Note!

Before using this information and the product it supports, be sure to read the general information in the About This Publication section.

3rd Edition (March 1998)


See the Summary of Changes for the changes made to this publication. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

The IBM Printing Systems Company welcomes your comments. A form for reader's comments is provided at the back of this publication. If the form has been removed, you may send your comments to the following address:

INFORMATION DEVELOPMENT
THE IBM PRINTING SYSTEMS COMPANY
DEPARTMENT H7FE, BUILDING 003G
PO BOX 1900
BOULDER, CO 80301–9191

If you prefer to send comments electronically, use one of the following methods:

- Internet: printpubs@us.ibm.com
- Fax: 1-800-524-1519

Internet


When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1985, 1998. All rights reserved.

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Trademarks and Acknowledgements ........................................... xiii

About This Publication .............................................................. xv
Who Should Use This Publication? .............................................. xv
How This Publication Is Organized ............................................. xvi
Highlighting .............................................................................. xvi
PPFA Command Notation ............................................................ xvii
Related Publications .................................................................. xvii
  Publications about AIX ........................................................... xvii
  Publications about Advanced Function Presentation .................... xvii
  Publications about IBM Print Services Facility for AIX ................. xvii
  Publications about BCOC A ....................................................... xviii
  Publications about OS/400 ....................................................... xviii
  Publications about IBM Advanced Function Printing PrintSuite for OS/400 xviii
  Publications about VSE, MVS, and VM .................................... xviii

Summary Of Changes ................................................................ xix

Part 1. What is PPFA? ................................................................. 1

Chapter 1. Introducing Page Printer Formatting Aid ....................... 3
Summary of a Form Definition ..................................................... 4
Summary of a Page Definition ..................................................... 5
Formatting Output of Different Data File Types .............................. 6
  Line-Data Files ........................................................................ 6
  Mixed-Data Files .................................................................... 6
  MO:DCA-P Data Files ............................................................ 6
  Unformatted ASCII Files ....................................................... 7
PPFA Concepts .......................................................................... 7
  Physical Page ......................................................................... 7
  Logical Page ......................................................................... 7
  Subpage ................................................................................ 7
PPFA Basic Terms ..................................................................... 8
  Printline ............................................................................... 8
  Direction .............................................................................. 8
  Rotation ............................................................................... 9
  Presentation ......................................................................... 9
  N_UP Partitions .................................................................. 10
  Modifications ....................................................................... 10
Definitions of Command, Subcommand, and Parameter .................. 11
  Commands ........................................................................... 11
  Subcommands ..................................................................... 11
  Parameters .......................................................................... 11
Basic Controls in Line Data ........................................................... 11
  Carriage Control Characters ................................................... 11
  Table-Reference Characters ................................................... 11
Structured Fields in Line Data ..................................................... 12
  Invoke Data Map .................................................................. 12
  Invoke Medium Map ............................................................ 12

© Copyright IBM Corp. 1985, 1998
Part 2. Examples of Using PPFA

Chapter 2. Using Form Definition Commands

Copy Groups and Subgroups

Commands to Create a Form Definition

Command Nesting Rules

Positioning a Logical Page on a Sheet

OFFSET Subcommand with Rotated Print Direction

Specifying Copies and Electronic Overlays

Overlay Names

Printing Constant Forms

Duplex Printing

Duplex Printing in Portrait and Landscape Presentations

Specifying Page Presentation on Continuous-Forms Printers

When to Use the PRESENT and DIRECTION Subcommands

When the PRESENT and DIRECTION Subcommands Are Not Required

The DOWN Direction for the 3835 or 3900 Printer

3800 Coexistence and Migration

Print Quality Control

Chapter 3. Using Page Definition Commands

Page Formats within Page Definitions

Page Definition Command Nesting

Command Nesting Rules

Defining Logical Page Size

Positioning the First Line of Data

Changing Logical Page Print Direction

Printing Line Data on a PSF Printer

The OS/400 Environment

Processing Fields

POSITION Subcommand as Used in this Example

FIELD Command as Used in this Example

Color on the IBM InfoPrint HiLite Color Post Processor

Setup Verification

Varying Fonts on a Page

Printing Lines in Two Directions on a Page

Printing Fields in Two Directions on the Same Page

Rotating Fonts

Using Traditional Kanji Formatting

Printing Multiple-Up Pages

Chapter 4. Creating Complex Printouts

Combining Field Processing and an Electronic Overlay

Using Suppressions to Vary Data Presentation

Incorporating Fixed Text into a Page Definition

Combining Two Reports into One Printout

Chapter 5. Conditional Processing

General Description
# Figures

1. Form Definition and Page Definition Environment ........................................... 4
2. Baseline Direction and Inline Direction ............................................................. 9
3. Portrait and Landscape Presentations ................................................................. 10
4. Origin of Logical Page ......................................................................................... 20
5. Origin of a Logical Page on a 3900 Sheet ............................................................ 20
6. The Meaning of OFFSET Parameters within a Landscape Page ............................. 21
7. Two Electronic Overlays Incorporated into Two Subgroups .................................. 22
8. Six-Page Formatted Data File .............................................................................. 23
9. Result of Using a Pair of FRONT and BACK Subgroups ....................................... 24
10. Form Definition EFGH Using DUPLEX with BOTH ............................................. 25
11. DUPLEX NORMAL: Portrait and Landscape Presentation .................................... 27
12. Result When Either TUMBLE or RNORMAL Is Specified ..................................... 28
13. Narrow and Wide Continuous Forms .................................................................. 29
14. The Results of Not Specifying PRESENT LANDSCAPE and DIRECTION DOWN on a 3835 or 3900 Printer ................................................................. 30
15. The Results of Specifying PRESENT LANDSCAPE and DIRECTION DOWN on a 3835 or 3900 Printer ................................................................. 31
16. The Results of Not Specifying PRESENT and DIRECTION When Migrating from a 3800 to a 3835 or 3900 Printer ................................................................. 32
17. PRESENT/DIRECTION Combinations When Using the Same Forms Type on 3800, 3835, and 3900 Printers ................................................................. 33
18. Logical Page Dimensions ..................................................................................... 37
19. LINEONE Coordinates ....................................................................................... 38
20. Logical Page Print Directions in Relation to Origin ............................................. 41
21. Line-Data File ....................................................................................................... 42
22. Data File Printed on a Line Printer ....................................................................... 43
23. Printout Examples Specifying POSITION MARGIN TOP ....................................... 44
24. Printout Example Specifying POSITION MARGIN 4.1 ......................................... 45
25. Print Example Specifying POSITION MARGIN TOP and POSITION MARGIN 4.1 ................................................................................................................... 46
26. Unformatted Print Data File ................................................................................ 47
27. Data Arranged on the Printed Page ...................................................................... 47
28. Unformatted Print Data File ................................................................................ 49
29. Data Arranged on the Printed Page with Color ................................................... 50
30. Data File Printed Using a Single Font .................................................................. 51
31. Font Change Using TRCREF Command ................................................................ 52
32. Font Change Using FONT Commands and Subcommands ................................. 53
33. A Printout with More Than One Line Direction .................................................. 54
34. Field Direction ..................................................................................................... 55
35. Character Rotation ............................................................................................... 56
36. Example of Assumed Data File and Rotation Specifications ................................ 56
37. 3820 Tate Presentation ......................................................................................... 57
38. Multiple-Up Page Layout .................................................................................... 58
39. Multiple-Up Page Layout after Page Definition Modification ............................. 59
40. Electronic Overlay and Data File for a Sales Report ............................................ 62
41. Sales Report ......................................................................................................... 63
42. Selective Suppression ........................................................................................... 64
43. Input for the Corporate Version of an Individual Sales Report ............................ 66
44. The Corporate Version of the Sales Report with Fixed Text ............................... 68
45. Input for a New Report Produced from the Combined Data Files .................... 69
Tables

1. Form Definition Tasks ........................................ 17
2. Duplex Specifications ..................................... 28
3. Page Definition Tasks ...................................... 35
4. Form Definitions and Page Definition Tasks .............. 61
5. Conditional Processing Tasks ............................ 73
6. Character Length for PPFA Names ........................ 121
7. AS/400 Printer File Parameter Table ...................... 206
8. ANSI Carriage Control Characters ......................... 210
10. The Effect of Additive DIRECTIONs on Formatting and Font Prefixes . 220
11. Differences in Measurements and REPEATs with AFP Utilities .... 222
12. Valid Code Pages and Type Styles ...................... 223
13. Valid Characters and Data Lengths ..................... 224
Trademarks and Acknowledgements

The following terms appear in this publication and are either trademarks or registered trademarks of the IBM Corporation:

Advanced Function Common Control Unit
Advanced Function Presentation
Advanced Function Printing
AFCCU
AFP
AIX
AIX/6000
AS/400
Bar Code Object Content Architecture
BCOCA
IBM
Intelligent Printer Data Stream
IPDS
Mixed Object Document Content Architecture
MO:DCA
PrintManager
Print Services Facility
PSF
OS/400
RISC System/6000
S/370
About This Publication

This publication describes how to use the Page Printer Formatting Aid (PPFA) to create and compile page definitions and form definitions for printing or viewing files with Advanced Function Presentation products, such as IBM Print Services Facility (PSF).

