE NPROMPT

Stops printing of INPUT and Edit prompt characters (default).

MODIFY/string-2/string-1/[G]]n]

Superposes string-1 onto string-2 for n lines. If G is omitted, only the first occurrence of string-1 on each line is modified; otherwise all occurrences of string-1 are modified.

MOVE buffer-1 { buffer-2 / /string/ }

Move string or contents of buffer-2 into buffer-1.

- NEXT [n]

Moves the pointer n lines forward or backward (default n=1).

NFIND string

Moves pointer down to first line NOT beginning with string.

NFIND(n) string

Moves pointer down to first line in which string does not start in column n.

- OVERLAY string

Superposes string on current line. Use tabs to start in middle of line; use ! to delete existing characters.

PAUSE

Returns to PRIMOS level without changing the Editor state.

POINT line-number

Relocates the pointer to line-number.

- PRINT [n]

Prints the current line or n lines beginning with the current line.

PSYMBOL

Prints a list of current symbol characters and their function.

PTABSET tab-1...tab-8

Provides for a setup of tabs on devices that have physical tab stops.

PUNCH (ASR) / (PTP) [n]

Punches n lines on high- or low-speed paper-tape punch.

QUIT

Returns control to PRIMOS without filing text.

RETYPE string

The current line is replaced by string.

SYMBOL name character

Changes a symbol name to character. Current default values are:

Name	Default Characters
KILL	?
ERASE	"
WILD	!
BLANK	#
TAB	\
ESCAPE	^
SEMICO	;
CPROMPT	$
DPROMPT	&

TABSET tab-1...tab-8

Sets up to eight logical tab stops to be invoked by the tab symbol (\).

- TOP

Moves the pointer one line before the first line of text.

- UNLOAD filename [n]

Copies n lines into filename.

UNLOAD filename TO string

Unloads lines from current file into filename until string is found.

- VERIFY

Displays each line after completion of certain commands. (Default).

WHERE

Prints the current line number.

XEQ buffer

Executes the contents of buffer. See MOVE.

*[n]

Repeat symbol. Causes preceding command to be repeated n times. If n is omitted, the command repeats until the bottom of file is reached.

LISTING PROGRAMS

Terminal Listing

Source programs may be listed at the terminal, by using the SLIST command described in Section 3.

Line Printer Listing

Use the SPOOL command (Section 3) to obtain a copy of a source file on the system line printer.

RENAMING AND DELETING PROGRAMS

Renaming

Programs may be renamed with the PRIMOS command CNAME (Section 3). You must have owner status in the UFD in order to use this command.

Deleting

Programs may be deleted with the PRIMOS command DELETE (Section 3). You must have owner status in order to use this command.

SECTION 5

COMPILING THE PROGRAM

INTRODUCTION

Prime's RPG II Compiler is a one-pass compiler, producing optimized code.

Source programs must meet the requirements of Prime RPG II as specified in this guide.

The compiler generates object code for the R-identity. R-identity code is loaded with Prime's Linking Loader (LOAD), described in Section 6.

USING THE COMPILER

The RPG II Compiler is invoked by the RPG command to PRIMOS:

RPG pathname [-options]

or

RPG [-option-1] -I pathname...[-option-n]

pathname	The pathname of the RPG II source program file.
options, etc.	The mnemonics for the options controlling compiler functions such as I/O device specification, listings, and others.

All mnemonic options must be preceded by a dash "-". The name of the source program file must be specified either as the first expression following RPG or as -I pathname (alternatively, -S pathname) but not both.

Examples:

RPG TEST1 -NOXREF -BANNER -LISTING SPOOL

or

RPG -LISTING SPOOL -NOXREF -INPUT TEST1 -BANNER

are equivalent.

The meanings of the options are discussed later in this section.

END OF COMPILATION MESSAGE

After the compiler has completed a pass of the specified input file, and generated object code and listing output to the devices specified by the option list, it prints an End of Compilation message at the user terminal.

The format of the compiler message is:

xxx WARNINGS, yyy ERRORS [RPG REV zz.zz]

xxx is the number of WARNING messages.

yyy is the number of ERROR messages.

zz.zz is the revision number of the RPG compiler.

Example:

ØØ1 WARNINGS, ØØØ ERRORS [RPG REV 16.1Ø]

After compilation of the source file, control returns to PRIMOS.

COMPILER ERROR AND WARNING MESSAGES

The general format of error and warning messages is:

****ERROR, LINE (xxxx) COLUMN yy-zz [contents]----message

or

**WARNING, LINE (xxxx) COLUMN yy-zz [contents]----message

xxxx	Line number of the statement in error.
yy-zz	Beginning and ending column numbers of the field in error. If yy is the same as zz, the -zz portion is omitted.
contents	The contents of columns yy-zz.
message	Some comments about the field in error. A list of the warning and error messages is in Appendix E.

COMPILER OPTIONS

Normally, the source file is stored in the disk file system, the binary (object) file is created on the disk, and the listing file (if any) is created either on the disk, at the user terminal, or spooled directly to the line printer. In these cases, all instructions to the compiler are given by mnemonics in the RPG command line.

The A-register settings are the instructions to the RPG II compiler (set at compilation time) telling it which functions are to be enabled, and specifying the I/O. Using the mnemonic options establishes the values of these registers for the user automatically. Most users will have no need to set the octal values in these registers explicitly.

It is possible for a user to employ other peripheral devices (paper tape punch/reader, card punch/reader, magnetic tape) for making source, listing, or binary files. It would generally be preferable to bring the source program onto the disk, compile using the option mnemonics, and then transfer the listing and/or binary files to the desired device using PRIMOS commands. If for some reason this is not possible, the user may explicitly set the A-register values to allow direct access to and from these devices. The previous method of specifying compiler options (by setting the A-register value explicitly) is still valid. This means existing command files which set the A register need not be changed. (See Section 1Ø).

Compiler Functions

The compiler functions enabled by the mnemonic options may be considered to fall into four groups (Table 5-1).

- Specify Input/Output Devices
- Enable Listings/Cross References
- Debugging Aids

The defaults listed in this section are those supplied by Prime. The System Administrator may change these at any particular installation. The programmer should check with the system manager to determine if defaults have been changed and, if so, which options are the new defaults.

Table 5-1. Compiler Option Mnemonics
(● indicates Prime-supplied defaults)

Specify Input/Output Devices

●	BINARY	Specify binary (object) file.
●	INPUT	Specify source program file.
●	LISTING	Specify listing file.
●	SOURCE	Specify source file (same as INPUT).

Enable Listings/Cross References

●	ERRTTY	Print error messages at user terminal.
	NOERRTTY	Suppress error messages to terminal.
	NOXREF	Suppress cross-reference listing.
●	XREF	Print cross-reference listing.

Debugging Aids

	BANNER	Print column index banner before each non-comment line.
●	NOBANNER	Suppress column index banner before each non-comment line.
●	NOOBDATA	Suppress octal listing of generated object data within source listing.
●	NOSEQCHK	Do not check columns 3-5 for proper sequencing.
	NOSTATUS	Suppress printing of current RPG status (H, F, E, L, I, C, or O).
	OBDATA	Include octal listing of generated object data within source listing.
	SEQCHK	Check columns 3-5 of source program for proper sequencing.
●	STATUS	Print current RPG status at user terminal.

Specify Input/Output Devices

These options allow the user to inform the compiler of the input source filename and to specify the listing and binary (object) files.

-INPUT	define input file/device. (alternatively -SOURCE) (example: -I TEST or -S TEST)
-I pathname	The source program filename is pathname.
-BINARY	To override default, define binary (object) file/device.
-B pathname	The binary file will be created with the pathname specified. (example -B BTEST)
-B NO	No binary file will be created. This might be chosen if only the listing file were desired at earlier stages of program development.
-B YES	The binary file is created with the default name B_filename, where filename is the name of the source program file in the UFD in which the source program file resides. The binary file, however, is created in the UFD to which the user is attached when invoking the compiler. If the BINARY option is not included in the command line option list, it is equivalent to -B YES.
-LISTING	To override default, define listing file.
-L pathname	The listing file will be created with the pathname specified. (Example -L ELM>LTEST)
-L NO	No listing file will be created. At later stages in program development or when minor modifications are made to programs, it may not be considered necessary to get a source program listing.
-L YES	The listing file is created with the default name L_filename, where filename is the name of the source program file in the UFD in which the source program file resides. The listing file, however, is created in the UFD to which the user is attached when invoking the compiler.
-L TTY	The listing is printed at the user terminal.

-L SPOOL — The listing file is spooled directly to the line printer.

If this option is not included in the command line option list, it is equivalent to -L YES.

Enable Listings/Cross References

These options enable or suppress error listings and cross-reference listings (concordances). Except for ERRTTY (defined below), the enabling has no effect unless an output device or file is specified by the -L option.

The error- and cross-reference listings discussed below are generated for the following RPG II program example, SAM1 (listed in Appendix A).

One warning will be found in this program: Ø1 will be found in columns 67-69 of statement 24; these columns should be blank.

Note

The letters F, I, C, O in the listing are printed by the compiler as it begins processing the corresponding specification sheet - File Description, Input, etc.

ERRTTY/NOERRTTY

ERRTTY (the default) prints error messages at the user terminal. This feature may be suppressed by including NOERRTTY in the option list.

Examples:

```
OK, RPG SAM1
GO
 F
**WARNING, LINE (ØØ24) COLUMN 67-69 [  Ø1  ]----SHOULD BE BLANK.
 I
 C
 O
ØØ1 WARNINGS, ØØØ ERRORS [RPG REV 16.1Ø]

OK, RPG SAM1 -NOERRTTY
GO
 F
 I
 C
 O
ØØ1 WARNINGS, ØØØ ERRORS [RPG REV 16.1Ø]

OK,
```

NOXREF/XREF

XREF is the default. XREF appends a concordance to the end of the listing in the listing file/device Concordances are cross-reference tables between program symbols, their line numbers and storage locations in memory. The cross-reference generation can be suppressed by including the option -NOXREF in the command line.

The concordance which is appended to the end of the listing file for the SAM1 program is:

```
                **********************
                *   CROSS REFERENCE    *
                **********************

0           [LITERAL------------>002172]   (0030)   (0031)
1           [LITERAL------------>002177]   (0034)   (0039)
5           [LITERAL------------>002204]   (0033)
505         [LITERAL------------>002211]   (0035)
COUNT       [FIELD--------------->002217]  (0030)   (0033)   (0033)   (0035)   (0058)
DISKOUT     [FILE---------------->000276]  (0024)   (0057)
NODATA      [FIELD--------------->002226]  (0028)
PRINT1      [FILE---------------->000603]  (0025)   (0041)
RECNBR      [FIELD--------------->002255]  (0031)   (0034)   (0034)   (0039)   (0039)
                                            (0044)   (0060)
REPEAT      [LABEL--------------->002263]  (0032)   (0037)
SAM2        [FILE---------------->000011]  (0023)   (0027)


001 WARNINGS, 000 ERRORS [RPG REV 16.10]

OK,
```

The first column is the symbol. The second and third columns (which are enclosed in square brackets) are the symbol type and storage address. The numbers in parentheses are the line numbers of the statements in which the symbol appears.

Debugging Aids

These options enable or suppress information display and results of tests which provide aid in debugging the RPG program. All printing is inserted into the listing file. Output can be sent to the terminal by inserting the option -L TTY in the command line.

BANNER/NOBANNER

The BANNER option causes each non-comment line in the program to be preceded by a column index banner. This enables the programmer to check if field items have been entered in the proper columns. It is especially useful if the programmer does not have a PRIME RPG II DEBUGGING TEMPLATE, FDR3275, or if the printout is not 12 pitch. The default is NOBANNER, which suppresses the printing of the column index banner.

Example:

```
OK, RPG SAM1 -L TTY -BANNER
GO
(0002)  *
(0003)       F**********************************************
          .
          .
          .
         ---------1---------2---------3---------4---------5---------6---------7--
(0030)       C   01                  Z-ADD0          COUNT   60
 C
         ---------1---------2---------3---------4---------5---------6---------7--
(0031)       C   01                  Z-ADD0          RECNBR  30
         ---------1---------2---------3---------4---------5---------6---------7--
(0032)       C             REPEAT    TAG
         ---------1---------2---------3---------4---------5---------6---------7--
(0033)       C   01        COUNT     ADD  5          COUNT
         ---------1---------2---------3---------4---------5---------6---------7--
(0034)       C   01        RECNBR    ADD  1          RECNBR
         ---------1---------2---------3---------4---------5---------6---------7--
(0035)       C   01        COUNT     COMP 505                    02
          .
          .
          .
001 WARNINGS, 000 ERRORS [RPG REV 16.10]

OK,
```

OBDATA/NOOBDATA

The OBDATA option in the command line causes an octal listing of the generated object data to be included in the listing of the program. The default, NOOBDATA, suppresses the generation of this listing.

Example:

```
OK, RPG SAM1 -L TTY -OBDATA
GO
    000000: -->RPG
    000001:[000000]
    000002:[000000]
    000003:[000000]
    000004:[000000]
    000005:[000000]
    000006:[000000]
    000007:[000000]
    000010:[000000]
(0002)  *
(0003)       F************************************************
          .
          .
          .
(0024)       FDISKOUT O   F 256 128 06AI     1 DISK
         01
    000276: 000003
    000277: 000305
    000300: 000030
    000301: 000000
    000302:  'DI'
    000303:  'SK'
    000304:  'OU'
    000305:  'T '
    000306: 020001
    000307: 001106
    000310: 000070
    000311: 000000
    000312: 000000
**WARNING, LINE (0024) COLUMN 67-69 [  01  ]----SHOULD BE BLANK.
    000313: 000000
    000314: 000000
    000315: 000200
    000316: 000100
    000317: 000001
    000320: 000000
       ..    ....
    000602: 000000
(0025)       FPRINT1  O   F  96  96           PRINTER
    000603: 000003
    000604: 000265
    000605: 000031
    000606: 000000
```

```
    000607:  'PR'
    000610:  'IN'
    000611:  'T1'
    000612:  '  '
    000613: 020001
    000614: 000000
    000615: 000040
    000616: 000000
    000617: 000000
    000620: 000000
    000621: 000000
    000622: 000140
    000623: 000060
    000624: 000000
    000625: 000000
       ..    ....
    001067: 000000
(0026)  *
          .
          .
          .
(0030)       C   01                Z-ADD0        COUNT   60
 C
    001123: 000001
    001124: 000020
    001125: 000036
    001126: 000002
    001127: 000063
    001130: 000000
    001131: 000000
    001132: 000000
    001133: 000000
    001134:[000000]
    001135: 000000
    001136:[000000]
    001137: 000000
    001140: 000000
    001141: 000000
    001142: 000000
          .
          .
          .
(0041)       OPRINT1  T  204   LR
 O
    001363: 000007
    001364: 000011
    001365: 000051
    001366:(000603)
    001367: 004000
    001370: 030002
    001371: 000002
    001372: 000000
    001373: 000000
```

```
          .
          .
          .
***************************
*   VARIABLE ALLOCATION   *
***************************

--------------------
Ø          (N)
      LINK>>ØØ1154
    ØØ2172: Ø1ØØØ1
    ØØ2173: ØØØØØ7
    ØØ2174: ØØØØØØ
    ØØ2175: ØØØØØØ
    ØØ2176:  'Ø '
--------------------
1          (N)
      LINK>>ØØ1354
    ØØ2177: Ø1ØØØ1
    ØØ22ØØ: ØØØØØ7
    ØØ22Ø1: ØØØØØØ
    ØØ22Ø2: ØØØØØØ
    ØØ22Ø3:  '1 '
--------------------
5          (N)
      LINK>>ØØ1214
    ØØ2204: Ø1ØØØ1
    ØØ2205: ØØØØØ7
    ØØ2206: ØØØØØØ
    ØØ2207: ØØØØØØ
    ØØ221Ø:  '5 '
--------------------
5Ø5        (N)
      LINK>>ØØ1254
    ØØ2211: Ø1ØØØ3
    ØØ2212: ØØØØØ7
    ØØ2213: ØØØØØØ
    ØØ2214: ØØØØØØ
    ØØ2215:  '5Ø'
    ØØ2216:  '5 '
--------------------
COUNT      (N)
      LINK>>ØØ2Ø7Ø
    ØØ2217: Ø1ØØØ6
    ØØ222Ø: ØØØØØØ
    ØØ2221: ØØØØØØ
    ØØ2222: ØØØØØØ
    ØØ2223:  'ØØ'
       ..    ....
    ØØ2225:  'ØØ'
--------------------
NODATA     (A)
      LINK>>ØØ1113
```

```
    002226: 000001
    002227: 000000
    002230: 000000
    002231: 000000
    002232:  '  '
--------------------
PAGE        (N)
      LINK>>000005
    002233: 010004
    002234: 000000
    002235: 000000
    002236: 000000
    002237:  '00'
    002240:  '00'
--------------------
PAGE1       (N)
      LINK>>000006
    002241: 010004
    002242: 000000
    002243: 000000
    002244: 000000
    002245:  '00'
    002246:  '00'
--------------------
PAGE2       (N)
      LINK>>000007
    002247: 010004
    002250: 000000
    002251: 000000
    002252: 000000
    002253:  '00'
    002254:  '00'
--------------------
RECNBR      (N)
      LINK>>002144
    002255: 010003
    002256: 000000
    002257: 000000
    002260: 000000
    002261:  '00'
    002262:  '00'
--------------------
REPEAT      (A)
      LINK>>001314
    002263: 000000
    002264: 000004
    002265: 001163
    002266: 000000
    002267:(001163)
--------------------
RPGREV      (N)
      LINK>>000010
    002270: 010004
```

```
    002271: 000000
    002272: 000000
    002273: 000000
    002274:  '00'
    002275:  '00'
--------------------
UDATE       (N)
      LINK>>000004
    002276: 010006
    002277: 000000
    002300: 000000
    002301: 000000
    002302:  '00'
       ..    ....
    002304:  '00'
--------------------
UDAY        (N)
      LINK>>000001
    002305: 010002
    002306: 000000
    002307: 000000
    002310: 000000
    002311:  '00'
--------------------
UMONTH      (N)
      LINK>>000003
    002312: 010002
    002313: 000000
    002314: 000000
    002315: 000000
    002316:  '00'
--------------------
UYEAR       (N)
      LINK>>000002
    002317: 010002
    002320: 000000
    002321: 000000
    002322: 000000
    002323:  '00'
          .
          .
          .
001 WARNINGS, 000 ERRORS [RPG REV 16.10]

OK,
```

SEQCHK/NOSEQCHK

The option SEQCHK in the command line causes the compiler to check columns 3-5 of the source program for proper sequencing. If a line is not in sequence, the warning message:

```
**WARNING, LINE (0006) COLUMN 01-05 [ 002 ]----NOT IN SEQUENCE.
```

is printed at the user terminal. The default is NOSEQCHK which suppresses sequence checking.

STATUS/NOSTATUS

The STATUS option, which is the default, causes the compiler to print a letter (in the listing file and at the user terminal) identifying each specification sheet as it begins processing that sheet. The letters (H, F, E, L, I, C, or O) are the same as the identifiers (column 6) for the respective specification sheets. The current status reporting can be suppressed by the option NOSTATUS in the command line.

```
OK, RPG SAM1 -NOSTATUS
GO


**WARNING, LINE (0024) COLUMN 67-69 [  01  ]----SHOULD BE BLANK.
001 WARNINGS, 000 ERRORS [RPG REV 16.10]

OK, RPG SAM1
GO
 F
**WARNING, LINE (0024) COLUMN 67-69 [  01  ]----SHOULD BE BLANK.
 I
 C
 O
001 WARNINGS, 000 ERRORS [RPG REV 16.10]

OK,
```

SECTION 6

LOADING THE PROGRAM

INTRODUCTION

The PRIMOS LOAD utility converts object modules generated by the RPG II Compiler into runfiles.

The following description emphasizes the loader commands and functions that are of most use to the RPG II programmer. For a complete description of all loader commands, including those for advanced system-level programming, refer to Reference Guide, LOAD and SEG.

USING THE LOADER UNDER PRIMOS

The PRIMOS command:

LOAD

transfers control to the R-mode loader, which prints a $ prompt character and awaits a loader subcommand. After executing a command successfully, the loader repeats the $ prompt character.

When a system error (FILE IN USE, ILLEGAL NAME, NO RIGHT, etc.) is encountered, the loader prints this system error and returns its prompt symbol, $.

The loader remains in control until a QUIT (or PAUSE) subcommand returns control to PRIMOS, or an EXECUTE subcommand starts execution of the loaded program.

Load subcommands can be used in command files, but comment lines result in a CM (command error) message.

LOADING

Sequential Files

If the RPG program to be loaded uses sequential files only, the following command sequence will save load and execution time (and memory image size).

Example:

```
OK, LOAD              Invoke the Loader.
GO
$ MODE D64R           Set the addressing mode to 64R.
$ DC                  Defer the placement of library COMMON.
$ LO B SAM11          Load the RPG II program.
$ LI RPGLIB           Load the RPG sequential file library.
$ LI                  Load the system (FORTRAN) library.
$ SAVE *SAM11         Copy the memory image to a disk file.
LOAD COMPLETE
$ QUIT                Exit from the loader.

OK,
```

If the program loaded under this sequence refers to non-sequential files during execution, the program will terminate with an error message indicating that the wrong library has been loaded.

If a program referencing only sequential files is loaded using the KIDALB library (Non-sequential file references), it will execute but will be slower and require more memory.

Non-sequential Files (MIDAS)

The following sequence of commands must be used if the program refers to any non-sequential files during execution. This specifically applies to all MIDAS files.

Example:

```
OK, LOAD              Invoke the Loader.
GO
$ MODE D64R           Set the addressing mode to 64R.
$ DC                  Defer the placement of library COMMON.
$ LO B SAM1           Load the RPG II program.
$ LI RPGKID           Load the complete RPG library.
$ LI KIDALB           Load the MIDAS library.
$ LI                  Load the system (FORTRAN) library.
$ SAVE *SAM1          Copy the memory image to a disk file.
LOAD COMPLETE
$ QUIT                Exit the Loader.

OK,
```

SECTION 7

EXECUTING THE PROGRAM

EXECUTING A LOADED PROGRAM

The implicit assumption for Prime RPG II is that all RPG program and data files are in the current UFD at run-time. Further, when executing an RPG II program, the status of the A-register on entry is used to control run-time options. The command to execute a previously loaded RPG II program is:

RESUME filename 1/abcdef

filename is the executable program produced from the load sequence and saved on disk in the current UFD. Program execution begins at octal location 1ØØØ (the default value). In the above command, 1 indicates the A-register, and abcdef is the A-register string controlling run-time options for external user indicators, UØ-U9. Bit definitions for the A-register string are detailed in Table 7-1. The default is Ø.

Note

If only part of the string is given, it will be accepted right-justified by the system.

Example:

RESUME *RPGSII

In the example above, the A-register string has been omitted; default applies: No user options are taken.

Messages may be output by the system to the user's device during execution of an RPG II program. These messages are self-explanatory and usually include the procedure for continuing. Appendix F details the message format and provides examples.

Table 7-1. Run-time, A-Register Bit Definitions
Governing External User Indicators

1/abcdef

/=Enable
x=Disable

	Display all input	Display all output
Ø (default)	x	x
2	x	/
4	/	x
6	/	/

Control for debug operation

Ø (default)	x
1	/

External Conditioning User Indicators

	a	b			c			d		
	UØ	U1	U2	U3	U4	U5	U6	U7	U8	U9
Ø (default)	x	x	x	x	x	x	x	x	x	x
1	/	x	x	/	x	x	/	x	x	/
2		x	/	x	x	/	x	x	/	x
3		x	/	/	x	/	/	x	/	/
4		/	x	x	/	x	x	/	x	x
5		/	x	/	/	x	/	/	x	/
6		/	/	x	/	/	x	/	/	x
7		/	/	/	/	/	/	/	/	/

SECTION 8

INTERFACE TO MIDAS

USING MIDAS FOR KEYED INDEX OR DIRECT ACCESS FILES

If keyed-index or direct-access files are used in an RPG program, a template file must be created prior to program execution using the CREATK utility of MIDAS. (Multiple Index Data Access System. See Reference Guide, MIDAS for complete details.)

Creating the MIDAS Template

CREATK asks a series of questions about filename, filetype, key length, etc. The collating sequence for key length is that of the ASCII character set. Unless changes are to be made, a template file need be created only once.

Example: a template is to be created for the file DISKOUT. This file has is keyed-index with a key length of 6 bytes; the key type is ASCII. Data records are 64 words (128 bytes) long.

```
OK, CREATK
GO
MINIMUM OPTIONS? YES

FILE NAME? DISKOUT
NEW FILE? YES
DIRECT ACCESS? NO              Keyed index file.

DATA SUBFILE QUESTIONS

KEY TYPE: A                    A=ASCII (bytes), B=bits.
KEY SIZE = : B 6               B=bytes, W=words.
DATA SIZE = : 64

SECONDARY INDEX

INDEX NO.? _                   No secondary indices desired.
OK,
```

A sequential file (column 32=blank) can be output as an index file (column 32=D) via CHAIN. Refer to the IBM System/3 Model 1Ø disk system RPG II reference manual (SC21-75Ø4-6). The record size specified must be the same as that of the template.

Creating a MIDAS File

After the MIDAS file template is created, the MIDAS file can be built using an RPG II program. For a simple example of such a program see Appendix A.

Deleting a MIDAS File

A MIDAS file consists of a number of linked subfiles and segment directories. Thus, it cannot be deleted using the PRIMOS DELETE command. Instead, use the MIDAS utility KIDDEL as shown below.

```
OK, KIDDEL
GO

FILE NAME? DISKOUT
INDICES? ALL

OK,
```

SECTION 9

PROGRAM CODING SPECIFICATIONS

SPECIFICATIONS COMMON TO ALL FORMS

Prime RPG source program statements should be in ascending numeric sequence by columns 1 through 5. Statements out of sequence are flagged.

Columns	Entry	Specifications
1-2		PAGE Arrange the specification sheets in the following order and number them in ascending sequence: 1. Control. 2. File Description. 3. Extension 4. Line Counter. 5. Input. 6. Calculation. 7. Output.
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 Ø25 would be inserted between lines Ø2 and Ø3.
6		FORM TYPE
	H,F,E,L, I,C,O	A code identifying the specification sheet is required.
7		COMMENTS
	*	Enter an asterisk in each line used as a comment line. The control specification line (line Ø1) cannot be used as a comment line.

CONTROL STATEMENT (HEADER) SPECIFICATIONS

Columns	Entry	Specifications
6	H	Specifies Header sheet.
7-20		Columns 7-20 are not used by Prime RPG. (16-20 must be blank).
21		INVERTED PRINT-DATE NOTATION
	Blank	United States format.
	I	World Trade format.
	J	World Trade format (leading zero remains for zero balance).
	D	United Kingdom format.
		Note
		If column 21 is blank, the Control Specification may be omitted.
22-80		These columns are not used by Prime RPG.

FILE DESCRIPTION SPECIFICATIONS

Columns	Entry	Specifications
6	F	Specifies File Description sheet.
7-14		FILENAME
	XXXXXXXX	A filename must appear for each file. It can be from one to eight characters long. It must begin in column 7 with an alphabetic character or $ or #, followed by any combination of alphabetic and numeric characters or $ @ or # (embedded blanks are not allowed).
15		FILE TYPE
	I	Input
	O	Output
	U	Update
	D	Display
16		FILE DESIGNATION
	P	Primary
	S	Secondary
	C	Chained
	T	Table or Array
	D	Demand
	Blank	Display files and all output files except chained output files.
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, or U, and column 16 contains P, S, or R.

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 all Input files except table, array, chained, demand, and record address files.
19		FILE FORMAT
	F	Fixed length file format.
	U	Uncompressed. This entry must be made if the file contains (on input) or is to contain (on output) packed decimal or binary data. If specified, the device specified in columns 4Ø-46 must not be CONSOLE.
	Blank	A blank is interpreted as F.
2Ø-23		These positions are ignored.
24-27		RECORD LENGTH
	1-9999 1-512	Disk (default is 12Ø) Console (default is 8Ø) Printer (default is 132) Special (default is 9999)
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.
29-3Ø		FIELD LENGTH
	1-32	Indexed file: Length of record key.
		Maximum record key length is 32 characters.

31		Prime RPG ignores the contents of this column.
32		FILE ORGANIZATION
	I	Indexed organization.
	D	Direct file.
	Blank	Sequential.
33-34		OVERFLOW INDICATOR
	OA...OG, OV	Overflow indicator used to condition records in the file. The indicator specified is the one used.
	Blank	No overflow indicator used (default will advance to the top of the next line and continue printing.) WARNING This will cause a spooler generated title line and pagination to be printed at the top of every page.
35-38		KEY STARTING LOCATION
	1-9999	Record position in which the key field begins.
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 further described on line counter specifications.
40-46		DEVICE
	CONSOLE DISK SPECIAL	Enter the appropriate device code for the input/output unit used by the file specified in columns 7-14. CONSOLE cannot be used with files containing packed decimal or binary data.
47-52		Leave these positions blank.
53		CONTINUATION LINES/LABELS
	K	Continuation Lines: A continuation record specified for SPECIAL.
	Blank	Labels: Leave this position

		blank unless using continuation lines.
54-59		CONTINUATION LINE OPTION
	ASCII	Accepted but ignored.
	XXXXXX	Name of array to be used by user.
		NAME OF LABEL EXIT
	Blank	No SPECIAL device used.
	XXXXXX	Name of the user-written subroutine which will perform the I/O operation for a SPECIAL device.
6Ø-65		Leave these columns blank.
66		FILE ADDITION
	A	New records are added to the file.
	Blank	In all other instances.
		This column applies to sequential and indexed disk files.
67-7Ø		Prime RPG does not use these columns.
71-72		FILE CONDITION UØ-U9
	UØ-U9	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 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.
73-8Ø		Leave these positions blank.