Who Should Use This Publication?

This publication is for anyone who wants to use PPFA to create form definitions and page definitions. This publication has been written assuming that you are one of the following:

- A first-time user
  You are using PPFA for the first time to create form definitions and page definitions. You are familiar with system commands, but you are not familiar with Print Services Facility (PSF) concepts and Page Printer Formatting Aid parameters. You should read all of the information contained in this publication, and then use it as a reference.

  For more information about Advanced Function Presentation concepts, refer to *Guide to Advanced Function Presentation*.

  For more information about AIX concepts, refer to *IBM Print Services Facility for AIX: AIX for Users of Print Services Facility*.

  For more information about OS/400 concepts, refer to OS/400 User's Guide.

  For more information about VSE, MVS, VM, or OS/400, refer to the Application Programming Guide for the platform you are using.

- An intermediate user
  You are familiar with PSF concepts and with Page Printer Formatting Aid parameters and you know the difference between a logical page and a physical page. You already know how to create and use form definitions and page definitions. Use this publication as a reference to learn more about PPFA commands and syntax. Refer to the examples for useful information.

- An advanced user
  You understand PSF concepts and have used PPFA to create form definitions and page definitions. You understand the use of data stream processing. You will use this publication mostly as a reference. Chapter 4, Creating Complex Printouts might be especially helpful.

**Note:** Not all of the functions provided by PPFA are supported in all PSF licensed programs. Refer to the information for the PSF licensed program that you are using to determine which functions are supported. For more information about a specific environment, see Appendix A, System Dependencies for PPFA for the steps required to process page definitions and form definitions.
How This Publication Is Organized

You can use this publication both as a guide and as a reference to help you learn about the following:

- Chapter 1, Introducing Page Printer Formatting Aid summarizes PPFA and describes the purpose of form definitions and page definitions. Key PPFA concepts and terms are defined in this section.
- Chapter 2, Using Form Definition Commands shows examples illustrating the use of basic form definition controls.
- Chapter 3, Using Page Definition Commands shows examples illustrating the use of basic page-definition controls.
- Chapter 4, Creating Complex Printouts shows examples of print jobs that require advanced use of form-definition and page-definition controls.
- Chapter 5, Conditional Processing shows examples of conditional processing used in formatting complex printing applications.
- Chapter 6, N_UP Printing describes how you can use N_UP printing.
- Chapter 7, PPFA Command Syntax defines the rules and the syntax for writing a set of PPFA commands.
- Chapter 8, Form Definition Command Reference defines all the PPFA form-definition commands, their subcommands, and their parameters.
- Chapter 9, Page Definition Command Reference defines all the PPFA page-definition commands, their subcommands, and their parameters.
- Appendix A, System Dependencies for PPFA shows the steps needed to create and use form definitions and page definitions in VSE, MVS, VM, AIX, and OS/400 systems.
- Appendix B, More about Direction expands on the directions information giving a lookup table.
- Appendix C, Differences in Measurements and REPEATs with AFP Utilities describes the differences in printing with measurements and REPEATs between PPFA, OGL, and PMF.
- Appendix D, More About Bar Code Parameters contains supplemental information about bar codes.
- Appendix E, PPFA Messages and Codes lists all diagnostic messages generated by PPFA and suggests a cause and solution for each.

A glossary of terms and an index are included at the back of the publication.

Highlighting

This publication uses the following highlighting conventions:

**Bold** Identifies commands, keywords, files, directories, and other items whose names are predefined by the system, or items that must be entered as is, such as `man` and `enq`.

*Italic* Identifies parameters whose actual names or values you supply. Italics also identify the names of publications.
Monospace Identifies examples of specific data values, examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, messages from the system, or information you should actually type. Titles of PSF for AIX SMIT menus and panels also appear in monospaced type.

PPFA Command Notation

When you use PPFA commands, you specify them using the following notational conventions:

[ ] Brackets enclose optional subcommands or parameters

| Logical OR symbol separates mutually exclusive parameters:
THIS ─ THAT

{} Braces enclose mutually exclusive parameters: { THIS ─ THAT }

_____ Underscores (in boxed syntax) indicate defaults for commands and subcommands

... An ellipsis following a command or a set of commands indicates the command or set of commands can be repeated.

Related Publications

This section contains a list of publications, by category, that may be helpful to you as you use PPFA.

Publications about AIX

The following publication contains information about or related to AIX general concepts and procedures:

AIX and Related Products Documentation Overview SC23-2456

Publications about Advanced Function Presentation

The following publications contain information about IBM’s Advanced Function Presentation concepts and procedures:

Guide to Advanced Function Presentation, G544-3876
Advanced Function Presentation: Printer Information, G544-3290
Advanced Function Presentation: Printer Summary, G544-3135
Mixed Object Document Content Architecture Reference, SC31-6802
Advanced Function Presentation: Programming Guide and Line Data Reference, S544-3884

Publications about IBM Print Services Facility for AIX

The following publications contain information about IBM Print Services Facility for AIX, which uses the form definitions and page definitions created with PPFA:

IBM Print Services Facility for AIX: AIX for Users of Print Services Facility, G544-3877
IBM Print Services Facility for AIX: Print Administration, S544-3817
IBM Print Services Facility for AIX: Print Submission, S544-3878
IBM Print Services Facility for AIX: Advanced Function Presentation Conversion and Indexing Facility, S544-3930
Using IBM InfoPrint for Production Printing, S544-5473
Publications about BCOCA

The following publication contains information about bar code concepts relating to the PPFA BARCODE subcommand:

*Data Stream and Object Architectures:*
  *Bar Code Object Content Architecture Reference*, S544-3766

Publications about OS/400

The following publications contain information about AS/400, which uses the form definitions and page definitions created with PPFA:

*AS/400 Guide to AFP and PSF* S544-5319
*AS/400 Printer Device Programming* SC41-3713
*IBM AS/400 Printing IV* GG24-4389

Publications about IBM Advanced Function Printing PrintSuite for OS/400

The following publication contains PrintSuite information:

*AS/400 Advanced Utility User’s Guide* S544-5351
*IBM Advanced Function Presentation:*
  *Toolbox for AS/400 User’s Guide*, S544-5368

Publications about VSE, MVS, and VM

The following publications contain information about the general concepts and procedures for VSE, MVS, and VM environments:

*Print Services Facility/VSE: Application Programming Guide*, S544-3666
*Print Services Facility/VSE: System Programming Guide* S544-3665
*Print Services Facility/MVS: Application Programming Guide*, S544-3673
*Print Services Facility/MVS: System Programming Guide* S544-3672
*Print Services Facility/VM: Application Programming Guide*, S544-3677
*Print Services Facility/VM: System Programming Guide* S544-3680
Summary Of Changes

Changes between this edition and the previous edition are marked by a vertical bar "|" in the left margin.

This publication contains information pertaining to PPFA support for the OS/400 operating environment, as well as the AIX, MVS, VM, and VSE operating environments.

This publication also contains information about bar code support for the AIX, MVS, VM, and VSE operating environments. Bar code support is not provided for the OS/400 operating environment.

Because AS/400 messages can be viewed online, they are not reproduced in this document.
<table>
<thead>
<tr>
<th>Chapter 1. Introducing Page Printer Formatting Aid</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary of a Form Definition</td>
<td>4</td>
</tr>
<tr>
<td>Summary of a Page Definition</td>
<td>5</td>
</tr>
<tr>
<td>Formatting Output of Different Data File Types</td>
<td>6</td>
</tr>
<tr>
<td>Line-Data Files</td>
<td>6</td>
</tr>
<tr>
<td>Mixed-Data Files</td>
<td>6</td>
</tr>
<tr>
<td>MO:DCAP Data Files</td>
<td>6</td>
</tr>
<tr>
<td>Unformatted ASCII Files</td>
<td>7</td>
</tr>
<tr>
<td>PPFA Concepts</td>
<td>7</td>
</tr>
<tr>
<td>Physical Page</td>
<td>7</td>
</tr>
<tr>
<td>Logical Page</td>
<td>7</td>
</tr>
<tr>
<td>Subpage</td>
<td>7</td>
</tr>
<tr>
<td>PPFA Basic Terms</td>
<td>8</td>
</tr>
<tr>
<td>Printline</td>
<td>8</td>
</tr>
<tr>
<td>Direction</td>
<td>8</td>
</tr>
<tr>
<td>Rotation</td>
<td>9</td>
</tr>
<tr>
<td>Presentation</td>
<td>9</td>
</tr>
<tr>
<td>N_UP Partitions</td>
<td>10</td>
</tr>
<tr>
<td>Modifications</td>
<td>10</td>
</tr>
<tr>
<td>Definitions of Command, Subcommand, and Parameter</td>
<td>11</td>
</tr>
<tr>
<td>Commands</td>
<td>11</td>
</tr>
<tr>
<td>Subcommands</td>
<td>11</td>
</tr>
<tr>
<td>Parameters</td>
<td>11</td>
</tr>
<tr>
<td>Basic Controls in Line Data</td>
<td>11</td>
</tr>
<tr>
<td>Carriage Control Characters</td>
<td>11</td>
</tr>
<tr>
<td>Table-Reference Characters</td>
<td>11</td>
</tr>
<tr>
<td>Structured Fields in Line Data</td>
<td>12</td>
</tr>
<tr>
<td>Invoke Data Map</td>
<td>12</td>
</tr>
<tr>
<td>Invoke Medium Map</td>
<td>12</td>
</tr>
<tr>
<td>Include Page Segment</td>
<td>12</td>
</tr>
<tr>
<td>Include Page Overlay</td>
<td>12</td>
</tr>
<tr>
<td>Normal Duplex and Tumble Duplex</td>
<td>12</td>
</tr>
</tbody>
</table>
Chapter 1. Introducing Page Printer Formatting Aid

Page Printer Formatting Aid (PPFA) is an IBM licensed program that enables users of IBM's Advanced Function Presentation (AFP) products to create their own formatting resources, called form definitions and page definitions. The form definitions and page definitions are stored in libraries ¹ as AFP resources. Using AFP resources requires IBM Print Services Facility (PSF), a licensed program or feature, which merges resources with user data files. This merging creates a data stream for printing or viewing.

Using a form definition or a page definition created by PPFA requires you to perform three steps:

1. Write a set of PPFA commands that define how to position the data or handle the physical sheets.
2. Run PPFA to build the specified page definition or form definition and store the output as resources in a library.
3. Submit the print file using PSF, specifying the page definition and form definition needed to accomplish the desired results.

Figure 1 on page 4 shows how form definition and page definition relate to Print Services Facility (PSF). In Figure 1 on page 4, the area inside the broken line represents steps 1 and 2. The area outside of the broken line shows how PSF merges resources with the specified print job to form a single print stream and sends it to a page printer.

¹ For purposes of this book, the term “library” includes AIX directories as well as MVS, VM, VSE, and OS/400 libraries.
Summary of a Form Definition

A PPFA command stream can contain form-definition commands. A form definition specifies how the printer controls the processing of the physical sheets of paper. In a form definition, you can specify modifications that distinguish formatting one print job from another when both are derived from the same data. Form definitions are used for all PSF print files regardless of data type.

Form definitions can specify the following functions:
- Position of a logical page on a physical page
- Duplex printing
- Inclusion of overlays, which substitute for preprinted forms
- Flash (the use of a forms flash—only on 3800 printers)
- Selection of the number of copies for any page of data
- Suppression (the exclusion of selected fields of data in one printed version of a page of data but not in another)
- Jog (the offset stacking of cut-sheet output or copy marking on continuous-forms output)
- Selection among paper sources in a cut-sheet printer
- Adjustment of the horizontal position of the print area on the sheet (only on 3800 printers)
- Quality (selection among print quality levels)
- Constant (allows front or back printing of a page without variable data)
- Printing one, two, three, or four logical pages on a single side of a page
- Postprocessing controls, such as:
  - Selecting functions
  - Selecting device-dependent functions defined by the postprocessing device
- Perforating
- Cutting

Summary of a Page Definition

A page definition specifies how you want data positioned on the logical page. A page definition can control the following functions:

- Dimensions of the logical page
- Print direction of the logical page
- Print direction of lines and fields relative to the logical page
- Conditional processing (different formats on different pages, based on content of data)
- Line spacing (number of lines per inch)
- Location of individual lines and fields
- Number of lines per page
- Page segments for inclusion in printed output
- Overlays for inclusion in printed output (positioned anywhere on the page)
- Page-ejection points
- Fonts and font rotation used on a page
- Multiple-up printing (placing more than one subpage on one side of a single sheet)
- Colors to be used (on printers that support this function)
Formatting Output of Different Data File Types

The four basic types of data printed on PSF printers are:
- Line-data files
- Mixed-data files
- MO:DCA-P data files (called AFPDS in OS/400)
- Unformatted ASCII files (typically AIX)

Line-data files, mixed-data files, and unformatted ASCII require a page definition and a form definition. MO:DCA-P data files require only a form definition.

Line-Data Files

Line data is EBCDIC data that is arranged for printing on line printers. These records may contain line-printer control characters such as carriage control characters (CC or FCFC), table-reference characters (TRC), or only data.

The following example shows two types of line data. The first type shows data arranged as it prints out and the second shows data that requires field processing.

<table>
<thead>
<tr>
<th>Formatted Print Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROBINSON E. CRUSOE</td>
</tr>
<tr>
<td>123 ISLAND PLACE</td>
</tr>
<tr>
<td>KEY WEST, FL 33040</td>
</tr>
<tr>
<td>USA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Print output is same</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ PAGEDEF with line formatting</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Unformatted Print Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name field</td>
</tr>
</tbody>
</table>

The technique of mapping the unformatted data to locations on the output pages is known as field processing or record processing and is available through use of page-definition controls. Field processing is explained in detail in “Processing Fields” on page 46.

Mixed-Data Files

Mixed-data files consist of MO:DCA-P data and line data or unformatted ASCII data. Such files may or may not specify the beginning and ending of pages and may or may not contain page addresses and data controls for page printing. The line-data portion of such files must be formatted for page printers by page-definition controls.

MO:DCA-P Data Files

MO:DCA-P data files are formed into pages before PSF receives them. These files already contain the imbedded controls for printing on page printers. They contain such things as page addresses and data controls for page printing functions.

Note: Refer to Mixed Object Document Content Architecture Reference (SC31-6802) and Advanced Function Presentation Programming Guide and Line Data Reference (S544-3884) for more information about MO:DCA-P.
data. User application programs can also generate MO:DCA-P data. In OS/400, MO:DCA-P print files are created automatically when DEVTYPE=AFPDS in the Printer File.

Unformatted ASCII Files

Unformatted ASCII files consist of ASCII data with no formatting controls (escape sequences) in the data.

The technique of mapping the unformatted ASCII data to locations on the output pages is known as field processing or record processing and is available through use of page-definition controls. Field processing is explained in detail in “Processing Fields” on page 46.

Unformatted ASCII data differs from unformatted EBCDIC data in that ASCII data is what is generally created on a personal computer or workstation, while EBCDIC data is what is generally created on a mainframe host, such as MVS, VM, or VSE, or on AS/400.

PPFA Concepts

The concepts of physical page, logical page, and subpage are basic to understanding form-definition and page-definition controls.

Physical Page

A physical page is the sheet of paper or other medium (a sheet of labels, for instance) that moves through the printer.

Logical Page

A logical page is the area you define in a PPFA command stream as the space on the physical page where data is printed. The logical page is positioned in relation to the media origin. For more information about the media origin of your printer, refer to your printer documentation or the Advanced Function Presentation: Printer Information. The positioning of the logical page on the sheet of paper is described in “Positioning a Logical Page on a Sheet” on page 19.

An N_UP command enables you to place one, two, three, or four logical pages on a single sheet. This is in contrast to multiple up, which enables you to place subpages on one logical page.

Subpage

A subpage is a part of a logical page on which line data may be placed. Subpages are used only with conditional processing. Multiple-up printing can be done with or without subpages being defined. In the page definition, multiple subpages can be placed on the physical page based on changes in the print data. A good example of this is the use of multiple up printing, which is printing two or four pages on a single side of a sheet. For more information, see “Subpage Description and Processing” on page 77.
PPFA Basic Terms

The following terms have meanings that are special to PPFA:

- Printline
- Direction
- Rotation
- Presentation
- N_UP partitions
- Modifications

Printline

A printline is a single line of text. In the formatting of line data and unformatted ASCII, a printline is normally the output generated by one record in the print file. However, printlines and print records are not the same.

PRINTLINE commands in the PPFA page definition define the number and position of printlines on a page. Each record in the print file is written to a single printline on a page. Usually, one print record is written to each printline. However, control information in the print data can specify two or more print records be written to the same printline, providing overprinting. Controls also can specify that print records skip printlines. For example, a print record may skip the remaining printlines on a page and print instead on the first printline of a new page.

Direction

Text can be printed in four print directions. A print direction is a combination of both inline and baseline directions. For each of the directions, characters can be printed in four rotations.

The line direction is the direction in which successive characters are added to a line of text. The four line directions are:

- ACROSS
  Text characters are placed in a line from left to right across the page.
- DOWN
  Text characters are placed in a line from top to bottom down the page.
- BACK
  Text characters are placed in a line from right to left across the page.
- UP
  Text characters are placed in a line from bottom to top up the page.

The baseline direction is the direction in which successive lines of text are added to a page. The four character rotations, measured clockwise around each inline direction, for each line direction are:

- 0°
- 90°
- 180°
- 270°

For example, the text in this paragraph is printed ACROSS the page, and its rotation is 0°.
Figure 2 on page 9 shows the four possible directions. For information about the combinations supported by the printer you are using, refer to Advanced Function Presentation.

Printer Information

![Figure 2. Baseline Direction and Inline Direction](image)

**Rotation**

Individual characters can be rotated. Character rotation can be 0°, 90°, 180°, or 270° relative to the inline direction of the printline or field.