EXTENSION SPECIFICATIONS

Columns	Entry	Specifications
6	E	Specifies Extension sheet.
7-10		Leave these columns blank.
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.
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.
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. Table names must begin with TAB; array names must not begin with TAB.
33-35		NUMBER OF ENTRIES PER RECORD Enter, right-justified, the number of entries on each table or array input record. This entry must be less than or equal to the number of entries per table or array in columns 36-38. These columns must contain an entry for compile and pre-execution time tables and arrays. These columns must be blank for execution time arrays. Maximum permissible entry is 4095.
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.

40-42 LENGTH OF ENTRY

Enter, right-justified, the length of each table or array entry. The maximum length is 255 for alphanumeric entries and 15 for numeric entries.

43 FIELD FORMAT

P — Data for the pre-execution time array/table named in columns 27-32 is in packed decimal format.

B — Data for the pre-execution time array/table named in columns 27-32 is in binary format.

Blank — Data for the array/table named in columns 27-32 is neither packed decimal nor binary format, or the array/table is loaded by input or calculation specifications.

Notes

1. This column must be blank if specifying an array/table to be loaded by input or calculation specifications.

2. Specification of P or B in column 43 requires that the file named in columns 11-18 must have been specified as having an 'uncompressed' file format (set U in column 19 of the File Description sheet).

3. If a TO file is specified, and P or B is specified in column 43, the TO file must have been specified as having an 'uncompressed' file format (set U in column 19 of the File Description sheet).

44 DECIMAL POSITIONS

Blank — Alphanumeric table or array.

Ø-9 — Number of positions to the right of the decimal.

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.
46-57		SECOND TABLE OR ARRAY
		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.
58-8Ø		COMMENTS
		Enter any information you wish to help you understand or remember what you are doing in each specification line.

LINE COUNTER SPECIFICATIONS

Columns	Entry	Specifications
6	L	Specifies Line Counter sheet.
7-14		FILENAME
	XXXXXXXX	Enter the name of a printer file for which you wish to specify a form size and overflow line.
15-17		LINE NUMBER/NUMBER OF LINES PER PAGE
	12-112	Number of lines available for printing on the printer form.
18-19		FORM LENGTH
		Enter FL to indicate the previous entry is the form length.
2Ø-22		LINE NUMBER/OVERFLOW LINE
	1-112	Number of the overflow line.
23-24		OVERFLOW LINE
		Enter OL to indicate the previous entry is the overflow line.
25-8Ø		These columns are not used by Prime RPG.

INPUT SPECIFICATIONS

Columns	Entry	Specifications
6	I	Specifies Input sheet.
7-14		FILENAME
	XXXXXXXX	Enter a valid Prime RPG filename for every Input and Update file your program uses.
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.
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.
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).

19-2Ø		RECORD IDENTIFYING INDICATOR, **
	Ø1-99	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.
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. Each field contains the following subdivisions:
		(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	This column must contain a C.

(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.

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 2Ø AND or OR lines in any combination may be used to describe the record identifying code.

42 Prime RPG does not use this column.

43 FIELD FORMAT

P Data for the field/array/table named in columns 53-58 is in packed decimal format.

B Data for the field/array/table named in columns 53-58 is in binary format.

Blank Data for the field/array/table named in columns 53-58 is in neither packed decimal nor binary format.

Note

Specification of P or B in column 43 requires that the file being used for input must have been specified as having an 'uncompressed' file format (set U in column 19 of the File Description sheet).

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.

52 DECIMAL POSITION

Blank Alphanumeric field.

Ø-9 The number of decimal positions in the

numeric field named in columns 53-58.

This column must contain an entry for numeric fields.

53-58 FIELD NAME

These columns must contain one of the following:

XXXXXX

1. A valid RPGII field name for each field defined in Field Location. (Prime RPG field names are governed by rules for valid file names, but are limited to 6 characters.)

2. An array name or array element.

59-6Ø 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.

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 or Update files with match fields. When you have just one Input or Update file with match fields, this entry causes only sequence checking.

63-64 FIELD RECORD RELATION

Ø1-99 Record identifying indicator assigned to a record type.

L1-L9 Control level indicator previously used.

MR Matching record indicator

U0-U9 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.

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 alphanumeric, an indicator can only be specified in Zero or Blank (columns 69-70).

71-80 These positions should be blank.

CALCULATION SPECIFICATIONS

Columns	Entry	Specifications
6	C	Specifies Calculation sheet.
7-8		CONTROL LEVEL
	Blank	Operation done at detail time.
	LØ	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 or OR lines are allowed to condition an operation. Entries must be in the order listed.
9-17		INDICATORS
		Enter one to three indicators. Any indicators except 1P and LØ 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.
18-27		FACTOR 1
28-32		OPERATION
		Enter an operation code, left justified.
33-42		FACTOR 2
		Factor 1 and Factor 2 may contain the following entries:

1. Name of any field that has been defined.

2. Alphanumeric or numeric literal.

3. Subroutine, table or array name, or array element.

4. Label for a TAG, BEGSR, or ENDSR opertion (Factor 1) or a label for a GOTO or EXSR operation (Factor 2).

5. Filename for a CHAIN, DEBUG, DSPLY, READ, SETLL, or FORCE operation (Factor 2).

43-48 RESULT FIELD

Enter the name of the field, table, 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.

49-51 FIELD ENTRY

Blank	Field defined elsewhere.
1-255	Result field length.

Maximum length of a numeric field is 15 digits; maximum length of an alphaumeric field is 255 characters. Entry must be right justified.

52 DECIMAL POSITION

Blank	Alphanumeric field or numeric field described elsewhere.
Ø-9	Number of decimal places in a numeric result field.

53

Blank	Do not round
H	Enable half-adjust (rounding)

Half-adjust can be used only with arithmetic operations. Half-adjust cannot be used with an MVR operation or with a division followed by an

MVR operation. Half-adjust is valid only for numeric fields with greater than zero decimal positions in the result field.

Operationally, the half-adjust option performs a rounding operation on the result field. To achieve this, +5 or -5 (depending upon the sign of the result) is added to the digit immediately to the right of the rightmost resulting decimal position.

54-59 RESULTING INDICATORS

Enter any of the following indicators: Ø1-99, 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, or COMP 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.

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 Operation: 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.

60-80 COMMENTS

Enter any meaningful information you wish to help you understand or remember what you are doing in each specification line.

OUTPUT SPECIFICATIONS

Columns	Entry	Specifications
6	O	Specifies Output sheet.
7-14		FILENAME
	XXXXXXXX	Enter a valid RPGII filename for each output and update file used by your program. Each filename need be specified only once, on the first line describing that file.
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.
15		TYPE
	H	Heading records.
	D	Detail records.
	T	Total records.
	E	Exception records.
16-22		These columns are used in the following ways:
		(ADD A RECORD)
	ADD	Enter ADD in columns 16-18 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 Specifications for the file to which a record is added.
		(FETCH OVERFLOW)
	F	Enter F in column 16 to fetch the overflow routine when overflow occurs, before the usual time in the cycle.
		(SPACE/SKIP)
	Blank	If columns 17-22 are blank single spacing occurs after each line is printed.

Ø-3 — Enter a number in either column 17 or column 18, or both, to indicate the number of lines spaced before or after a line is printed.

Blank — Blanks in columns 19-22 indicates no skipping.

Ø1-99 — Enter one of the 2-digit numbers 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.

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 2Ø).

32-37 FIELD NAME

Enter one of the following to name every field written out:

XXXXXX

1. Any field name previously defined in this program.
2. A previously defined table name, array name, or array element.
3. The special words, PAGE, PAGE1, PAGE2, UDATE, UDAY, UMONTH, UYEAR, RPGREV. RPGREV is preset to the current RPG revision number. This item is referenced as a 4-digit numerical value with 2 digits following the decimal point (e.g., 16.ØØ).

These columns must be blank if a constant is entered on columns 45-7Ø of the line. If an entry is made under Field Name, columns 7-22 must be blank.

38 EDIT CODES

Use column 38 for the following:

1. Suppress leading zeroes 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.

The following table summarizes EDIT CODE options:

COMMAS	ZERO BALANCES TO PRINT	NO SIGN	CR	-	X=REMOVE PLUS SIGN
YES	YES	1	A	J	Y=DATE FIELD EDIT
YES	NO	2	B	K	
NO	YES	3	C	L	Z=ZERO SUPPRESS
NO	NO	5	D	M	

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 alphanumeric 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.

4Ø-43 END POSITION IN OUTPUT RECORD

Columns 4Ø-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.

44 FIELD FORMAT

P — Data for the field/array/table named in columns 32-37 is in packed decimal format.

B — Data for the field/array/table named in columns 32-37 is in binary format.

Blank — Data for the field/array/table named in columns 32-37 is in neither packed decimal nor binary format.

Notes

1. Specification of P or B in column 44 requires that the file being used for output must have been specified as having an 'uncompressed' file format (set U in column 19 of the File Description sheet).

2. Specification of P or B in column 44 requires that column 38 and columns 45-70 be blank.

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 sign).

SECTION 10

COMPILER REFERENCE

PRIME RPG II COMPILER REFERENCE

All parameters are preceded by a dash, "-", in the command line. Parameters that are the PRIME-supplied default parameters (i.e., those that need not be included) are indicated. The System Administrator may have changed the defaults; if so, the programmer should obtain a list of the installation-specific defaults. (See figure 10-1).

BANNER

Causes a column index banner to be printed preceding each non-comment source line. This is useful for checking that items are in their proper columns. See NOBANNER.

B[INARY] { pathname / YES / NO }

Specifies the binary (object) output file. If pathname is given, then that will be the name of the binary file. If YES is used, the name of the binary file will be B_PROGRAM (where PROGRAM is the source filename). If NO is used, then no binary file is created. Omitting the parameter is equivalent to the inclusion of -BINARY YES. (See Table 10-1.)

ERRTTY Default

Prints error messages at the user terminal. The normal system default causes each statement containing an error to be printed at the user terminal. This feature is especially useful when a corrected program is being recompiled, to confirm that the errors have been properly corrected. See NOERRTTY.

I[NPUT] pathname

Specifies the pathname of the input source program (See Table 10-1). This parameter must not be used if the source filename immediately follows the RPG command; otherwise, it must be included in the parameter list. See SOURCE.

Default A Register Bit			Reset (0)	Set (1)
0	0	1	NOSEQCHR	SEQCHK
0	0	2	NOBANNER	BANNER
	0	3	NOOBDATA	OBDATA
	0	4	XREF	NOXREF
3	0	5		
	1	6	NOSTATUS	STATUS
	1	7	NOERRTTY	ERRTTY
7	1	8		INPUT
	1	9		SOURCE
	1	10		
7	1	11		LISTING
	1	12		
	1	13		
7	1	14		
	1	15		BINARY
	1	16		

Figure 10-1. Bit-Mnemonic Correspondence (A Register)

Table 1Ø-1. Compiler File Specifications

Compiler Mnemonics	INPUT or SOURCE	LISTING	BINARY
pathname	Looks for file named pathname as source file.	Opens file named pathname as listing file.	Opens file named pathname as (object) file.
YES	Not applicable.	Uses default filename for listing file. L_PROGRAM.	Uses default file name for binary file, B_PROGRAM
NO	Not applicable.	No listing file..	No listing file.
TTY	Compiler will compile program as entered from the terminal.	Print listing on user terminal.	Not applicable.
SPOOL	Not applicable.	Spool listing directly to line printer.	Not applicable.
option not invoked	Source filename must be first option after FTN command.	Same as YES.	Same as YES.

To use other peripheral devices such as magnetic tape, card reader, or paper tape punch/reader for file location, see Table 10-2 for A-register settings.

L[ISTING] { pathname | YES | NO | TTY | SPOOL }

Specifies the listing device/filename:

pathname	Opens this file for the listing.
YES	Uses the default name for the listing file L_PROGRAM (where PROGRAM is the source).
NO	No listing file is created.
TTY	The listing file is printed on the user terminal.
SPOOL	The listing file is spooled directly to the line printer.

If this parameter is omitted from the parameter list, it is equivalent to the -LISTING YES parameter inclusion (i.e., a listing file is created with the name L_PROGRAM).

NOBANNER Default

Suppresses the printing of a column index banner preceding each non-comment source line.
See Banner.

NOERRTTY

No Terminal Error Messages. Suppresses the printing of error messages on the users terminal. See ERRTTY.

NOOBDATA Default

Suppress octal listing of the generated object data with the source listing. See OBDATA.

NOSEQCHK Default

Suppresses checking of columns 3-5 of the source program for proper sequencing. See SEQCHK.

NOSTATUS

Suppresses printing, at user terminal, of current RPG status (H,F,E,L,I,C, or Ø). See STATUS

NOXREF

Suppresses printing of the cross reference (concordance) listing following the program listing. See XREF

OBDATA

Causes an octal listing of the generated object data to be included within the source listing. See NOOBDATA.

SEQCHK

Causes columns 3-5 of the source program to be checked for proper sequencing. See NOSEQCHK.

S[OURCE]
Same as I[NPUT] (See INPUT).

STATUS Default

Causes the current RPG status (H,F,E,L,I,C, or O) to be listed at the user terminal. See NOSTATUS.

XREF Default

Enable Concordance. Appends a concordance cross-reference) listing to the end of the program listing. The concordance includes all symbols in the program unit. See NOXREF.

EXPLICIT SETTING OF THE A REGISTER

If you will not be using the paper tape punch/reader, card punch/reader or magnetic tape for I/O devices at compilation time you need not read this section.

Operation

RPG pathname [1/a-register]

pathname is the pathname of the RPG II source file. a-register is the value of the A register. If the default value is used, this parameter may be omitted.

The default value of the A register is:

'3777 (binary = ØØØØØ1111111111)

Input file on disk
listing file on disk
Binary file on disk
Print error messages at user terminal
Print current status at user terminal
Print cross reference listing

Note

The A register is actually set to Ø for the default values. The RPG compiler converts an A-register value of Ø to '3777; all other A-register values are interpreted as is.

Spaces should be used to separate components of the command line. The bit values corresponding to the mnemonic parameters are given in Table 1Ø-2.

Input/Output Specifications

Additional devices are accessible to users explicitly setting the A register. I/O is specified by the A-register setting as:

Type	Bits
Input (source)	8-1Ø
Listing	11-13
Binary (object)	14-16

The settings corresponding to I/O files and devices are given in Table 1Ø-3.

Table 1Ø-2. A-Register Bit Correspondences of Parameter Mnemonics (Prime-supplied defaults are indicated)

A (x,y) = Ø (or 1): the mnemonic parameter causes the value of bits x and y in the A register to be Ø (or1).