*Note:* On the 3800 printers only, character rotation differs between bounded-box fonts and unbounded-box fonts. Bounded-box fonts rotate the fonts; unbounded-box fonts are rotated by selecting the correct font.

**Presentation**

Presentation describes the shape of the page as it is viewed by the reader. Figure 3 on page 10 shows an example of how text is presented (positioned) on the page. There are two page presentations—portrait and landscape.

- **Portrait** is designed to be viewed with the short side at the top of the page.
- **Landscape** is designed to be viewed with the long side at the top of the page.
**N_UP Partitions**

Some printers allow the physical sheet of paper to be divided into equal-sized partitions. For two or three partitions, each sheet is divided along one or two lines equally spaced along the longer side of the sheet. The printer will position a logical page of print data in each partition. This enables printing multiple logical pages with different formats and modifications on a single sheet of paper.

The size and arrangement of the partitions on the sheet depends on the number of partitions and the shape and size of the paper. For two or three partitions, each sheet is divided at two or three points equally spaced along the longer side of the sheet. For four partitions, each sheet is equally divided both vertically and horizontally. See Chapter 6, “N_UP Printing” on page 97 for more information.

**Modifications**

*Modifications* are sets of form definition controls that apply to one page of a data file. With these controls, you can:

- Define the type of duplex printing to be done
- Define one, two, three, or four partitions for N_UP
- Select an overlay
- Suppress the appearance of a field
- Select the forms flash option (only for the 3800 printer)
- Specify the number of copies for a set of modifications
- Specify post-printing processing options

You can specify different sets of modifications for the same page of data in one form definition, and therefore in one print job, by a series of SUBGROUP commands. For example, a form definition with two SUBGROUP commands is said to have two sets of modifications. The same page of data is printed for each set of modifications, resulting in a slightly different output for each printing.
Definitions of Command, Subcommand, and Parameter

Commands, subcommands, and parameters are terms used throughout this publication to refer to the contents of PPFA control statements. Chapter 8, Form Definition Command Reference and Chapter 9, Page Definition Command Reference describe these commands with all their applicable subcommands.

Commands

Commands are the major controls composing form definitions and page definitions.

Subcommands

Subcommands are used to further define commands. The absence of subcommands means that the default values specified with those subcommands are used. Three command terms also appear as subcommand terms—FONT, OVERLAY, and SUPPRESSION. These subcommand terms further define other commands.

Parameters

You can specify parameters with subcommands or accept the defaults; valid entries and their defaults are shown in the command reference chapters.

Basic Controls in Line Data

The following line-printer controls may be included in a line data or unformatted ASCII file and can be used by a page definition to enable AFP functions:

- Carriage control characters
- Table-reference characters

Carriage Control Characters

Carriage control characters, which control line skipping, line spacing, and page ejection on line printers, are fields within line-data and unformatted-ASCII records. They are compatible with page printers when page definitions format the printed data. In page definitions, you can specify CHANNEL subcommands that correspond to carriage control characters corresponding to channels 1 through 12 in the data. When you do so, the carriage control characters operate just as they do in a line-printer environment.

Note: ASCII ANSI, ANSI, and EBCDIC (machine) handle carriage control characters differently. See the SPACE_THEN_PRINT subcommand listed in “Subcommands” on page 153 for more information.

Table-Reference Characters

Table-reference characters (TRCs) control font selection in line-data and unformatted-ASCII output. Page definitions can be used to map table-reference characters to AFP fonts for use with page printers.
Structured Fields in Line Data

To make use of the full function of page definitions and form definitions, MO:DCA-P structured fields may be required in the user's data. The following MO:DCA-P structured fields can be included in a line-data or unformatted ASCII file (typically AIX) to activate AFP functions:

- Invoke Data Map
- Invoke Medium Map
- Include Page Segment
- Include Page Overlay

Invoke Data Map

Add the Invoke Data Map structured field to the line-data or unformatted ASCII file at a point that requires switching from one page format to another. The term “data map” is the name used for the term “page format” in PSF publications and PSF terminology.

Invoke Medium Map

Add the Invoke Medium Map structured field to the line-data or unformatted-ASCII file at a point that requires switching from one copy group to another. The term “medium map” is the name used for the term “copy group” in PSF publications and PSF terminology.

Include Page Segment

Position the Include Page Segment structured field within the line or unformatted ASCII data for placing the page segment on the page.

Include Page Overlay

Position the Include Page Overlay structured field within the line or unformatted ASCII data for placing the overlay anywhere on the page.

Normal Duplex and Tumble Duplex

Some page printers can print on both sides of a sheet, which is called duplex printing. Duplex printing can be done in four ways:

- Normal duplex
- Tumble duplex
- Rotated normal duplex
- Rotated tumble duplex

In normal duplex, both sides have the same orientation, as in most books. In tumble duplex, the back of each page is upside down with respect to the front of the page: the top of one side of the sheet is at the same edge as the bottom of the other side. These two types of duplex allow you to specify top binding or side binding of the printed pages.

Duplex also involves the commands RNORMAL (rotated normal) and RTUMBLE (rotated tumble), which are used with landscape-presentation pages to specify the type of duplex printing. See Figure 11 on page 27 and Figure 12 on page 28 for illustrations of duplex printing.
# Part 2. Examples of Using PPFA

## Chapter 2. Using Form Definition Commands
- Copy Groups and Subgroups .................................. 17
- Commands to Create a Form Definition ....................... 18
  - Command Nesting Rules .................................. 19
- Positioning a Logical Page on a Sheet ....................... 19
- OFFSET Subcommand with Rotated Print Direction .......... 20
- Specifying Copies and Electronic Overlays ................ 21
  - Overlay Names ........................................ 21
- Printing Constant Forms .................................. 22
- Duplex Printing .......................................... 23
- Duplex Printing in Portrait and Landscape Presentations 26
- Specifying Page Presentation on Continuous-Forms Printers 28
  - When to Use the PRESENT and DIRECTION Subcommands 29
  - When the PRESENT and DIRECTION Subcommands Are Not Required 29
  - The DOWN Direction for the 3835 or 3900 Printer ........ 30
- 3800 Coexistence and Migration ........................... 31
- Print Quality Control ................................... 33

## Chapter 3. Using Page Definition Commands
- Page Formats within Page Definitions ....................... 35
- Page Definition Command Nesting .......................... 36
  - Command Nesting Rules .................................. 36
- Defining Logical Page Size ................................ 36
- Positioning the First Line of Data ........................... 37
- Changing Logical Page Print Direction ....................... 39
- Printing Line Data on a PSF Printer ......................... 41
  - The OS/400 Environment .................................. 46
- Processing Fields ........................................ 46
  - POSITION Subcommand as Used in this Example ........... 48
  - FIELD Command as Used in this Example ................ 48
- Color on the IBM InfoPrint HiLite Color Post Processor .. 48
  - Setup Verification ........................................ 50
- Varying Fonts on a Page ................................ 50
- Printing Lines in Two Directions on a Page ................ 54
- Printing Fields in Two Directions on the Same Page ....... 54
- Rotating Fonts ........................................... 55
- Using Traditional Kanji Formatting ......................... 57
- Printing Multiple-Up Pages ................................ 57

## Chapter 4. Creating Complex Printouts
- Combining Field Processing and an Electronic Overlay .... 61
- Using Suppressions to Vary Data Presentation ............... 63
- Incorporating Fixed Text into a Page Definition ............. 65
- Combining Two Reports into One Printout .................... 68

## Chapter 5. Conditional Processing
- General Description ....................................... 73
  - Using Conditional Processing versus Normal Line Data Processing 73
  - Using Conditional Processing to Set Up the Environment 74
  - Selecting a Copy Group ................................. 74
Medium Overlays and Page Overlays .................................. 114
N_UP Compared to Multiple-up ....................................... 115
Chapter 2. Using Form Definition Commands

A form definition is a resource, used by PSF, that specifies how the printer controls the processing of the sheets of paper. With form definitions, you can perform the tasks listed in Table 1.

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Location of Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating a form definition</td>
<td>“Commands to Create a Form Definition” on page 18</td>
</tr>
<tr>
<td>Positioning a logical page</td>
<td>“Positioning a Logical Page on a Sheet” on page 19</td>
</tr>
<tr>
<td>Specifying landscape presentation</td>
<td>“OFFSET Subcommand with Rotated Print Direction” on page 20</td>
</tr>
<tr>
<td>Specifying copies and electronic overlays</td>
<td>“Specifying Copies and Electronic Overlays” on page 21</td>
</tr>
<tr>
<td>Printing constant forms</td>
<td>“Printing Constant Forms” on page 22</td>
</tr>
<tr>
<td>Duplex printing in two orientations</td>
<td>“Duplex Printing” on page 23</td>
</tr>
<tr>
<td>Printing portrait and landscape</td>
<td>“Duplex Printing in Portrait and Landscape Presentations” on page 26</td>
</tr>
<tr>
<td>Specifying the page presentation on</td>
<td>“Specifying Page Presentation on Continuous-Forms Printers” on page 28</td>
</tr>
<tr>
<td>continuous-forms printers</td>
<td></td>
</tr>
<tr>
<td>Migrating from 3800 printers to other</td>
<td>“3800 Coexistence and Migration” on page 31</td>
</tr>
<tr>
<td>IPDS printers</td>
<td></td>
</tr>
</tbody>
</table>

Copy Groups and Subgroups

A single form definition can contain several subsets of page controls, called *copy groups*. Copy groups define each physical page in the file. When you are printing jobs in duplex, the copy group defines both sides of the physical paper. Copy groups, in turn, can contain up to 127 *subgroups*, each of which creates a different set of modifications for the same page of data.

A series of copy groups can be used where either the data or the printing requirements call for a variety of page control schemes. Part of the file can be printed from one (bin) paper source and part from another. Part can be printed duplex; part can be printed simplex. Duplex commands can be specified for a printer that does not support this function. This command treats the two adjacent pages as duplexed. A variety of controls can be contained in one form definition having several copy groups.

You can control the following options within a copy group:

- Position of the logical page on a sheet of paper
- Duplex printing
- Type of cut-sheet paper to be printed on (by choosing between paper input sources in page printers that have more than one paper source)
- Offset stacking or copy marking of parts of a print job in the output stacker
• Printing one, two, three, or four logical pages on a single side of a sheet
• Vendor-attached devices for post-processing functions to be performed on the sheet
• Print-quality level

To access a new copy group within a form definition you can:

• Add to your data file an Invoke Medium Map structured field immediately before the page of data that requires the new copy group.
• Use a page definition that specifies conditional processing. When you access a new copy group, printing begins on the next physical sheet of paper.

For more information on the Invoke Medium Map structured field, refer to *Mixed Object Document Content Architecture Reference*.

Subgroups allow the same page of data within a file to be printed more than once, using different sets of modifications each time the page is printed. One example is the printing of an invoice and a packing list from the same records in a data file.

The following modifications to the page of data can be specified in a subgroup:

• Selection of suppressed fields for the page
• Selection of overlays used with the page
• Selection of forms flash with the page (only on the 3800 printer)
• Selection of the modification for front, back, or both sides of a sheet
• Selection of the number of copies of the subgroup to print
• Selection of the input bin

### Commands to Create a Form Definition

The following simplified command stream shows the proper nesting of commands and the sequence in which the commands must be entered when you are creating a form definition:

```
[SETUNITS]
FORMDEF
  [SUPPRESSION]
  [COPYGROUP]
  [OVERLAY]
  [SUBGROUP]
```

**Notes:**

1. Brackets enclosing a command mean the command is optional.
2. A command including its subcommands ends with a semicolon.
3. Indentations are used to improve readability.
4. Complete definitions of commands are in Chapter 8, “Form Definition Command Reference” on page 125.
Command Nesting Rules

1. SUPPRESSION commands must be specified immediately after FORMDEF commands.

2. SUBGROUP commands are specified under their associated COPYGROUP command or under the FORMDEF command.

3. OVERLAY commands are specified immediately after COPYGROUP commands.

4. The first COPYGROUP command can be omitted in a form definition if the form definition has only one copy group, and if it contains no OVERLAY commands.

5. A SETUNITS command can be placed anywhere in the PPFA command stream and is in effect until another SETUNITS command is encountered.

6. More than one of each command can appear under one form definition.

Positioning a Logical Page on a Sheet

The example in this section shows how the OFFSET subcommand is used to position the logical page on the physical sheet. A logical page is the area on a sheet of paper where all printing occurs. You establish the logical page origin, the point nearest the media origin, with the OFFSET subcommand. The OFFSET subcommand requires two coordinates and may have four. The first x and y coordinate defines the position on the front of the sheet, and the second x and y coordinate defines the position on the back of the sheet. A sample form definition that specifies the logical page position for a simplex sheet is:

FORMDEF ABCD
  OFFSET 1 IN 1 IN ;

Note: The 1 IN 1 IN is an abbreviation for 1 INCH 1 INCH. PPFA supports a number of different units of measurement formats. See "Units of Measurement" on page 123 for all the different formats.

The example places the logical page origin one inch to the right of and one inch down from the media origin.

Figure 4 on page 20 shows the meaning of the x and y coordinates. In writing an OFFSET subcommand, the first parameter specifies x; the second parameter specifies y. If the x and y are repeated for the offset of the back side of the physical page, the same applies. The x defines the horizontal offset; the y defines the vertical offset. In this example, the logical page direction is ACROSS. The arrows within the logical page indicate the inline direction for text on the page. The lines of text are added according to the baseline direction.
Figure 4. Origin of Logical Page

Figure 5 shows the meaning of $x$ and $y$ in a logical page specification for a 3900 sheet. The 3900 sheet does not have an unprintable area, but FORMDEFs supplied with PSF have a 1/6 inch offset.

OFFSET Subcommand with Rotated Print Direction

Figure 6 on page 21 shows that the media origins and logical page origins do not change when the print direction of the page changes, although the way you view the page does change. The arrows within the logical page show the DOWN print direction—producing landscape page presentation.

Be careful to coordinate form definitions and page definitions when you change between portrait and landscape presentations.
Specifying Copies and Electronic Overlays

This example shows how to specify different electronic overlays in different subgroups. The electronic overlays you specify are created separately, using a program such as IBM Overlay Generation Language/370 or the AFP Utilities/400, and are stored as resources in the overlay library. No positioning controls are needed in the form definition with an overlay; the overlays are merely named. The overlay contains its own positioning data relative to the physical sheet. A form definition containing two overlays might look like this:

```
FORMDEF SLSCOM ;
COPYGROUP SLSCOM ;
  OVERLAY SLSRPT M1001 ; /*LOCAL NAME AND USER-ACCESS NAME*/
  OVERLAY M1002 ; /*USER-ACCESS NAME ONLY*/
SUBGROUP COPIES 2
  OVERLAY SLSRPT ;
SUBGROUP COPIES 3
  OVERLAY M1002 ;
```

The steps to write this form definition are:

1. Create a copy group.
   a. Write a COPYGROUP command.
   b. Write an OVERLAY command for each overlay.

2. Create two subgroups by writing two SUBGROUP commands. Each subgroup contains an OVERLAY subcommand naming one of the selected overlays.

   **Note:** The overlays must be named in each copy group.

Overlay Names

To identify overlays by name, you must be aware of the three possible names for an overlay: a local name (SLSRPT) and two system names (M1001, O1M1001). The local name is used only within the PPFA command stream; its use is optional. An example of this is SLSRPT in the first OVERLAY command of the previous sample command stream.
The system name identifies an overlay in the library. It has two forms: the user-access name (M1001 in the sample set of commands) and the library-resource name. Of these, you use only the user-access name. PPFA automatically adds the O1 overlay prefix to the user-access name, which identifies the resource in the library. An overlay referenced through a form definition built with PPFA, therefore, must begin with the O1 prefix. An example of the result is O1M1001, the library-resource name.

You can make up your own local name for an overlay. However, the local name must be used in the OVERLAY subcommand in the subgroup if it is used in an OVERLAY command for the copy group. If it is not, the subgroup must specify the user-access name, as has been done for overlay M1002 in the example.

This example, specifying copies and electronic overlays, also specifies the number of copies of each subgroup. More than one copy of printed output can be requested by placing the COPIES subcommand and the number of copies of the subgroup desired in the SUBGROUP command. This example specifies that two copies of the first subgroup and three copies of the second subgroup are to be printed.

Figure 7 shows the result of printing a job including the overlays as specified in the sample command stream at the beginning of this example.

---

**Printing Constant Forms**

This example shows how to specify the constant-forms function using the CONSTANT command. The constant-forms function allows you to print overlays or a forms flash on blank pages without adding blank pages to your print job. Instead, the CONSTANT command generates blank pages on which to print the requested overlays and forms flash. These pages are called constant forms because no variable data from the print file is printed on the pages.

You specify the CONSTANT command for an entire copy group; you identify the overlays and forms flash in the subgroups of the copy groups.
The sample form definition XMPXXX shown below specifies that overlay XMP be printed on the back of each sheet with no variable data from the print job. The data from the print file is printed only on the front side of each sheet.

```
FORMDEF XMPXXX
   REPLACE YES
   DUPLEX NORMAL ;
   COPYGROUP XMPXXY
      CONSTANT BACK ;
      OVERLAY XMP;
   SUBGROUP FRONT ;
   SUBGROUP BACK
      OVERLAY XMP;

PAGEDEF XMPXXX
   REPLACE YES ;
   FONT NORMALFONT GMT10 ;
   PAGEFORMAT XMPXXX ;
      PRINTLINE CHANNEL 1 REPEAT 20
         POSITION 1 1 ;
```

The steps to write this form definition are:
1. Create a copy group.
   a. Specify duplex printing.
   b. Specify printing of a constant form as the back side of each sheet.
   c. Write an OVERLAY command.
2. Create two subgroups by writing two SUBGROUP commands. The subgroup for the back side specifies the overlay to be printed.

**Note:** If you do not specify an overlay in the subgroup for the back, the back side of each sheet will be blank.

---

**Duplex Printing**

Printing on both sides of a sheet (duplex printing) can be done in two ways: by the use of the FRONT and BACK subcommand combination or by the use of the BOTH subcommand. If FRONT and BACK are chosen, the number of copies requested for each must be the same.