BANNER	A(2)=1
B[INARY]	(14,15,16)=object file definition (See Table 1Ø-3); PRIMOS BINARY command
ERRTTY	A(F)=1;default
I[NPUT]	A(8,9,1Ø)=input file definition (See Table 1Ø-3)
L[ISTING]	A(11,12,13)=listing file definition (See Table 1Ø-3);PRIMOS LISTING command
NOBANNER	A(2)=Ø;default
NOERRTTY	A(7)=Ø
NOOBDATA	A(3)=Ø; default
NOSEQCHK	A(1)=Ø; default
NOSTATUS	A(6)=Ø
NOXREF	A(4)=1
OBDATA	A(3)=1
SEQCHK	A(1)=1
S[OURCE]	A(8,9,1Ø)=input file definition (See Table 1Ø-3); same as I[NPUT]
STATUS	A(6)=1; default
XREF	A(4)=Ø; default

Table 10-3. Bit/Device Correspondences

Bits	Octal	Device	Mnemonic Parameter
000	0	None	NO
001	1	User terminal	TTY
010	2	Paper tape reader/punch	---
011	3	Reserved for card reader/punch	---
100	4	Reserved for line printer	---
101	5	Magnetic tape, unit 1	---
110	6	Diskette	---
111	7	Disk (PRIMOS file system)	---

Disk (PRIMOS file system)

	Defaults	
Source	7	File System
Listing	7	File System
Binary	7	File System

File Unit Usage

Three file units may be active during a compilation:

File Type	PRIMOS file unit
Source	1
Listing	2
Object	3

If the disk is specified as the device for the listing and/or object file causes these files to be opened on the disk with default names constructed as follows: if the source file has the pathname

[MFD]>UFD1>. . . .>filename

the listing file and the object file are opened as L_filename and B_filename respectively in the current UFD. Upon completion of the RPG command all files are closed and command returns to PRIMOS.

If the user desires the listing or binary files to have names other than the default names, and/or to be opened in UFD's other than the current one, this must be done prior to invoking the RPG command.

The PRIMOS Commands

LISTING filename-2 opens a listing file with the specified name filename-2 (in the current UFD) on PRIMOS file unit 2. This inhibits RPG from opening a default listing file.

Note

Unless bits 11-13 of the A-register are set to '7, nothing is written into this file.

The listing output of more than one source file can be concatenated if all listings are generated prior to closing the listing file.

For example:

```
LISTING filename
   .
   .
   .
RPG source-1 1/aregister
   .
   .
   .
RPG source-n 1/aregister
   .
   .
   .
CLOSE ALL
```

(System responses are not printed in this example.)

The listing file, filename, will contain the concatenation of all listing outputs from source-1,..., source-n (for those compilations wherein listings were specified).

BINARY filename-3 opens a binary (object) file with the specified name filename-3 (in the current UFD) on PRIMOS file unit 3. This inhibits RPG from opening a default object file.

Note

> The default value of bits 14-16 of the A-register is '7-disk file system. If not using the default A-register values be sure to set bits 14-16 to '7 or nothing will be written into the object file. Object files can also be concatenated in the same manner as listing files.

If the BINARY or LISTING commands are used prior to RPG to establish non-default file, then RPG does not close these files upon completion.

After RPG returns command to PRIMOS, these files shoud be closed by the user by:

```
                  2           3
CLOSE     filename-2  filename-3

 or

CLOSE     ALL
```

SECTION 11

PACKED DECIMAL AND BINARY DATA TYPES

OVERVIEW

Packed decimal and binary data types in files require the specification of sequential files which do not use compression control characters. This specification is called 'uncompressed'.

An 'uncompressed' file is comprised of fixed length records separated by a word containing the .NL. and .NULL. characters ('1Ø5ØØØ). If fewer words than the specified (or default) record size are written, the remainder of the record is blank filled ('ØØ4Ø) Default record sizes are discussed in Section 9. Uncompressed files are written by the subroutine O$ADØ8, and are read with PRWF$$. The keys used for PRWF$$ are K$READ+K$POSR; POS has a value of ØØØØØ1.) See the Subroutine Reference Guide (PDR3621) for details on these subroutines.

SIZE CONSIDERATIONS

The following is implied for packed decimal and binary fields (all lengths are in bytes):

PACKED DECIMAL		BINARY	
Unpacked Decimal Length LOE	Implied Packed Decimal Length IL	Decimal Length LOE	Implied Binary Length IL
1	1	4	2
2-3	2	9	4
4-5	3		
6-7	4		
8-9	5	LOE - length of entry	
1Ø-12	6	IL - implied length	
12-13	7		
14-15	8		

Extension Specification

When defining an array/table on the extension specification sheet, the "length of entry" (columns 4Ø-42 and 52-54) is specified as the decimal length of the entry.

Input Specification

When specifying a packed decimal or binary array/table on the input specification sheet, the entries made in the FROM-TO columns (44-47, 48-51) represent the number of entries being obtained from the record times the "implied length". The "implied length" is computed from "length of entry" on the extension specification sheet.

The following is the sequence of events:

1. Obtain "length of entry" entered on extension specification sheet; let this value be assigned to LOE.

2. Determine "implied length" based upon the entry in column 43 of the input specification sheet and assign to IL as follows:

   If column 43=P then IL=(LOE/2)+1
   If column 43=B then IL=(LOE/3)+1

3. Let N equal the number of entries to be obtained from the record.

4. Let WIDTH=N*IL

Where WIDTH = ((TO - FROM)+1)*N

When specifying a packed decimal or binary field, (or an array element) on the input specification sheet, IL is specified; i.e. the packed decimal or binary length is entered.

Output Specification

When specifying a packed decimal or binary array/table on the output specification sheet, the end position in the output record (columns 40-43) must allow for the number of entries to be placed in the record (let this value be assigned to NE) times the implied length (IL). Thus, if the previous end position were 100, then the end position (EP) is determined by:

EP=(NE*IL)+100

When specifying a packed decimal or binary field (or an array element) on the output specification sheet,

IL + previous end position

is specified; i.e. the packed decimal or binary length is entered.

ADDITIONAL INFORMATION

WARNING

> Once a file has been created in 'uncompressed' format, it may not be edited with ED. It is possible to examine an uncompressed file with ED, but changes may not be made to the file in a successful manner since the editor only operates on compressed ASCII files.

The last record written to an 'uncompressed' file should have a /* in columns 1-2 to indicate end-of-file. If this is not done, and end-of-file is detected, an error message is printed. (Typing an S followed by a carriage return will allow the program to continue its execution.)

If a numeric field is read or created in unpacked format, and is to be written out in packed decimal format, the field size may change when read in later by another program.

Example:

Program 1		Program 2
Unpacked Length	Packed Length	Unpacked Length
4	3	5
5	3	5

If a numeric field is read or created in unpacked format and is to be written out in binary format, the field size must have been specified as either 4 or 9.

ERROR PROCESSING

During execution, the following messages are produced for invalid operations on packed decimal or binary fields:

**** ERROR. ATTEMPT TO CONVERT NON-NUMERIC DATA TO PACKED DECIMAL FROM FIELD NEAR LOCATION .....

explanation: the field or array/table entry defined near location ..... contains non-numeric data.

**** ERROR. ATTEMPT TO CONVERT PACKED DECIMAL DATA TO NUMERIC FIELD FROM POSITION ..... OF FILE ..... IS UNSUCCESSFUL

explanation: the field or array/table entry contains a sign other than :15 (D), :14 (C), or :17 (F) at the specified position within the input record.

**** ERROR. ATTEMPT TO CONVERT NON-NUMERIC DATA TO BINARY FROM FIELD NEAR LOCATION .....

explanation: the field or array/table entry defined near location ..... contains non-numeric data.

**** ERROR. ATTEMPT TO CONVERT BINARY DATA TO NUMERIC FIELD FROM POSITION ..... OF FILE ..... IS UNSUCCESSFUL

explanation: if the specified width of the binary field is 4, then the binary field contains a number not within the range -9999..9999 or if the specified width of the binary field is 9, then the binary field contains a number not within the range -999999999..999999999.

In all of the above instances, program execution may be restarted by typing an S followed by a carriage-return. See Appendix F for more details.

APPENDIX A

SAMPLE RPG II PROGRAMS

SAMPLE PROGRAM 1

The program SAM1 loads 100 records into an index disk file. The records are calculated in a program loop. The coding specifications for SAM1 are shown in Figure A-1; a program listing is shown in Figure A-2.

The file DISKOUT is a MIDAS file; a template must be created for it prior to executing SAM1. The template is built with the CREATK utility described in Section 8.

SAMPLE PROGRAM 2

The program SAM2, which is executed after SAM1, prints out the indexed file, verifying the load procedure. The coding specifications for SAM2 are shown in Figure A-3; a program listing is shown in Figure A-4.

PROGRAM DEVELOPMENT

The programs SAM1 and SAM2 were entered at the terminal using the Editor (Section 4). Compilation, loading, and execution of these programs are discussed in Appendix B.

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

Date
Program
Programmer

Punching Instruction: Graphic / Punch — Page (1 2) — Program Identification (75 76 77 78 79 80)

Line	Form Type	Filename / Comment	File Type	File Format	Block Length	Record Length	Length of Record Address Field	Record Address Type	Type of File Organization	Key Field Starting Location	Device
01	F										
02	F										
03	F	**									
04	F	* *									
05	F	* THIS PROGRAM - *									
06	F	* *									
07	F	* 1. LOADS 1ØØ RECORDS TO AN INDEXED FILE. *									
08	F	* *									
09	F	* 2. READS ONE RECORD FROM FILE SAM2 FOR *									
10	F	* INPUT. *									
11	F	* *									
12	F	* 3. CREATES THE OUTPUT DATA USING A *									
13	F	* LOOP IN THE CALCULATION SPECIFICATIONS. *									
14	F	* *									
15	F	* 4. USES KEYS FROM ØØØØØ5 THROUGH ØØØ5ØØ *									
	F	* IN INCREMENTS OF 5. *									
	F	* *									
	F	* 5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2 *									
	F	* TO VERIFY THAT THE FILE WAS PROPERLY *									
	F	* LOADED. *									
01	F	**									
02	F	SAM2	IP	F	96	96					DISK
03	F	DISKOUT	O	F	256	128	Ø6	A	I	1	DISK
04	F	PRINT	O	F	96	96					PRINTER
05	F										

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

Date
Program
Programmer

Punching Instruction: Graphic / Punch — Page (1 2) — Program Identification (75 76 77 78 79 80)

Line	Form Type	Filename	Sequence	Resulting Indicator	Field Location From	Field Location To	Field Name
01	I						
02	I	SAM2	NS	Ø1	1	1	NODATA
03	I						

Figure A-1. SAM1 Coding Specifications
(Sheet 1 of 2)

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS

Line	Form Type	Control Level (L0-L9, LR)	Indicators: And/Not	Factor 1	Operation	Factor 2	Result Field	Field Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus	Minus	Zero or Blank	Compare: High 1>2	Low 1<2	Equal 1=2	Comments
02	C		01		Z-ADD0		COUNT	60									
03	C		01		Z-ADD0		RECNBR	30									
04	C			REPEAT	TAG												
05	C		01	COUNT	ADD	5	COUNT										
06	C		01	RECNBR	ADD	1	RECNBR										
07	C		01	COUNT	COMP	505										02	
08	C		01N02		EXCPT												
09	C		01N02		GOTO	REPEAT											
10	C				SETON										LR		
11	C	LR		RECNBR	SUB	1	RECNBR										
12	C																

REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

Line	Form Type	Filename	Type (H/D/T)	Space: Selector / Before / After	Skip: Before / After	Output Indicators: And/Not	Field Name	Zero Suppress (Z) / Blank After (B)	End Position in Output Record	Packed Field (P)	Constant or Edit Word	Sterling Sign Position
02	O	PRINT1	T	2	01	LR						
03	O								18		'SAMPLE PROGRAM HAS'	
04	O								25		'LOADED'	
05	O						RECNBRZ		29			
06	O								37		'RECORDS'	
07	O								59		'INTO AN INDEXED FILE.'	
08	O		T	2		LR						
09	O								21		'KEYS ARE IN ASCENDING'	
10	O								42		'SEQUENCE STARTING AT'	
11	O								64		'000006 AND INCREASING'	
12	O								84		'IN INCREMENTS OF 5.'	
13	O		T		01	LR						
14	O								21		'SAMPLE PROGRAM 2 WILL'	
15	O								44		'PRINT FROM THE INDEXED'	
	O								65		'FILE TO SHOW THAT'	
	O								86		'WAS PROPERLY LOADED.'	
	O	DISKOUT E				01N02						
	O						COUNT		6			
	O								94		'RECORD NUMBER'	
01	O						RECNBR		128			
02	O											

Figure A-1. SAM1 Coding Specifications
(Sheet 2 of 2)

```
*SAMPLE PROGRAM #1
*
     F**************************************************
     F*                                                *
     F* THIS PROGRAM -                                 *
     F*                                                *
     F*   1. LOADS 100 RECORDS TO AN INDEXED FILE.     *
     F*                                                *
     F*   2. READS ONE RECORD FROM FILE SAM2 FOR       *
     F*      INPUT.                                    *
     F*                                                *
     F*   3. CREATES THE OUTPUT DATA USING A           *
     F*      LOOP IN THE CALCULATION SPECIFICATIONS.   *
     F*                                                *
     F*   4. USES KEYS FROM 000005 THROUGH 000500      *
     F*      IN INCREMENTS OF 5.                       *
     F*                                                *
     F*   5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2    *
     F*      TO VERIFY THAT THE FILE WAS PROPERLY      *
     F*      LOADED.                                   *
     F*                                                *
     F**************************************************
     FSAM2    IP  F  96  96           DISK
     FDISKOUT O   F 256 128 06AI    1 DISK
     FPRINT1  O   F  96  96           PRINTER
*
     ISAM2    NS  01
     I                                        1   1 NODATA
*
     C   01               Z-ADD0        COUNT   60
     C   01               Z-ADD0        RECNBR  30
     C           REPEAT   TAG
     C   01      COUNT    ADD  5        COUNT
     C   01      RECNBR   ADD  1        RECNBR
     C   01      COUNT    COMP 505                  02
     C   01N02            EXCPT
     C   01N02            GOTO REPEAT
     C                    SETON                   LR
     CLR         RECNBR   SUB  1        RECNBR
*
     OPRINT1  T  204   LR
     O                                     18 'SAMPLE PROGRAM HAS'
     O                                     25 'LOADED'
     O                         RECNBRZ     29
     O                                     37 'RECORDS'
     O                                     59 'INTO AN INDEXED FILE.'
     O       T  2     LR
     O                                     21 'KEYS ARE IN ASCENDING'
     O                                     42 'SEQUENCE STARTING AT'
     O                                     64 '000005 AND INCREASING'
     O                                     84 'IN INCREMENTS OF 5.'
     O       T     01 LR
     O                                     21 'SAMPLE PROGRAM 2 WILL'
     O                                     44 'PRINT FROM THE INDEXED'
     O                                     65 'FILE TO SHOW THAT IT'
     O                                     86 'WAS PROPERLY LOADED.'
     ODISKOUT E        01N02
     O                         COUNT       6
     O                                     94 'RECORD NUMBER'
     O                         RECNBR    128
```