To demonstrate some of the functions available for duplex printing, assume you want to print a six-page data file (a simplified version is shown in Figure 8).

```
<table>
<thead>
<tr>
<th>Page 1</th>
<th>Page 2</th>
<th>Page 3</th>
<th>Page 4</th>
<th>Page 5</th>
<th>Page 6</th>
</tr>
</thead>
</table>
```

*Figure 8. Six-Page Formatted Data File*
Assume, too, that the file is already composed and formatted, so only a form definition is needed. The first form definition follows:

```plaintext
FORMDEF ABCD
  DUPLEX NORMAL ;
  OVERLAY AB ;
  SUBGROUP FRONT
    OVERLAY AB ;
  SUBGROUP BACK ;
```

In this command stream, form definition ABCD contains two subgroups, one specified with a FRONT subcommand and the other with a BACK subcommand.

By including a pair of FRONT and BACK subcommands within the copy group, you can specify that the front and back of printed sheets are to be controlled by different subgroups. The purpose of this is to allow modifications (overlays or suppressions, for example) to be separately specified for the front and back of sheets. Figure 9 shows the result of using this control where the front sheets have a header (OVERLAY AB) that the backs do not have.

![Figure 9. Result of Using a Pair of FRONT and BACK Subgroups](image)

The rules of the FRONT and BACK subcommands are:

- FRONT and BACK subgroups must be specified in pairs.
- Subgroups specifying FRONT must always immediately precede subgroups specifying BACK.
- FRONT and BACK subgroups must agree in the number of copies.

The BOTH subcommand also can be used with a form definition or a copy group that specifies duplex printing. An example of this type of form definition is:

```plaintext
FORMDEF EFGH
  DUPLEX NORMAL ;
  SUBGROUP BOTH
    COPIES 2 ;
```
The form definition EFGH contains only one SUBGROUP command.

Notes:

1. The copy group actually contains the subgroup, but if a form definition contains only one copy group, the copy group need not be specified.

2. With the BOTH subcommand, you specify only one subgroup: both sides of all sheets have the same modifications.

3. The above form definition does not put the same data on the front and back of the same sheet. Internally to PPFA, a single BOTH subgroup actually produces two subgroups. As a result, two pages of data (one for each internal subgroup) are processed before copy number 2 is made. For more information about this topic, see “SUBGROUP Command” on page 146.

Figure 10 shows a sample print resulting from using the FORMDEF EFGH specifying BOTH to control the printing of the six-page (2 copies) data file.
Duplex Printing in Portrait and Landscape Presentations

Duplex printing with PPFA and PSF printers offers several other options. This example shows the combination of portrait and landscape presentations with normal and tumble duplex printing.

Note: The terms normal, tumble, portrait, and landscape are used in this example. They are explained in this chapter and in the Glossary.

NORMAL and TUMBLE are parameters of a DUPLEX subcommand.

For example, a form definition specifying DUPLEX NORMAL could be written this way:

FORMDEF ABCD ;
COPYGROUP ABCD
   DUPLEX NORMAL ;
SUBGROUP BOTH
   COPIES 1 ;

Document A in Figure 11 on page 27 shows the result of a DUPLEX NORMAL specification in the portrait presentation. Document B shows the result of the same form definition when a landscape presentation is specified. The printout in landscape presentation is really in a tumble-duplex format, having the tops (of the front side) and the bottoms (of the back side) of the logical pages toward the same edge of the sheet.

Although tumble duplex can be specified in this manner for landscape pages, another parameter, RTUMBLE (rotated tumble), exists to make the form definition look more sensible for use in landscape print jobs. It also produces the results shown in Figure 11, depending on whether the form definition called for portrait or landscape presentation. For landscape, the form definition should be written as follows:

FORMDEF ABCD
   PRESENT LANDSCAPE ;
COPYGROUP ABCD
   DUPLEX RTUMBLE ;
SUBGROUP BOTH
   COPIES 1 ;

Note: The example presented is for continuous printers. You must use N_UP for cut-sheet printers. In Chapter 8, “Form Definition Command Reference” on page 125, see the PRESENT subcommand of COPYGROUP.
The DUPLEX NORMAL and DUPLEX RTUMBLE controls actually produce the same result on the physical page. RTUMBLE is used to maintain an association between duplex specifications and logical page print direction. The same relationship exists between the RNORMAL and the TUMBLE parameters as exists between the NORMAL and the RTUMBLE parameters; that is, within the two sets the terms are interchangeable.

For example, you could write a form definition using DUPLEX TUMBLE as follows:

```
FORMDEF DEFG ;
COPYGROUP DEFG
   DUPLEX TUMBLE ;
SUBGROUP BOTH
   COPIES 1 ;
```

Documents C and D in Figure 12 on page 28 are the results, depending on how page definition direction is specified to achieve either a portrait page or a landscape page.
To help you remember, use Table 2.

<table>
<thead>
<tr>
<th>Table 2. Duplex Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>If the form definition duplex specification is . . .</td>
</tr>
<tr>
<td>DUPLICATE NORMAL</td>
</tr>
<tr>
<td>DUPLICATE RTUMBLE</td>
</tr>
<tr>
<td>DUPLICATE TUMBLE</td>
</tr>
<tr>
<td>DUPLICATE RNORMAL</td>
</tr>
</tbody>
</table>

Note: Other control combinations are not recommended.

Specifying Page Presentation on Continuous-Forms Printers

This example shows how to specify the page presentation (portrait or landscape) on printers that use continuous-forms paper. The page presentation is specified in the form definition using the PRESENT subcommand in conjunction with the DIRECTION subcommand.

The PRESENT subcommand specifies how your pages will be presented when they are printed and has two valid values: PORTRAIT and LANDSCAPE.

The DIRECTION subcommand specifies the inline direction in which your pages have been formatted by the page definition (see “FIELD Command” on page 158) or by the program formatting the data. The DIRECTION subcommand has two valid values: ACROSS and DOWN.
The conditions in which you should use these subcommands and some conditions in which they are not required are described below. For more information about how these subcommands work with data sent to specific printers, refer to the appropriate printer documentation.

In order to understand the description that follows, you must be aware of the difference between the two types of continuous forms: narrow and wide. Narrow forms are forms that have perforations on the shorter edge of the paper and tractor holes on the longer edge. Wide forms are forms that have perforations on the longer edge of the paper and tractor holes on the shorter edge. The two types of forms are illustrated in Figure 13.

![Figure 13. Narrow and Wide Continuous Forms](image)

**When to Use the PRESENT and DIRECTION Subcommands**

You should use the PRESENT and DIRECTION subcommands if you are building a form definition that will be used:

- With wide forms on an IBM 3835 or 3900 Page Printer when the print data has been formatted in the DOWN print direction (see “The DOWN Direction for the 3835 or 3900 Printer” on page 30)
- When you do not know which type of form (narrow or wide) will be used on a 3835 or 3900 printer (see “The DOWN Direction for the 3835 or 3900 Printer” on page 30)
- To print data formatted for a 3800 printer on a 3835 or 3900 printer (see “3800 Coexistence and Migration” on page 31)
- To migrate data previously printed on a 3800 printer to a 3835 or 3900 printer (see “3800 Coexistence and Migration” on page 31)

**Note:** References to the IBM 3835 Page Printer point of origin also apply to all PSF continuous-forms printers except the 3800.

**When the PRESENT and DIRECTION Subcommands Are Not Required**

You do not need to use the PRESENT and DIRECTION subcommands if you are building a form definition that will be used:

- With cut-sheet printers only
- With narrow forms only
• With the 3800 printer only
• With print data that has been formatted in the BACK direction by the page definition or the program formatting the data

The DOWN Direction for the 3835 or 3900 Printer

If your data has been formatted in the DOWN print direction for landscape page presentation and is to be printed on wide forms on a 3835 or 3900 printer, you must specify LANDSCAPE on the PRESENT subcommand to produce readable output.

If PRESENT LANDSCAPE and DIRECTION DOWN are not specified on the FORMDEF command, the data will be printed in the landscape presentation; however, the data will be upside down, as shown in Figure 14. The data is upside down in this case because the media origin for the 3835 or 3900 printer is located on the same corner of the form, regardless of whether a narrow or wide form is being used (see Figure 14).

If PRESENT LANDSCAPE and DIRECTION DOWN are specified on the FORMDEF command, the data will be printed as shown in Figure 15. In this example, line data is formatted using a page definition.

PRESENT LANDSCAPE and DIRECTION DOWN can also be specified for data formatted in the DOWN print direction that will be printed on narrow forms. Although PRESENT LANDSCAPE and DIRECTION DOWN do not need to be specified in this case in order to produce readable output, specifying them will enable you to use the same form definition regardless of whether the data will be printed on wide forms or narrow forms.

Note: If you are building a form definition that can be used with both wide and narrow forms, remember that the left margin as viewed by the reader will become the top margin from the printer’s perspective (and vice versa). Because many printers have an unprintable area at the margins, you should
To position the logical page using the OFFSET subcommand in the form definition, so data will not be placed in the unprintable area on either wide or narrow forms.

```
PAGEDEF XYZ
DIRECTION DOWN;
```

```
FORMDEF XYZ
PRESENT LANDSCAPE
DIRECTION DOWN;
```

**Wide Form**

```
ABCD
```

**Narrow Form**

```
ABCD
```

*Figure 15. The Results of Specifying PRESENT LANDSCAPE and DIRECTION DOWN on a 3835 or 3900 Printer*

### 3800 Coexistence and Migration

The PRESENT and DIRECTION subcommands should be used if you are doing either of the following:

- Building a form definition that will be used to print data formatted for the 3800 on the 3800, 3835, and the 3900 printers
- Migrating data formatted for the 3800 printer to the 3835 or 3900 printer
- Migrating data formatted for an impact printer

If the PRESENT and DIRECTION subcommands are not specified, the print data may exceed the valid printable area on the 3835 or 3900 printer, as shown in Figure 16 on page 32. The data exceeds the valid printable area in this case because the media origin of the 3835 or 3900 printer is different from the media origin of the 3800 printer.
When you are building a form definition for migration or coexistence from the 3800 to the 3835 or 3900, code the PRESENT and DIRECTION subcommands as follows:

1. If you have a 3800 wide-forms application formatted in the ACROSS or UP direction that you want to print on wide or narrow forms on a 3835 or 3900, specify PRESENT LANDSCAPE and DIRECTION ACROSS.

2. If you have a 3800 wide-forms application formatted in the DOWN direction that you want to print on wide or narrow forms on a 3835 or 3900, specify PRESENT PORTRAIT and DIRECTION DOWN.

3. If you have a 3800 narrow-forms application formatted in the ACROSS direction that you want to print on narrow or wide forms on a 3835 or 3900, specify PRESENT PORTRAIT and DIRECTION ACROSS.

4. If you have a 3800 narrow-forms application formatted in the DOWN direction that you want to print on narrow or wide forms on a 3835 or 3900, specify PRESENT LANDSCAPE and DIRECTION DOWN.

**Note:** You can only specify ACROSS and DOWN on the DIRECTION subcommand when you are building a form definition for 3800 to 3835 or 3900 migration or coexistence.

Figure 17 on page 33 illustrates the output on 3800, 3835, or 3900 printers with all four combinations of PRESENT and DIRECTION. These illustrations assume that the type of forms were not changed from one printer type to another.

The PRESENT and DIRECTION subcommands should also be used if you want to print data formatted for the 3800 printer on a different type of forms (narrow versus wide) on the 3835 or 3900 printer. The combinations of PRESENT and DIRECTION to use when performing this type of migration are included in Line Item 2, as previously listed.

**Note:** When you migrate an application from one type of forms to another, remember that the top and left margins from the printer’s perspective will change places. Because many printers have an unprintable area at the margins, you should position the logical page using the OFFSET subcommand in the form definition, so data will not be placed in the unprintable area on the forms you are migrating to.
If you want output formatted for the 3800 to look like this on the 3835 or 3900:

Specify:  
Or use this IBM supplied FORMDEF:

```
FORMDEF ABCD
PRESENT LANDSCAPE
DIRECTION ACROSS ;
```

```
FORMDEF ABCD
PRESENT PORTRAIT
DIRECTION DOWN ;
```

```
FORMDEF ABCD
PRESENT PORTRAIT
DIRECTION ACROSS ;
```

```
FORMDEF ABCD
PRESENT LANDSCAPE
DIRECTION DOWN ;
```

*Figure 17. PRESENT/DIRECTION Combinations When Using the Same Forms Type on 3800, 3835, and 3900 Printers*

**Print Quality Control**

If your printer has more than one print-quality selection, you can specify different levels of print quality. For more information refer to the manual for your printer.
Chapter 3. Using Page Definition Commands

A page definition specifies how you want data positioned on the logical page.

A page definition is a resource used by PSF that defines the rules of transforming line data and unformatted ASCII into composed pages and text controls for printing. With page definitions, you can perform the tasks listed in Table 3.

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Location of an Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating a page definition</td>
<td>“Page Definition Command Nesting” on page 36</td>
</tr>
<tr>
<td>Defining logical page size</td>
<td>“Defining Logical Page Size” on page 36</td>
</tr>
<tr>
<td>Positioning data on a logical page</td>
<td>“Positioning the First Line of Data” on page 37</td>
</tr>
<tr>
<td>Changing the print direction</td>
<td>“Changing Logical Page Print Direction” on page 39</td>
</tr>
<tr>
<td>Printing line data</td>
<td>“Printing Line Data on a PSF Printer” on page 41</td>
</tr>
<tr>
<td>Processing fields</td>
<td>“Processing Fields” on page 46</td>
</tr>
<tr>
<td>Changing fonts</td>
<td>“Varying Fonts on a Page” on page 50</td>
</tr>
<tr>
<td>Printing in different directions</td>
<td>“Printing Lines in Two Directions on a Page” on page 54</td>
</tr>
<tr>
<td>Printing fields in two directions</td>
<td>“Printing Fields in Two Directions on the Same Page” on page 54</td>
</tr>
<tr>
<td>Rotating fonts</td>
<td>“Rotating Fonts” on page 55</td>
</tr>
<tr>
<td>Printing kanji</td>
<td>“Using Traditional Kanji Formatting” on page 57</td>
</tr>
<tr>
<td>Printing multiple up</td>
<td>“Printing Multiple-Up Pages” on page 57</td>
</tr>
</tbody>
</table>

Page Formats within Page Definitions

Just as form definitions can include more than one copy group, page definitions can include several page formats. Page formats use the same subcommands (except REPLACE) as page definitions, and if a subcommand is specified in a page format, it overrides the value specified in the page definition for the page format. A single page definition may contain multiple page formats. If pages in a file are to be formatted differently, specify more than one page format in your page definition. Within a page definition, page formats are generated in the order in which they are specified.

Using more than one page format to control different pages requires one of the following:

- Adding the Invoke Data Map structured field to the data file each time you want to change page formats
- Using conditional processing.
Refer to Advanced Function Presentation: Programming Guide and Line Data Reference for more information about the Invoke Data Map structured field.

Page Definition Command Nesting

The following simplified command stream shows the proper nesting of commands and the order in which they must be entered when you create a page definition:

```
[ SETUNITS]
PAGEDEF
  [FONT]
  [PAGEFORMAT]
    [TRCREF]
    [SEGMENT]
    [OVERLAY]
  [PRINTLINE
    [FIELD]
      [CONDITION]
  [ENDSUBPAGE]
  [SETUNITS]
```

Notes:

1. Brackets enclosing a command mean the command is optional.
2. A command and its subcommands end with a semicolon.
3. Indentations are used to improve readability.
4. Complete definitions of all commands are included in Chapter 9, “Page Definition Command Reference” on page 151.

Command Nesting Rules

1. FONT commands must be specified immediately after PAGEDEF commands.
2. A SETUNITS command can be specified anywhere in the PPFA command stream and is in effect until another SETUNITS command is specified.
3. TRCREF, SEGMENT, and OVERLAY commands must be specified under their associated PAGEFORMAT command.
4. The first PAGEFORMAT command can be omitted in a page definition, if the page definition has only one page format.
5. At least one PRINTLINE command is required.

Defining Logical Page Size

“Positioning a Logical Page on a Sheet” on page 19 shows how to establish the origin point of a logical page, relative to the media origin on a sheet of paper, using the OFFSET subcommand. The following example shows you how to establish the width and height of the logical page relative to this origin point. This example illustrates how the dimensions of a logical page are determined by form definitions and page definitions.
FORMDEF ABCD
OFFSET (1)(2) ;
PAGedef ABCD
WIDTH (3)
HEIGHT (4) ;
PRINTLINE ;

Note: The parenthetical numbers represent dimensions. Figure 18 shows how these dimensions relate to the logical page.

Normally, all parameters consist of a number and a unit of measurement, for example, 6 IN. (See “Units of Measurement” on page 123 for information on units that are available.) Numbers can be specified with up to three decimal places. The PRINTLINE command is included because at least one is required for all page definitions; see “PRINTLINE Command” on page 178 for more information.

![Figure 18. Logical Page Dimensions](image)

The OFFSET subcommand (1) (2) in the sample form definition establishes the corner or origin of the logical page relative to the physical sheet. The WIDTH and HEIGHT subcommands, (3) and (4), specify the dimensions of the logical page relative to the logical page origin.

Note: Be careful not to define a logical page larger than the physical sheet. PPFA does not check the size of the physical sheet.

“Positioning the First Line of Data” shows you two ways to position the first line of data on the page.

---

Positioning the First Line of Data

The previous section showed you how to define the size of a logical page. The next two examples show you how to position the first line of data inside the logical page, using the LINEONE subcommand. This subcommand position is relative to the logical page origin, as shown in Figure 19 on page 38. The two coordinates, (1) and (2), of the LINEONE parameter define the starting point for the first line of text.
This starting point works with the POSITION, MARGIN, and TOP subcommands (of the PRINTLINE command) to position lines of print on a page.