Figure A-2. SAM1 Program Listing

REPORT PROGRAM GENERATOR FILE DESCRIPTION SPECIFICATIONS

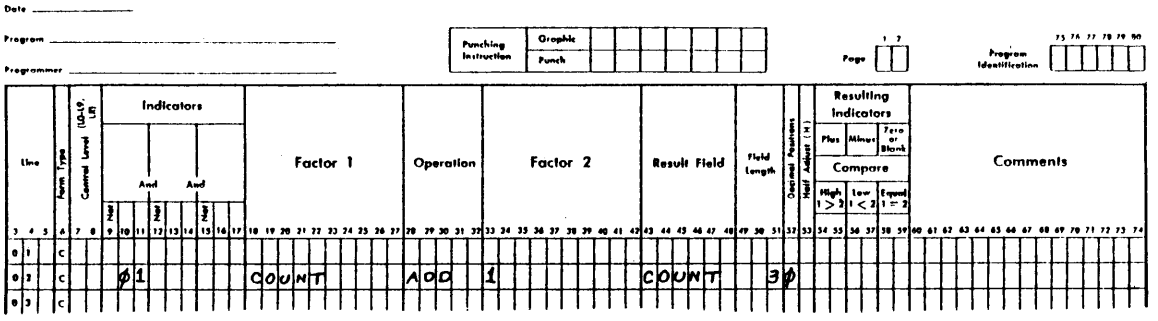
```
     F***********************************************
     F*                                             *
     F* THIS PROGRAM -                              *
     F*                                             *
     F*    1. MUST BE PRECEDED BY SAMPLE PROGRAM 1  *
     F*       WHICH LOADS AN INDEXED FILE.          *
     F*                                             *
     F*    2. READS AN INDEXED FILE SEQUENTIALLY.   *
     F*                                             *
     F*    3. USES A BLOCK LENGTH FOR DISK WHICH    *
     F*       IS DIFFERENT FROM THAT USED FOR       *
     F*       LOADING THE FILE IN SAMPLE PROGRAM 1. *
     F*                                             *
     F*    4. COUNTS THE NUMBER OF RECORDS READ SO  *
     F*       THAT THE USER CAN QUICKLY VERIFY THAT *
     F*       1ØØ RECORDS WERE LOADED.              *
     F*                                             *
     F***********************************************
Ø1   FDISKOUT IPE F 512 128 Ø6AI     1 DISK
Ø2   FPRINT2  O   F  96  96     OF     PRINTER
Ø3   F
```

REPORT PROGRAM GENERATOR INPUT SPECIFICATIONS

```
Ø1   I
Ø2   IDISKOUT NS  Ø1   1 CØ
Ø3   I                                         1   6 KEY
Ø4   I                                        82  94 DESC
Ø5   I                                       126 128ØRECNBR
Ø6   I
```

Figure A-3. SAM2 Coding Specifications
(Sheet 1 of 2)

REPORT PROGRAM GENERATOR CALCULATION SPECIFICATIONS



REPORT PROGRAM GENERATOR OUTPUT-FORMAT SPECIFICATIONS

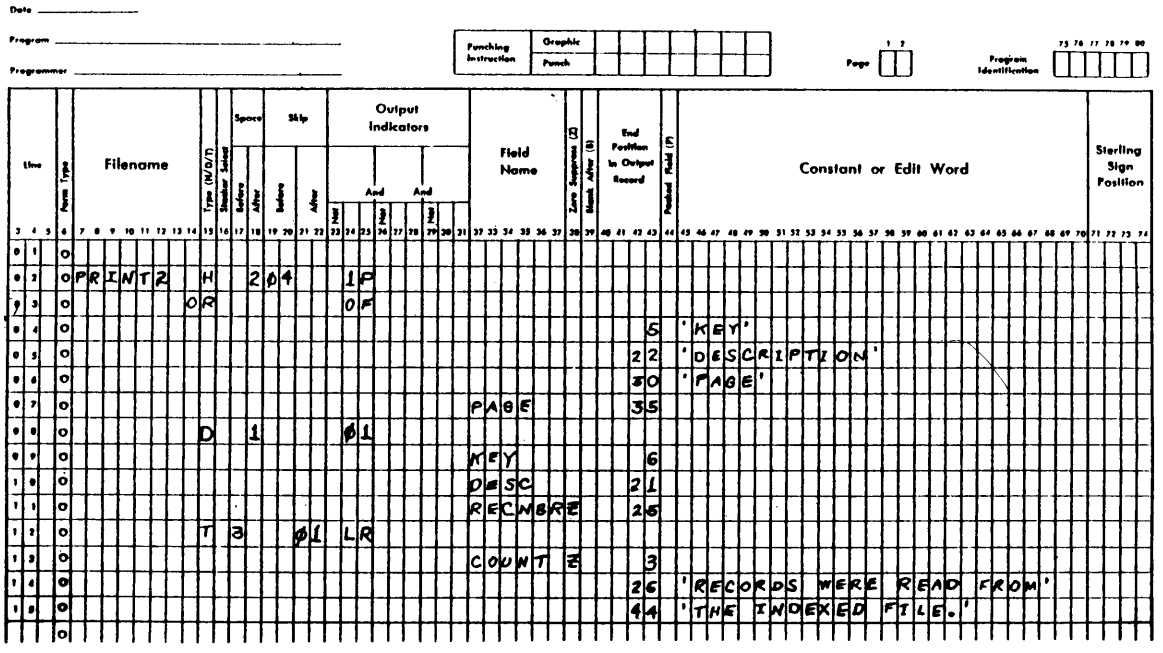


Figure A-3. SAM2 Coding Specifications
(Sheet 2 of 2)

```
*SAMPLE PROGRAM #2
*
     F*************************************************
     F*                                               *
     F* THIS PROGRAM -                                *
     F*                                               *
     F*   1. MUST BE PRECEDED BY SAMPLE PROGRAM 1     *
     F*      WHICH LOADS AN INDEXED FILE.             *
     F*                                               *
     F*   2. READS AN INDEXED FILE SEQUENTIALLY.      *
     F*                                               *
     F*   3. USES A BLOCK LENGTH FOR DISK WHICH       *
     F*      IS DIFFERENT FROM THAT USED FOR          *
     F*      LOADING THE FILE IN SAMPLE PROGRAM 1.    *
     F*                                               *
     F*   4. COUNTS THE NUMBER OF RECORDS READ SO     *
     F*      THAT  THE USER CAN QUICKLY VERIFY THAT   *
     F*      100 RECORDS WERE LOADED.                 *
     F*                                               *
     F*************************************************
     FDISKOUT IPE F 512 128 06AI     1 DISK
     FPRINT2  O   F  96  96     OF     PRINTER
*
     IDISKOUT NS  01   1 C0
     I                                        1   6 KEY
     I                                       82  94 DESC
     I                                      126 1280RECNBR
*
     C   01       COUNT     ADD  1          COUNT   30
*
     OPRINT2  H  204   1P
     O        OR       OF
     O                                        5 'KEY'
     O                                       22 'DESCRIPTION'
     O                                       30 'PAGE'
     O                          PAGE         35
     O         D  1     01
     O                          KEY          6
     O                          DESC        21
     O                          RECNBRZ     25
     O         T 3    01 LR
     O                          COUNT Z      3
     O                                       26 'RECORDS WERE READ FROM'
     O                                       44 'THE INDEXED FILE.'
```

Figure A-4. SAM2 Program Listing

APPENDIX B

SAMPLE PROGRAM DEVELOPMENT

COMPILATION

The two sample programs, SAM1 and SAM2, must first be compiled with the RPG compiler.

```
OK, RPG SAM1
 F
 I
 C
 O
ØØØ WARNINGS, ØØØ ERRORS [RPG REV 16.1Ø]

OK, RPG SAM2
 F
 I
 C
 O
ØØØ WARNINGS, ØØØ ERRORS [RPG REV 16.1Ø]

OK,
```

The listing files generated by the compiler may be examined with SLIST.

```
OK, SLIST L SAM1
GO
   [RPG REV 16.10] SOURCE:  SAM1                                  13:50:11  12/11/78
(0001)  *SAMPLE PROGRAM #1
(0002)  *
(0003)       F**************************************************
(0004)       F*                                                *
(0005)       F* THIS PROGRAM -                                 *
(0006)       F*                                                *
(0007)       F*   1. LOADS 100 RECORDS TO AN INDEXED FILE.    *
(0008)       F*                                                *
(0009)       F*   2. READS ONE RECORD FROM FILE SAM2 FOR       *
(0010)       F*      INPUT.                                    *
(0011)       F*                                                *
(0012)       F*   3. CREATES THE OUTPUT DATA USING A           *
(0013)       F*      LOOP IN THE CALCULATION SPECIFICATIONS.   *
(0014)       F*                                                *
(0015)       F*   4. USES KEYS FROM 000005 THROUGH 000500      *
(0016)       F*      IN INCREMENTS OF 5.                       *
(0017)       F*                                                *
(0018)       F*   5. SHOULD BE FOLLOWED BY SAMPLE PROGRAM 2    *
(0019)       F*      TO VERIFY THAT THE FILE WAS PROPERLY      *
(0020)       F*      LOADED.                                   *
(0021)       F*                                                *
(0022)       F**************************************************
(0023)       FSAM2    IP  F  96  96            DISK
(0024)       FDISKOUT O   F 256 128 06AI     1 DISK
(0025)       FPRINT1  O   F  96  96            PRINTER
(0026)  *
(0027)       ISAM2    NS  01
(0028)       I                                        1   1 NODATA
(0029)  *
(0030)       C   01               Z-ADD0         COUNT   60
(0031)       C   01               Z-ADD0         RECNBR  30
(0032)       C           REPEAT   TAG
(0033)       C   01      COUNT    ADD  5         COUNT
(0034)       C   01      RECNBR   ADD  1         RECNBR
(0035)       C   01      COUNT    COMP 505                    02
(0036)       C   01N02            EXCPT
(0037)       C   01N02            GOTO REPEAT
(0038)       C                    SETON                     LR
(0039)       CLR         RECNBR   SUB  1         RECNBR
```

```
(Ø04Ø)  *
(Ø041)       OPRINT1  T  2Ø4   LR
(Ø042)       O                                    18 'SAMPLE PROGRAM HAS'
(Ø043)       O                                    25 'LOADED'
(Ø044)       O                          RECNBRZ   29
(Ø045)       O                                    37 'RECORDS'
(Ø046)       O                                    59 'INTO AN INDEXED FILE.'
(Ø047)       O        T  2     LR
(Ø048)       O                                    21 'KEYS ARE IN ASCENDING'
(Ø049)       O                                    42 'SEQUENCE STARTING AT'
(Ø05Ø)       O                                    64 'ØØØØØ5 AND INCREASING'
(Ø051)       O                                    84 'IN INCREMENTS OF 5.'
(Ø052)       O        T     Ø1 LR
(Ø053)       O                                    21 'SAMPLE PROGRAM 2 WILL'
(Ø054)       O                                    44 'PRINT FROM THE INDEXED'
(Ø055)       O                                    65 'FILE TO SHOW THAT IT'
(Ø056)       O                                    86 'WAS PROPERLY LOADED.'
(Ø057)       ODISKOUT E        Ø1NØ2
(Ø058)       O                          COUNT      6
(Ø059)       O                                    94 'RECORD NUMBER'
(Ø06Ø)       O                          RECNBR   128
```

```
            ************************
            *   CROSS REFERENCE    *
            ************************
```

```
Ø         [LITERAL------------>ØØ2172]   (Ø03Ø)  (Ø031)
1         [LITERAL------------>ØØ2177]   (Ø034)  (Ø039)
5         [LITERAL------------>ØØ22Ø4]   (Ø033)
5Ø5       [LITERAL------------>ØØ221Ø]   (Ø035)
COUNT     [FIELD--------------->ØØ2217]   (Ø03Ø)  (Ø033)  (Ø033)  (Ø035)  (Ø058)
DISKOUT   [FILE---------------->ØØØ276]   (Ø024)  (Ø057)
NODATA    [FIELD--------------->ØØ2226]   (Ø028)
PRINT1    [FILE---------------->ØØØ6Ø3]   (Ø025)  (Ø041)
RECNBR    [FIELD--------------->ØØ2255]   (Ø031)  (Ø034)  (Ø034)  (Ø039)  (Ø039)
                                          (Ø044)  (Ø06Ø)
REPEAT    [LABEL--------------->ØØ2263]   (Ø032)  (Ø037)
SAM2      [FILE---------------->ØØØØ11]   (Ø023)  (Ø027)
```

ØØØ WARNINGS, ØØØ ERRORS [RPG REV 16.1Ø]

```
OK, SLIST L SAM2
GO
   [RPG REV 16.10] SOURCE:  SAM2                                  13:50:20  12/11/78
(0001)  *SAMPLE PROGRAM #2
(0002)  *
(0003)       F**************************************************
(0004)       F*                                                *
(0005)       F* THIS PROGRAM -                                 *
(0006)       F*                                                *
(0007)       F*   1. MUST BE PRECEDED BY SAMPLE PROGRAM 1     *
(0008)       F*      WHICH LOADS AN INDEXED FILE.              *
(0009)       F*                                                *
(0010)       F*   2. READS AN INDEXED FILE SEQUENTIALLY.       *
(0011)       F*                                                *
(0012)       F*   3. USES A BLOCK LENGTH FOR DISK WHICH        *
(0013)       F*      IS DIFFERENT FROM THAT USED FOR           *
(0014)       F*      LOADING THE FILE IN SAMPLE PROGRAM 1.     *
(0015)       F*                                                *
(0016)       F*   4. COUNTS THE NUMBER OF RECORDS READ SO      *
(0017)       F*      THAT  THE USER CAN QUICKLY VERIFY THAT    *
(0018)       F*      100 RECORDS WERE LOADED.                  *
(0019)       F*                                                *
(0020)       F**************************************************
(0021)       FDISKOUT IPE F 512 128 06AI     1 DISK
(0022)       FPRINT2  O   F  96  96     OF     PRINTER
(0023)  *
(0024)       IDISKOUT NS  01   1 C0
(0025)       I                                      1   6 KEY
(0026)       I                                     82  94 DESC
(0027)       I                                    126 1280RECNBR
(0028)  *
(0029)       C   01      COUNT     ADD  1         COUNT   30
(0030)  *
(0031)       OPRINT2  H  204   1P
(0032)       O       OR        OF
(0033)       O                                     5 'KEY'
(0034)       O                                    22 'DESCRIPTION'
(0035)       O                                    30 'PAGE'
(0036)       O                          PAGE      35
(0037)       O        D  1      01
(0038)       O                          KEY       6
(0039)       O                          DESC      21
(0040)       O                          RECNBRZ   25
(0041)       O        T  3    01 LR
(0042)       O                          COUNT Z   3
(0043)       O                                    26 'RECORDS WERE READ FROM'
(0044)       O                                    44 'THE INDEXED FILE.'
```

```
                ************************
                *   CROSS REFERENCE    *
                ************************

1           [LITERAL------------>001315]    (0029)
COUNT       [FIELD-------------->001322]    (0029)   (0029)   (0042)
DESC        [FIELD-------------->001330]    (0026)   (0039)
DISKOUT     [FILE--------------->000011]    (0021)   (0024)
KEY         [FIELD-------------->001343]    (0025)   (0038)
PRINT2      [FILE--------------->000316]    (0022)   (0031)
RECNBR      [FIELD-------------->001374]    (0027)   (0040)


000 WARNINGS, 000 ERRORS [RPG REV 16.10]

OK,
```

For details of compiling see Section 5.

LOADING

After compilation, the executable memory image is created with the LOAD utility (see Section 6 for details).

```
OK, LOAD
$ DC
$ LO B SAM1
$ LI RPGKID
$ LI KIDALB
$ LI
$ SAVE *SAM1
LOAD COMPLETE
$ QUIT

OK, LOAD
$ LO B SAM2
$ LI RPGKID
$ LI KIDALB
$ LI
$ SAVE *SAM2
LOAD COMPLETE
$ QUIT

OK,
```

CREATING THE MIDAS TEMPLATE

Prior to executing SAM1, a MIDAS template must be created. This procedure is described in Section 8.

```
OK, CREATK
GO
MINIMUM OPTIONS? YES

FILE NAME? DISKOUT
NEW FILE? YES
DIRECT ACCESS? NO              Keyed index file.

DATA SUBFILE QUESTIONS

KEY TYPE: A                    A=ASCII (bytes), B=bits.
KEY SIZE = : B 6               B=bytes, W=words.
DATA SIZE = : 64

SECONDARY INDEX

INDEX NO.? _                   No secondary indices desired.
OK,
```

EXECUTING

After the memory images have been created, the programs can be executed (see Section 7). SAM1 prints a status message into the file PRINT1. SAM2 prints its output into the file PRINT2. These files can be examined with SLIST.

```
OK, R *SAM1

OK, SLIST PRINT1


SAMPLE PROGRAM 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.


OK, R *SAM2

OK, SLIST PRINT2

      .

  KEY      DESCRIPTION    PAGE 0001

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

KEY DESCRIPTION PAGE 0002

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
000280 RECORD NUMBER 56
000285 RECORD NUMBER 57
000290 RECORD NUMBER 58
000295 RECORD NUMBER 59
000300 RECORD NUMBER 60

KEY DESCRIPTION PAGE 0003

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


  KEY       DESCRIPTION     PAGE 0004

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.

OK,
```

APPENDIX C

ASCII CHARACTER SET

The standard character set used by Prime is the ANSI, ASCII 7-bit set.

PRIME USAGE

Prime hardware and software uses standard ASCII for communications with devices. The following points are particularly important to Prime usage.

- Output Parity is normally transmitted as a zero (space) unless the device requires otherwise, in which case software will compute transmitted parity. Some controllers (e.g., MLC) may have hardware to assist in parity generations.
- Input Parity is ignored by hardware and by standard software. Input drivers are responsible for making the parity bit suit the host software requirements. Some controllers (e.g., MLC) may assist in parity error detection.
- The Prime internal standard for the parity bit is one, i.e., '200 is added to the octal value.

KEYBOARD INPUT

Non-printing characters may be entered into text with the logical escape character ^ and the octal value. The character is interpreted by output devices according to their hardware.

Example: Typing ^207 will enter one character into the text.

CTRL-P	('220)	is interpreted as a .BREAK.
.CR.	('215)	is interpreted as a newline (.NL.)
"	('242)	is interpreted as a character erase
?	('277)	is interpreted as line kill
\	('334)	is interpreted as a logical tab (Editor)

Table C-1

ASCII Character Set (Non-Printing)

Octal Value	ASCII Char	Comments/Prime Usage	Control Char
200	NULL	Null character - filler	^@
201	SOH	Start of header (communications)	^A
202	STX	Start of text (communications)	^B
203	ETX	End of text communications	^C
204	EOT	End of transmission (communications)	^D
205	ENQ	End of I.D. (communications)	^E
206	ACK	Acknowledge affirmative (communications)	^F
207	BEL	Audible alarm (bell)	^G
210	BS	Back space one position (carriage control)	^H
211	HT	Physical horizontal tab	^I
212	LF	Line feed; ignored as terminal input	^J
213	VT	Physical vertical tab (carriage control)	^K
214	FF	Form feed (carriage control)	^L
215	CR	Carriage return (carriage control) (1)	^M
216	SO	RRS-red ribbon shift	^N
217	SI	BRS-black ribboon shift	^O
220	DLE	RCP-relative copy (2)	^P
221	DC1	RHT-relative horizontal tab (3)	^Q
222	DC2	HLF-half line feed forward (carriage control)	^R
223	DC3	RVT-relative vertical tab (4)	^S
224	DC4	HLR-half line feed reverse (carriage control)	^T
225	NAK	Negative acknowledgement (communications)	^U
226	SYN	Synchronocity (communications)	^V
227	ETB	End of transmission block (communications)	^W
230	CAN	Cancel	^X
231	EM	End of Medium	^Y
232	SUB	Substitute	^Z
233	ESC	Escape	^[
234	FS	File separator	^\
235	GS	Group separator	^]
236	RS	Record separator	^^
237	US	Unit separator	^_

Notes

1. Interpreted as .NL. at the terminal.

2. .BREAK. at terminal. Relative copy in file; next byte specifies number of bytes to copy from corresponding position of preceeding line.

3. Next byte specifies number of spaces to insert.

4. Next byte specifies number of lines to insert.

Conforms to ANSI X3.4-1968

The parity bit ('2ØØ) has been added for Prime-usage.

Non-printing characters (^c) can be entered at most terminals by typing the (control) key and the c character key simultaneously.

Table C-2

ASCII Character Set (Printing)

Octal Value	ASCII Character	OCTAL Value	ASCII CHaracter	OCTAL Value	ASCII Character
24Ø	.SP.(1)	3ØØ	@	34Ø	` (9)
241	!	3Ø1	A	341	a
242	" (2)	3Ø2	B	342	b
243	# (3)	3Ø3	C	343	c
244	$	3Ø4	D	344	d
245	%	3Ø5	E	345	e
246	&	3Ø6	F	346	f
247	' (4)	3Ø7	G	347	g
25Ø	(	31Ø	H	35Ø	h
251	)	311	I	351	i
252	*	312	J	352	j
253	+	313	K	353	k
254	, (5)	314	L	354	l
255	-	315	M	355	m
256	.	316	N	356	n
257	/	317	O	357	o
26Ø	Ø	32Ø	P	36Ø	p
261	1	321	Q	361	q
262	2	322	R	362	r
263	3	323	S	363	s
264	4	324	T	364	t
265	5	325	U	365	u
266	6	326	V	366	v
267	7	327	W	367	w
27Ø	8	33Ø	X	37Ø	x
271	9	331	Y	371	y
272	:	332	Z	372	z
273	;	333	[	373	{
274	<	334	\	374	|
275	=	335	]	375	}
276	>	336	^ (7)	376	~ (1Ø)
277	? (6)	337	_ (8)	377	DEL (11)

Notes

1. Space forward one position
2. Terminal usage - erase previous character
3. £ in British use
4. Apostrophe/single quote
5. Comma
6. Terminal usage - kill line
7. 1963 standard ↑; terminal use - logical escape
8. 1963 standard ←
9. Grave
10. 1963 standard ESC
11. Rubout - ignored

Conforms to ANSI X3.4-1968
1963 variances are noted

The parity bit ('2ØØ) has been added for Prime usage.

APPENDIX D

SUMMARY OF DIFFERENCES FROM IBM SYSTEM 3, MODEL 1Ø

Prime's RPG II is a dynamically evolving product with enhancements continually being considered for each master disk revision. The following features are not supported as of Rev. 16; check with your local Prime Sales Office for the current status of RPG II features.

1. Devices other than DISK or CONSOLE. Prime computers are disk-oriented rather than card- or tape-oriented. The large amount of disk storage will make it unnecessary for most users to need some of the devices they currently utilize. Devices current not supported are:

    - Combined files.
    - Stacker select.
    - Additional I/O areas.
    - Multiple volumes.

2. EBCDIC related operations:

    - Zones (MHLZO, MLHZO, MHHZO, MLLZO, TESTZ).
    - Bit operations (TESTB, BITON, BITOF).
    - Zone/Digit in record identification codes.

3. File Translation. Prime's native environment is ASCII. See Section 4 for ASCII/EBCDIC translation using the MAGNET utility.

4. Forms positioning (1P indicator in column 41 of Header Specifications) is not supported. This feature may be simulated by inserting halt indicators in the source program.

5. Record Address (ADDROUT) files containing binary relative numbers.

6. Special Words: *PRINT and *PLACE.

7. Overprint Option

    - Space zero before (column 17 of Output).
    - Space zero after (column 18 of Output).

8. Packed Keys.

9. Other than domestic edit code format.

1Ø. Telecommunications. Prime has its own networking facility - PRIMENET.

APPENDIX E

COMPILER ERROR AND WARNING MESSAGES

During compilation, if an error or warning condition is found (or a condition not supported by Prime RPG II), a message is inserted into the listing surrounded by blank lines. These may optionally be listed on the users terminal as well with the -ERRTTY option.

The format of these messages is:

```
****ERROR, LINE (xxxx) COLUMN yy-zz [contents]----message.
**WARNING, LINE (xxxx) COLUMN yy-zz [contents]----message.
```

xxxx	Line number of the statement in error.
yy-zz	Beginning and ending column numbers of the field in error. If yy is the same as zz, the zz portion is omitted.
contents	The contents of the columns yy-zz.
message	Some comments about the field in error. These messages are listed below in alphabetical order within the specification sheet from which they may arise.

Examples:

```
****ERROR, LINE (0012) COLUMN 44-51 [ 34-23 ]----FIELD TOO LARGE.

**WARNING, LINE (0025) COLUMN 42-58 [ 2     ]----SHOULD BE BLANK.

**WARNING, LINE (0035) COLUMN 01-05 [ 03015  ]----NOT IN SEQUENCE.
```

An error message requires that the fault be corrected and the program recompiled prior to loading. Warning messages indicate a potential problem. The programmer must judge whether the program will operate as intended if the memory image is build using this binary file. As a general rule, all faults, either error or warning, should be corrected before building the executable memory image.

COMPILER MESSAGES

General Error Messages

These messages can be generated during the compilation of any sheet.

INPUT SPECS ON PREV LINE INCOMPLETE.

Warning: No input field descriptors have been provided for the file in the previous line. Action: Insert input field descriptors in the source program.

OUTPUT SPEC ON PREVIOUS LINE INCOMPLETE.

Error: No output field descriptors have been given for the output file named in the previous line. Action: Add output field descriptors.

STMT OUT OF SEQUENCE -- IGNORED.

Warning: A source statement is out of order. Action: Determine if the source line should beincluded in the source file. If so, put it in the proper place (H, F, E, L, I, O, C) or else delete it from the source text.

UNIDENTIFIED STATEMENT.

Error: A line specification type is not H, F, E, L, I, C, or O. Action: If the line should be included in the file, then correct column 6, else delete the line.

Calculation Specification Sheet Messages

These messages are generated during the compilation of the Calculation Specification sheet.

DECIMAL COUNT NOT ALLOWED.

Warning: The field is alpha but column 52 is non-blank. Action: Remove the entry in column 52.

FIELD LENGTH NOT ALLOWED.

Error: The resulting field was previously defined. Action: Remove entry in columns 49-51.

FIELD LENGTH TOO LARGE.

Warning: A numeric field size cannot be larger than 15 positions. Action: Decrease resulting field size.

FIELD SHOULD BE BLANK.

Warning: A field name should not have been specified in the columns mentioned for the operation specified. Action: Remove the field name.

HALF-ADJUST FIELD.

Error: The entry in column 53 is neither blank nor H. Action: Correct column 53 and recompile.

IMPROPER CONTROL LEVEL.

Error: The contents of columns 7-8 of the calculation specifications are not AN, OR, LR, SR, or LØ-L9. Action: Specify the proper entry.

IMPROPER EXTERNAL NAME.

Error: The name given for the external subroutine name does not conform to the standard of 5 or 6 characters, the first four of which are SUBR and the remaining alphabetic. Action: Change the name, and recompile.

IMPROPER FIELD ENTRY.

Error: The field name referenced in the calculation specifications is invalid. The field name should consist only of letters or digits and/or a comma (,) if it has been defined as an array in the extension specifications. Action: Make the proper field name or constant entry.

INCONSISTENT FIELD SIZE.

Error: A special variable, (UDAY, PAGE, etc.) was specified in the result field and resulting field length was given. The size of the special fields is predetermined. Action: Remove entries in 49-52 and recompile.

INDICATOR REQUIRED.

Warning: Depending upon the operation to be performed, a resulting indicator must be present in 53-54, 55-56, or 57-58. Action: Insert the proper entry in 53-58.

KEY LENGTH CONFLICT.

Error: The lower limit set via SETLL is incompatible with the key field size as defined in the file specification line. Action: Correct either the file specification, or the field size specified in the calculation line, whichever is in error.

LIMITS TO BE SET BY RAF FILE.

Error: A SETLL operation was used for a file specified for sequential processing within limits by using a record address file containing limit records. SETLL does not specify an upper limit, whereas RAF does. Action: Either remove the file and extension specifications for the RAF file and use SETLL or remove the SETLL operation.

MULTIPLE DEFINITION.

Error: The label name given with this tag statement has been used before. Action: Correct spelling or use another statement label name.

MUST BE [SR].

Error: The calculation specification is not SR in column 7-8, whereas the previous line was. SR specifications are always after all other calculations. Action: Either specify SR in 7-8 or move the calculation line to the proper place in the calculation section.

MUST BE [SR].

Error: Either a BEGSR or an ENDSR was found with no SR in 7-8. Action: Insert SR in columns 7-8, recompile.

NO INDICATORS ALLOWED.

Warning: Based on the calculation operation requested in 28-32, no resulting indicators are allowed in 53-58. Action: Remove entry in 53-58 and recompile.

NO MATCHING BEGSR.

Error: An ENDSR was found before a BEGSR. Action: Insert a BEGSR label line and recompile.

NOT ALLOWED WITHIN A SUBR.

Error: A BEGSR is found within a subroutine. An RPG subroutine must begin with a BEGSR and end with a ENDSR with no other subroutines nested between. Action: Adjust subroutine nesting and recompile.

NOT AN INDEX DEMAND FILE.

Error: A SETLL operation has been requested but the file description does not specify the file designation as a demand index file. (D in column 18 and I in column 32). Action: If SETLL is intended, define the file accordingly, otherwise remove SETLL line.

OPERATOR NOT IMPLEMENTED.

Error: The requested operation code in 28-32 of the calculations, though valid, has not been implemented in Prime's RPG II compiler. Action: Delete the line and revise the program logic accordingly.

RESULT DECIMAL POSITION.

Error: If the result field is to be numeric, Ø-9 has not been entered in column 52. If the result field is to be alpha, column 52 should be blank. If numeric, the number of decimal positions cannot be longer that the field length. Action: Correct column 52, depending upon whether the field is alpha or numeric.

RESULT FIELD LENGTH.

Error: The resulting field length exceeds 15 for numeric and 256 for alpha. Action: Alter columns 49-51 appropriately.

UNRECOGNIZED OPERATOR.

Error: Column 28-32 of the calculation line is not a valid operator. Action: Enter the proper RPG II operation code in 28-32 and recompile.

WRONG FILE TYPE.

Error: The operation specified (CHAIN, READ, or FORCE) specified on the file is invalid for the file's type. Action: Change the operation code to one appropriate to the file type, or redefine the file.

Extension Specification Sheet Messages

These messages are generated during the compilation of the Extension Specification Sheet.

COMPILE TIME ARRAY/TABLE.

Error: Compile time tables or arrays do not allow packed data. Action: To avoid this error on the next compilation, leave columns 43 and 55 blank.

DECIMAL POSITIONS FIELD.

Error: The entry found in column 44 or 56 is neither blank nor Ø-9. Action: Make the proper decimal position entry (Ø-9, blank) in columns 44 and 56.

ENTRY LENGTH.

Error: The ARRAY/TABLE data should be numeric. The length of each entry is either not Ø-15 or the data is packed binary and the entry is neither 4 nor 9. For numeric tables or arrays in packed decimal format, enter the unpacked decimal length in columns 4Ø-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 4Ø-42 is 4. For a 4-character binary field the entry is 9. Action: Enter proper decimal entry for the ARRAY/TABLE type.

EXEC TIME P/B ARRAY/TABLE.

Error: The extension line indicates an execution time array or table with packed data. Packed data can only be used with pre-execution time tables or arrays. Action: To avoid this error on the next compilation leave columns 43 and 55 blank.

FILE HAS TOO MANY EXTENSIONS.

Error: The file mentioned as the FROM file in the extension specifications has previously had an extension specification or no extension was anticipated based upon the file descriptor. Action: Make sure the entry refers to the required file.

FILE NAME MISSING.

Error: 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 the TO Filename (columns 19-26). Action: Insert file name in 19-26.

IMPROPER ENTRIES/RECORD.

Error: The value given in column 33-35 of the extension specifications must be 1-999. Action: Supply proper entry in columns 33-35.

IMPROPER ENTRIES/TABLE.

Error: The entry given in columns 36-39 of the extension specifications must be 1-9999. Action: Insert proper entry.

IMPROPER ENTRY LENGTH.

Error: The value contained for the length of entry is not 1-256. Action: Adjust length of entry value.

IMPROPER FILE NAME.

Error: The file name used does not meet the proper file name characteristics. Action: Change file name and recompile.

IMPROPER TABLE/ARRAY NAME.

Error: The table or array name given does not start with an alpha character, $, @, or #. Action: Change name to conform to requirements.

INCONSISTENT ENTRIES.

Error: Packed data has been specified but number of decimal positions has been left blank. Action: Enter Ø-9 in columns 44 and 56.

INCORRECT FILE DESIGNATION.

Error: An extension specification has been given for a RAF file, but the file type (col. 16 of file specification) is neither P, S, or D. Action: Change file designation and recompile.

ONLY ONE RAF FILE ALLOWED.

Error: Only one record address file can be used in a program. Action: Change file type and remove extension specification.

P/B NEEDS UNCOMPRESSED FILE.

Error: The FROM or TO file has designated the data type of the array or table as packed but the corresponding file format (col. 19 of file specifications) is not uncompressed. Action: Correct either column 43 of the extension specifications or column 19 of the files specifications for the given file.

PRE-EXEC TIME P/B ARRAY/TABLE.

Error: The extension line indicates a pre-execution packed decimal array or table but the number of entries per record column has been left blank. Action: Make the proper entry in columns 33-35.

SEQUENCE FIELD.

Error: The entry found in column 45 and/or 57 is not blank, A, or D. Action: Enter required sequence checking option.

TOO MANY COMPILE-TIME TABLES OR ARRAYS.

Error: More than 3Ø compile time tables have been used in the program. Action: Reduce the number of compile time tables used in the program.

TOO MANY ENTRIES PER RECORD.

Error: The number of entries per record is greater than the number of entries per table. Action: Adjust either the number of entries per record or the entries per table.

UNRECOGNIZED ENTRY.

Error: The data type given for column 43 or 55 is not P (Packed), B (Binary), or blank (Alphanumeric or unpacked decimal). Action: Insert correct entry in the respective column.

UNRECOGNIZED FILE NAME.

Error: The filename in the noted columns of the extension specifications has not been defined in the file specifications. Action: Define the file in the file specifications or correct the name in the extension specifications.

File Specification Sheet Messages

These messages are generated during the compilation of the File Specification Sheet.

ADDITION DEVICE MUST BE DISK.

Error: File addition (A in column 66) has been specified with a device other than disk. Action: Change device to disk.

CONTINUATION-LINE FIELD.

Error: The entry in column 53 is neither blank nor K. Action: Make proper entry in column 53.

DEVICE CANNOT BE CONSOLE.

Error: The file format (column 19) for console files cannot be uncompressed. Action: Revise either the device or file format.

DEVICE FIELD.

Error: The entry in columns 4Ø-46 is not a valid device name. Action: Enter the proper device name in column 4Ø-46.

DEVICE NAME NOT 'SPECIAL'.

Error: A subroutine name has been given in column 54-59 but the device given is other than SPECIAL. Action: Either remove the entry in 54-59 else change device name to SPECIAL.

DUPLICATE FILE NAMES.

Error: The file name has already been used. File names must be unique within a program. Action: Assign a unique name to the file.

END-OF-FILE FIELD.

Error: The entry in column 17 of the file specifications is neither E nor blank. Action: Make the proper end of file entry in column 17.

EXTENSION CODE REQUIRED.

Error: The file designation is either table or record address and thus an extension specification is necessary. Action: Enter an E in column 39.

EXTENSION-CODE FIELD.

Error: The entry in column 39 is not blank, E, or L. Action: Make the proper entry in column 39.

FIELD LENGTH.

Error: The entry in columns 29-30 is not 0-32. The key field must be 32 or less for unpacked keys. Packed keys are not allowed. Action: Make the length of key field entry in columns 29-30 a valid key length.

FILE ADDITION.

Error: The entry in column 66 is neither blank nor A. Action: Make the proper entry in column 66.

FILE CONDITION FIELD.

Error: The entry in 71-72 is neither blank nor U1-U8. Action: Insert proper entry in 71-72.

FILE DESIGNATION FIELD.

Error: The entry in column 16 is invalid. Action: Make the proper entry in column 16.

FILE FORMAT FIELD.

Error: The entry in column 19 must be U for uncompressed files, blank for variable length, or blank or F for fixed length records. Action: Make the proper entry in column 19.

FILE ORGANIZATION TYPE.

Error: The entry given in column 32 is not I, D, or blank. Action: Make appropriate entry in column 32.

FILE TYPE FIELD.

Error: The file type entry in column 15 is not I, O, U, or D. Action: Enter the proper file type in column 15.

ILLEGAL FILE (32) OR MODE (28).

Error: File addition (A in column 66) has been requested with sequential-within-limits and/or demand processing. File addition can be specified for sequential and indexed output files on disk only. Action: Make compatibility adjustments to columns 28,32, and 66.

IMPROPER FILE NAME.

Error: The file name given in columns 7-14 of the file specification either does not begin with A-Z, $, or #; or one of the characters in 8-14 is not A-Z, Ø-9, $, @, or #. Action:Change file name to meet standards.

KEY START LOCATION.

Error: Columns 35-38 do not contain a number from 1-9999 (for an index file) or is not blank. Action: Make the proper key start entry in columns 35-38.

NO LINE-CTR EXTN FOR INPUT FILE.

Error: The file type is input and a line counter code (L in 39) has been specified. Line counters are valid only for output files. Action: Change column 15 to Ø or column 39 to E or blank.

NO SUBR NAME PRESENT.

Error: 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 I/O operations for a file assigned to a special device. The subroutine name must start with *, since it will be a memory image. Action: Enter a subroutine name.

NOT AN ARRAY NAME.

Error: The name given in columns 54-59 is not a valid array name. Action: Change name in accordance with naming conventions.

NOT IMPLEMENTED.

Warning: Either BUFOFF or INDEX has been specified in 54-59 but these functions have not been implemented in Prime's RPG. Action: Alter program.

OVERFLOW INDICATOR FIELD.

Error: The entry in columns 33-34 was not OA-OG, or OV. Action: Enter OA-OG or OV in columns 33-34 to specify overflow for this file, otherwise leave columns 33-34 blank.

PROCESSING MODE.

Error: The entry in column 28 is not L, R, or blank. Action: Enter proper mode of processing in column 28.

RECORD ADDRESS TYPE.

Error: The entry in column 31 is neither A nor blank.. Packed keys are not supported. Action: Make the proper record address type entry in column 31.

RECORD LENGTH.

Error: The record length specified in columns 24-27 of the file specifications is not a number from Ø-9999. Action: Correct entry, by using Ø-9999 in columns 24-27.

SEQUENCE FIELD.

Error: The entry in column 18 is not A, D, or blank. Action: Make the proper entry in column 18.

SHOULD BE BLANK.

Warning: The columns mentioned are non-blank and they should not contain an entry. Action: Remove the entry.

SPECIAL NAME FIELD.

Error: The subroutine name in columns 54-59 is not a valid file name. Action: Make proper file name entry in 54-59.

Header Specification Sheet Messages

These messages are generated during the compilation of the Header (Control Card) Specification Sheet.

DATE/NUMBER FORMAT.

Error: The entry in column 21 is not blank, D, I, or J. Action: Make the proper entry in column 21.

ONLY ASCII SUPPORTED.

Error: Column 43 is neither blank nor F. Action: Make proper entry in column 43.

SHOULD BE BLANK.

Error: An entry is contained in the column(s) noted, and the column(s) should be blank. Action: Remove the entry.

Input Specification Sheet Messages

These messages are generated during the compilation of the Input Specification Sheet.

CHARACTER DESCRIPTOR.

Error: The entry in column 26, 33, or 4Ø is not a C (Z and D are not supported). Action: Make the proper entry in column 26, 33, or 4Ø.

CONTROL LEVEL INDICATOR.

Error: The control level indicator in column 59-6Ø is not L1-L9 or blank. Action: Make the proper entry in columns 59-6Ø.

FIELD DECIMAL POSITIONS.

Error: The entry in column 53 is not blank, Ø-9, or N. Action: Make the proper entry in column 53.

FIELD END LOCATION.

Error: The entry in columns 48-51 is not 1-9999. Action: Enter a number from 1-9999 in columns 48-51.

FIELD ID POSITION.

Error: The entry in columns 21-24, 28-31, or 36-38 is not 1-9999 or blank. Action: Make the proper entry in the column in error.

FIELD NAME.

Error: The entry contained in columns 53-58 is not a valid field name. Action: Correct spelling and/or revise name to conform to standards.

FIELD START LOCATION.

Error: The entry in columns 44-47 is not 1-9999. Action: Make columns 44-47 a number from 1-9999, right-justified.

FIELD TOO LARGE.

Error: The field size is execessive for its type. Maximum alpha field length is 256 characters and maximum numeric field length is 15. Action: Revise field start and/or end position to conform with field length requirements.

FIELD TOO SMALL.

Error: The field start location is greater than the field end location. Action: Make the proper entries in columns 44-51.

FILE DEFINED AS OUTPUT.

Error: The file for which input specifications have been provided is not I, U, or D. Action: Make the proper entry for the file in column 15 of the file specification, or remove input descriptor source lines that pertain to this file.

FILE NAME.

Error: The name given as the file name in columns 7-14 is an invalid file name. Action: Make correction to file name.

IMPROPER PREVIOUS LINE.

Error: For AND lines: the previous line has not fully utilized the record identification codes; less than 3 were used. Action: Use all record identification codes on a source line before adding an AND line. For OR lines: the previous line contained no record identification codes. Action: Make the proper record identification entries in the line preceeding this OR line.

INCONSISTENT FIELD.

Error: A previous definition for the field exists and the field length and/or number of decimal positions is incompatible with the previous definition. Only special name fields can be defined with various lengths. Action: If a special name is intended use the special name, otherwise correct the field size.

INCONSISTENT USAGE.

Error: The previous definition of the field, array or table is incompatible with the current input line. Action: Compare the previous definition with the current definition and adjust accordingly.

MATCHING FIELDS.

Error: The entry in columns 61-62 is not blank or M1-M9. Action: Make the proper entry in column 61-62.

NO FILE SPECIFIER.

Error: No file specification has been given (F in column 6) for the name given in columns 7-14. Action: Correct file name or add a file specification to the source program.

NO INPUT FILE SPECIFIED.

Error: An input field descriptor has been detected but an input file descriptor has not been associated with it. Action: Enter a source line containing the appropriate input file descriptor.

'NOT' FIELD.

Error: The entry contained in column 25, 32, or 39 is neither N nor blank. Action: Make the proper entry in the column in error.

NUMBER OF RECORD TYPE.

Error: Column 17 entry is neither 1 nor N. Action: If sequence checking is intended, provide proper entry in 17, otherwise make it blank.

OPTIONAL RECORD TYPE.

Error: Sequence checking option has been selected but the entry in column 18 is neither blank nor O. Action: Make appropriate entry in column 18.

P/B NEEDS UNCOMPRESSED FILE.

Error: Data in the file is either packed or binary but column 19 of the respective file description specifications is not U. Action: Enter U in column 19 of the file descriptor specification or remove the entry in column 43 of the input specifications if the data is unpacked.

PREV LINE NOT FIELD SPECIFIER.

Warning: An input file descriptor has been given immediately following another input file descriptor. Action: Check validity of source statements and add field descriptor lines if necessary.

PREV RCRD HAS NO FIELD DESCRIPTOR.

Warning: The previous source statement was an input file descriptor. Action: Make sure that all fields to be used from input records are described.

SEQ NUMBER NOT LTRS OR DIGITS.

Error: The sequence entry in columns 15-16 is neither a two-digit number nor a two-character alphabetic entry. Action: Make the proper sequence entry.

SHOULD BE BLANK.

Error: The column(s) noted should not contain an entry. Action: Remove the entry in the columns.

SIZE FOR BINARY FIELD.

Error: Binary field length specified is neither 2 nor 4 bytes. Action: Adjust field start and end entries accordingly.

SIZE FOR PACKED FIELD.

Error: A packed field must be greater than zero and less than 16 unpacked characters in length. Action: Revise field start and field end position.

UNRECOGNIZED ENTRY.

Error: The entry in column 43 is not blank, P, or B. Action: Make the proper entry in column 43.

WRONG TYPE OF FILE DESIGNATION.

Error: Look-ahead fields cannot be specified for chained or demand files. Action: Make sure that look ahead records are not specified for demand or chained files.

Line Counter and Extension Specification Sheet Messages

These messages are generated during the compilation of the Line Counter Specification Sheet or the the Extension Specification Sheet.

FORM LENGTH.

Error: The entry found in column 18-19 is not FL. The only valid entry is FL. Action: Enter FL in columns 18-19.

IMPROPER FILE NAME.

Error: The file name given in columns 7-14 is invalid according to file naming conventions. Action: Revise file name.

IMPROPER FORM LENGTH.

Error: The form length specified in columns 15-17 of the line counter specification is greater than 112 lines. Action: Make the proper entry in columns 15-17.

IMPROPER OVERFLOW LINE NO.

Error: The entry found in columns 20-22 is greater than 112. Action: Specify the overflow line number to be less than 112.

LINE CNTR NOT EXPECTED.

Warning: Column 39 of the related file specification line contains an E and thus the line counter specification is inappropriate. Action: Correct either the file specification entry or remove the line counter.

MULTIPLE EXTENSIONS FOR THIS FILE.

Error: An extension or line counter specification line is already assigned to this file. Action: Make proper entry.

NO FILE SPECIFICATION LINE.

Error: A file specification line does not exist for the file. Action: Correct file name or add a specification line.

OVERFLOW LINE.

Error: Columns 23-24 are not OL. Action: Make the entry in 23-24, OL.

Output Specification Sheet Messages

These messages are generated during the compilation of the Output Specification Sheet.

BLANK AFTER FIELD.

Error: Column 38 contians an entry that is other than blank or B. Action: Make the proper entry in column 38.

FETCH OVERFLOW.

Error: Column 16 does not contain an F or a blank. Action: Make the proper entry in column 16.

FIELD END POSITION.

Error: Column 4Ø-43 does not contain a number from 1-9999. Action: Make the proper entry in column 4Ø-43.

FIELD NAME.

Error: The field name in columns 32-37 is invalid according to naming conventions. Action: Make the proper field name entry name entry starting in column 32.

FILE DEFINED AS INPUT.

Error: Output cannot be performed on a file specified as an input file in the file specifications. Action: Determine intentions for this file and change either the file specifier or the output specifications.

FILE NAME.

Error: The file name in columns 7-14 does not meet file naming conventions. Action: Make proper file name entry starting in column 7.

IMPROPER PREVIOUS LINE.

Error: An AND entry is given in columns 14-16 but the previous line was neither an AND line nor an output file descriptor. Action: Make sure that record identification entries in column 15-31 or another AND entry precedes any AND line.

MISSING ENDSR OPERATOR.

Error: A calculation subroutine was not completed properly. An ENDSR calculation specification must be the terminating line of a subroutine. Action: Add a calculation specification line with an ENDSR operation to terminate the open subroutine.

MISSING FIELD NAME.

Error: Neither a field name (columns 32-37) nor a constant (columns 45-70) have been given on the output line. Action: Make the proper entry for the field name or the constant.

MISSING INDICATOR.

Error: Column 24-26, 27-28, or 30-31 is blank while the respective negate column contains an N. Action: Remove the N or enter the necessary indicator in the columns in error.

MUST BE BLANK.

Error: The columns mentioned should be blank, for the type of output specification. Action: Blank the columns in error.

MUST START WITH A QUOTE.

Warning: The constant or edit word given in column 45-70 does not start with a quote. Action: Adjust the entry in 45-70 so that it starts and ends with a quote.

NEGATE FIELD.

Error: The entry in column 23, 26, or 29 is not blank or N. Action: Make the proper entry in the column in error.

NO ADD SPECIFIED ON FILE DESCRIPTOR.

Error: ADD has been specified in columns 16-18 but an A was not found in column 66 of the field descriptor. Both entries are necessary in order to add to a file. Action: Either add the A in column 66 of the field descriptor or remove ADD from the output specification.

NO FIELD-END POSITION.

Error: The entry in column 40-43 is either blank else 0. Action: Enter a positive, non-zero number in column 40-43.

NO FILE SPECIFIER.

Error: No file specification line has been given for the file named in columns 7-14. Action: Either correct entry in 7-14 else add a file specification line.

NO OUTPUT FILE SPECIFIED.

Error: No file specification source line has been provided for the output file specified in columns 7-14. Action: Correct the entry in 7-14 or add a file specification line.

NO PREVIOUS REFERENCE.

Error: The field name given in columns 32-37 was not defined previously in input or calculation specifications. Action: Make the proper field entry starting in column 32.

NOT A NUMERIC FIELD.

Warning: The type of the field to be edited is not unpacked numeric. Action: Remove the entry in column 38.

OUTPUT RECORD TYPE.

Error: The entry in column 15 is not H, D, T, or E. Action: Make the proper entry in column 15.

P/B NEEDS UNCOMPRESSED FILE.

Error: Packed decimal or packed binary data are to be written to the file specified but the file specification did not contain a U in column 19. Action: Correct file specification or remove the P or B from column 44.

PACKED/BINARY SPEC.

Error: Edit code cannot be specified with packed decimal or packed binary data fields. Action: Specify another field name or remove the entry in column 38.

PREV LINE NOT FILE SPECIFIER.

Error: The output field line has not been attached to an output file line. Action: Provide an output file specification line in the source file.

PREV RCRD HAS NO FIELD DESCRIPTORS.

Error: The previous output specification statement has no field specification lines associated with it. If indicator conditions are satisfied, the output that will be generated will be a blank line. Action: Verify that this is the intended situation; if not, add output field specification lines.

SKIP-AFTER TOO LARGE.

Error: The maximum number of lines that can be skipped after an output line is 99. Action: Alter the entry in colulmns 21-22 to be a number from 1-99.

SKIP-BEFORE TOO LARGE.

Error: The maximum number of lines that can be skipped prior to an output line is 99. Action: Revise entry in columns 19-2Ø to be a number from Ø-99.

SPACE-AFTER FIELD.

Error: The entry in column 18 is not blank, or Ø-3. Action: Make the proper entry in column 18.

SPACE-BEFORE FIELD.

Error: The entry found in column 17 is not blank, or Ø-3. Action: Make the proper entry in column 17.

UNRECOGNIZED EDIT CODE.

Error: The edit code specified in column 38 is not blank, 1-4, X-Z, A-D, or J-M. Action: Make the proper edit code entry in column 38.

UNRECOGNIZED ENTRY.

Error: Column 44 is not blank, P, or B. Action: Make the proper entry in column 44.

UNRECOGNIZED INDICATOR.

Error: The entry in columns 24-25, 27-28, or 3Ø-31 is not blank, LR, MR, OA-OG, OV, UØ-U9, LØ-L9, HØ-H9, or ØØ-99. Action: Make the proper entry in the columns in error.

APPENDIX F

RUN-TIME USER MESSAGES

Messages may be output by the system to the user device during execution of an RPG II program. These messages, along with their cause and suggested procedure for continuing, are listed later in this appendix.

Each message is followed by a PAUSE prompt on a new line:

```
****PA

OK,
```

At this point, the user types S(CR) to proceed. The consequences of proceeding are different for each message and are described under that message.

RUN-TIME MESSAGES

****ABOVE RECORD FROM filename NOT IN ORDER. TYPE S(CR) TO IGNORE RECORD.

Reason: The program is sequence checking records from this file and the most recent record read is not in the expected sequence.

Suggested action: If this record should be in the file, reorganize.

Effects of typing S(CR): Next record is read from this file.

****ABOVE RECORD FROM filename OUT OF SEQUENCE. TYPE S(CR) TO IGNORE RECORD.

Reason: While using matching fields to process records, a record was found to be out of sequence.

Suggested action: Examine file.

Effects of typing S(CR): Next record is read from this file.

****ABOVE RECORD FROM filename UNRECOGNIZED. TYPE S(CR) TO IGNORE RECORD.

Reason: The record read does not relate to a record ID indicator.

Suggested action: Examine record and if it should not be a part of the file, remove it, otherwise identify the record and assign an indicator for its type in the program.

Effects of typing S(CR): Next record is read from this file.

****DATA IN FILE filename NOT IN SEQUENCE. TYPE S(CR) TO CONTINUE.

Reason: The pre-execution table from the file mentioned does not contain data in the sequence specified.

Suggested action: Alter the extension specification reference, otherwise fix file data.

Effects of typing S(CR): Program execution continues with out-of-sequence table.

****ERROR. ATTEMPT TO CONVERT BINARY DATA TO NUMERIC FIELD FROM POSITION position OF FILE filename IS UNSUCCESSFUL. TYPE S(CR) TO CONTINUE.

Reason: The data contained in the field, defined as binary on the input specifications, is either too small or too large.

Suggested action: Check definition of input field. If field is not defined incorrectly, data have become corrupted.

Effects of typing S(CR): Program exection continues with possibly unreliable data.

****ERROR. ATTEMPT TO CONVERT NON-NUMERIC DATA TO BINARY FROM FIELD NEAR LOCATION location. TYPE S(CR) TO CONTINUE.

Reason: The input data field contains non-numeric data. This situation was detected when binary output was attempted.

Suggested action: Check reliability of field to be stored as packed binary.

Effects of typing S(CR): Program continues but result field may contain erroneous results.

****ERROR. ATTEMPT TO CONVERT NON-NUMERIC DATA TO PACKED DECIMAL FROM FIELD NEAR LOCATION location. TYPE S(CR) TO CONTINUE.

Reason: The data field to be written to the output file contains non-numeric data. Only numeric data can be packed.

Suggested action: Remove P in column 44 of output specification and adjust end position in output specification, otherwise correct incoming data field contents.

Effects of typing S(CR): Results contained in the field in error may be erroneous.

****ERROR. ATTEMPT TO CONVERT PACKED DECIMAL DATA TO NUMERIC FIELD FROM POSITION position OF FILE filename IS UNSUCCESSFUL. TYPE S(CR) TO CONTINUE.

Reason: As defined on the input, a field is packed decimal but contains non-numeric data.

Suggested action: Check definition of the input field; if it is correct, then the data have been corrupted.

Effects of typing S(CR): Operations performed on the field to be converted may contain erroneous results.

***FILE filename CONTAINS UNREAD DATA. TYPE S(CR) TO CONTINUE.

Reason: Data in the file remain beyond the requirements for the table or array.

Suggested action: Check table/array defintion in extension specifications. If extension specifications are correct, remove extraneous data from the file.

Effects of typing S(CR): Program execution continues with the table filled with data as required by the extension specifications.

****HALT INDICATOR xxxx IS SET. TYPE S(CR) TO CONTINUE.

Reason: This is a halt that was built into the program.

Suggested action: Consult operator documentation to determine the intention of this halt.

Effects of typing S(CR): Indicator is set off and program execution continues at the next line.

****HI/LO LOOKUP BUT TABLE UNORDERED. TYPE S(CR) TO SKIP LOOKUP.

Reason: A resulting indicator of a LOKUP operation is either HI or LO, but the table or array definition from the extension specifications has not specified that the data is in either ascending descending order.

Suggested action: Revise extension specification definition or resulting indicator type.

Effects of typing S(CR): Search is done but results may be erroneous.

****IMPROPER SUBROUTINE NESTING. TYPE S(CR) TO CONTINUE.

Reason: All subroutines must reside after all other calculations. A subroutine may be called from the main portion of calculations or by another subroutine. A subroutine cannot call itself or call the subroutine that called it.

Suggested action: Check the use and placement of the subroutine in error.

Effects of typing S(CR): Execution continues, ignoring nesting.

KIDA [KIDA error message and number]

Reason: KIDA/MIDAS has detected a fault condition.

Suggested action: Consult Reference Guide, MIDAS for detailed information.

Effects of typing S(CR): Not applicable.

****LINE xxxx ATTEMPTS A MINUS SQUARE ROOT. PRESS S(CR) TO CONTINUE.

Reason: Square root operation is invalid for a negative number.

Suggested action: Check possible range of field and condition square root operation accordingly.

Effects of typing S(CR): Program continues, first converting the number inside the radical to positive, then square root is performed.

****LINE xxxx CONTAINS A DIVIDE-BY-ZERO. PRESS S(CR) TO CONTINUE.

Reason: The contents of the divisor field is zero and division by zero is mathematically undefined.

Suggested action: Condition division operations based on non-zero divisor. That is, check contents of divisor field prior to division.

Effects of typing S(CR): Result is set to zero.

****LINE xxxx CONTAINS ARGUMENT RANGE ERROR. PRESS S(CR) TO CONTINUE.

Reason: The number of decimal positions of factor-1, factor-2, or the result field is not Ø-9 or the field width is not Ø-24.

Suggested action: Check field size allocations for all fields.

Effects of typing S(CR): Program will continue without executing this computation.

****LINE xxxx CONTAINS IMPROPER BRANCH. TYPE S(CR) TO IGNORE BRANCH.

Reason: An improper GOTO operation has been attempted.

Suggested action: If a subroutine is involved, consult rules for use of the GOTO operation within a subroutine.

Effects of typing S(CR): Program execution continues at the next line.

****LINE xxxx CONTAINS IMPROPER SUBSCRIPT. PRESS S(CR) TO USE ENTRY 1.

Reason: The subscript given explicitly or as a variable is less than 1 or greater than the maximum number of elements for the table (columns 36-39 of extension specification).

Suggested action: If explicit reference, change the subscript to the proper number within the range. If a variable has been used, check its range prior to this operation.

Effects of typing S(CR): Subscript number is set to one and processing continues for the line.

****MORE THAN 13 FILES OPEN. JOB IS ABORTED.

Reason: The file specifications request more than 13 files to be used in processing.

Suggested action: The only way to avoid this message is to reduce the number of files.

Effects of typing S(CR): Not applicable.

****NOT ENOUGH DATA IN FILE filename. TYPE S(CR) TO CONTINUE.

Reason: The definition of the table/array on the extension specifications requires more data than have been provided in the file.

Suggested action: Increase the data in file or change the extension specification definition of the array/table.

Effects of typing S(CR): Program execution continues with the table/array containing unknown data from the point where file data ran out.

operation OPERATION REQUESTED, LINE xxxx

Reason: The operation requested has not been implemented.

Suggested action: Alternate programming method must be used to arrive at the desired result.

Effects of typing S(CR): Not applicable.

filename keyfield RECORD IN USE. TYPE S(CR) TO TRY AGAIN.

Reason: The record has been locked in anticipation of an update. If no other user is accessing the file, the file was not properly closed on a previous usage.

Suggested action: If another user is accessing the file, take the S(CR) option, otherwise close all files and perform necessary steps, depending upon your application, to have a reliable file.

Effects of typing S(CR): The desired KIDA operation is executed again.

****UNSUCCESSFUL CHAIN TO ABOVE RECORD. TYPE S(CR) TO SKIP PAST OUTPUT.

Reason: A chain operation has been attempted on the file and KIDA/MIDAS was unable to retrieve the record.

Suggested action: This is a serious message and indicates that the file is probably corrupted, or the file is an improper type.

Effects of typing S(CR): The RPG program cycle is restarted, skipping all operations that would involve this record.

****UNSUCCESSFUL READ AT LINE xxxx, TYPE S(CR) TO IGNORE RECORD.

Reason: A READ operation from within the calculation routine has been attempted.

Suggested action: Determine if the file is in error. If not, utilize condition indicators to avoid this message.

Effects of typing S(CR): Program execution continues at the next line.