The defaults for LINEONE are:

\[ x = 0, \]
\[ y = 80\% \text{ of one line space from the top of the logical page}: \]
\[ \quad 80\% \text{ of } 1/6 \text{ inch if } lpi = 6, \]
\[ \quad 80\% \text{ of } 1/8 \text{ inch if } lpi = 8, \]
\[ \text{and so on.} \]

These defaults leave room for the character ascenders in the first line of text.

**Note:** PPFA subtracts one logical unit (L-unit) from the \( y \) value to compensate for the fact that the printer counts L-units beginning with the number 0. Therefore, if you specify the offsets to the first line in L-units (PELS is the measurement command for L-units) using the LINEONE subcommand, you must remember to subtract one L-unit from the \( y \) offset value. This is necessary to prevent descenders on the last printed line from dropping off the bottom of the logical page.

The following examples illustrate two methods for positioning the first line of text:

1. The position of the first line of data defaults by specifying the SETUNITS command prior to the PAGEDEF command, like this:

   ```plaintext
   SETUNITS 1 IN 1 IN
   LINESP 8 LPI;
   FORMDEF ABCD
   OFFSET 0 .5;
   PAGEDEF ABCD
   WIDTH 7.5
   HEIGHT 10
   DIRECTION ACROSS;
   FONT GS12 GS12;
   PRINTLINE REPEAT 60
   FONT GS12
   POSITION 0 TOP;
   ````

   **Note:** It is important that the LINESP subcommand (of the SETUNITS command) must precede the PAGEDEF commands.
If the LINESP subcommand follows the PAGEDEF command, PPFA will then use the default LINESP value to calculate the y offset value, which is used to position the first line of print.

The default for the LINESP subcommand of the SETUNITS command is 6 lpi. If LINEONE is allowed to default, based upon the LINESP default, the LINEONE value will be 31 L-units:

\[
\text{LINEONE} = \left( \frac{240 \text{ L-units}}{6 \text{ lpi}} \times 80\% \right) - 1 \text{ L-unit} = 31 \text{ L-units.}
\]

This value will be the vertical (y) position of the printline because TOP is specified in a later POSITION subcommand. However, this value may cause the data to exceed the bottom boundary of the logical page if the LINESP value is changed later.

2. Another way you can specify the starting position for the first print line is to specify LINEONE explicitly, like this:

```
FORMDEF ABCD
  OFFSET /zerodot .5;
PAGEDEF ABCD
  WIDTH 7.5
  HEIGHT 10
  LINEONE /zerodot PELS 23 PELS
  DIRECTION ACROSS;
SETUNITS 1 IN 1 IN
  LINESP 8 LPI;
FONT GS12 GS12;
PRINTLINE REPEAT 60
  FONT GS12
  POSITION /zerodot TOP;
```

In this example, the LINESP subcommand following the PAGEDEF command will not cause a data placement problem because the LINEONE command determines explicitly where the first line of text is positioned, and no default LINESP value is used:

\[
\text{LINEONE} = \left( \frac{240 \text{ L-units}}{8 \text{ lpi}} \times 80\% \right) - 1 \text{ L-unit} = 23 \text{ L-units}
\]

If you use the LINEONE command to specify an absolute starting position for the first line, in L-units, you must remember to subtract one L-unit from that value.

---

### Changing Logical Page Print Direction

Logical pages can have four different print directions: ACROSS, DOWN, BACK, and UP. This example shows that all four directions can be specified in relation to one offset specification:
FORMDEF ABCD
  OFFSET (1) (2) ;
PAGedef DEFG ;
  PAGEFORMAT DEFG1
    WIDTH (3)
    HEIGHT (4)
    DIRECTION ACROSS ;
  PRINTLINE ;
PAGedef DEFG2
    WIDTH (3)
    HEIGHT (4)
    DIRECTION DOWN ;
  PRINTLINE ;
PAGedef DEFG3
    WIDTH (3)
    HEIGHT (4)
    DIRECTION BACK ;
  PRINTLINE ;
PAGedef DEFG4
    WIDTH (3)
    HEIGHT (4)
    DIRECTION UP ;
  PRINTLINE ;

One page definition is used to simplify the example, yet four logical pages are specified. The PAGEFORMAT commands create subsets of page definitions for each logical page.

**Note:** The page formats in this example require an Invoke Data Map structured field at the place in the data file where you want to change page formats. The PRINTLINE commands are required but are not relevant in the example.

The DIRECTION subcommand with one of its four direction parameters, ACROSS, DOWN, UP, or BACK, specifies the print direction of the logical page.

Figure 20 on page 41 shows the format of each of the logical pages specified in the page definition with the direction specification of each. The pages with the ACROSS and BACK directions are in portrait presentation. The pages with the DOWN and UP directions are in landscape presentation.
Figure 20. Logical Page Print Directions in Relation to Origin

The media origins and logical page origins do not change with the presentation of the data on the page. The OFFSET subcommand of the form definition need not change. However, the width and height dimensions do change; that is, the WIDTH subcommand always governs the horizontal (inline) dimension as you view the page, and the HEIGHT subcommand always governs the vertical (baseline) dimension whether the page is in portrait or in landscape presentation. Ensure that these specifications do not cause the logical page to cross the edge of the physical page.

However, if the DOWN direction is specified for use with the IBM 3835 or 3900 Page Printer, the PRESENT and DIRECTION subcommands may need to be specified in the form definition. See “Specifying Page Presentation on Continuous-Forms Printers” on page 28 for more information.

Printing Line Data on a PSF Printer

This example shows how you can print a data file developed for a line printer on a page printer without altering the data. The example compares the effects of line printer controls with the corresponding controls in the PPFA commands and subcommands. PRINTLINE, LINESP, POSITION, CHANNEL, and REPEAT are page definition controls related to the lines of text in your printout. Line printer
controls examined are the forms control buffer (FCB) and carriage control characters.

As shown in Figure 21, a file consisting of 13 records is to be printed. Several different printouts of this data are formatted in the following examples. In the first two printouts, records 1–6 are printed on page 1, records 7–9 on page 2, and records 10–13 on page 3.

```
  Carriage-Control Character
    1 | RECORD 1
        | RECORD 2
        | RECORD 3
        | RECORD 4
        | RECORD 5
        | RECORD 6
        1  | RECORD 7
        | RECORD 8
        | RECORD 9
        1  | RECORD 10
        | RECORD 11
        | RECORD 12
        | RECORD 13

Data
```

*Figure 21. Line-Data File*

Figure 22 on page 43 shows the formatting process used when the file is printed on a line printer. For many line printers, an FCB is used to format the output in the S/370 (MVS, VM, VSE) environment. The sample FCB represented in Figure 22 on page 43 determines that no printed page contain more than eight lines. A page can have exactly eight lines without using carriage control characters in the data. A page may contain any number of lines fewer than eight; this is effected by placing fewer than eight records between the carriage control characters in the data. In the data file in Figure 21, fewer than eight records are, in all cases, placed between channel 1 carriage control characters. A ninth record, if encountered before a carriage control character, would cause a page eject and a return to the beginning of the FCB. The printout shown in Figure 22 on page 43 results from the data being formatted by this FCB.
A page definition can work exactly the same way. Consider the following example:

```
SETUNITS 1 IN 1 IN
   LINESP 6 LPI ;
PAGEDEF ABCD
   WIDTH 5
   HEIGHT 7
   LINEONE .5 .5 ;
PRINTLINE CHANNEL 1
   POSITION MARGIN TOP
   REPEAT 8 ;
```

This command stream contains one new command (PRINTLINE) and four new subcommands (LINESP, CHANNEL, POSITION, and REPEAT) related to controlling individual lines.

- The LINESP subcommand has the same function as the LPI specifications in the FCB or in a Printer File; it defines the line density in lines per inch.
- The PRINTLINE command contains the controls for one or more lines.
- The CHANNEL subcommand has the same function as the channel 1 control character in the FCB, causing a page eject at each channel 1 control character encountered in the data records.
- The POSITION subcommand establishes the location of the first line relative to the upper-left corner of the logical page. This example uses the MARGIN and TOP parameters; however, numeric parameters similar to those used with the
OFFSET subcommand can also be used. Those values are also relative to the logical page.

- The REPEAT subcommand is a commonly used control in PPFA text formatting. It is the way you specify the total number of PRINTLINEs in a logical page.

**Note:** The constraints in specifying a REPEAT value and, thereby, the number of lines per page are: the lines-per-inch specification, the height of the logical page, and the font selection. The REPEAT variable “8” is chosen to equal the maximum number of records to be printed per page. As in the line printer version, if a ninth record were encountered before a channel 1 carriage control character, a page eject would occur and the line would be printed as the first line at the top of the next page.

The result of this page definition is represented in Figure 23.

![Printout](image)

*Figure 23. Printout Examples Specifying POSITION MARGIN TOP*

Changing line printing specifications is shown in the next example:

```plaintext
SETUNITS 1 IN 1 IN
LINESP 6 LPI ;
PAGEDEF ABCD
   WIDTH 5
   HEIGHT 7
   LINEONE .1 .1 ;
   PRINTLINE CHANNEL 1
      POSITION MARGIN 4.1
      REPEAT 8 ;
```

Note that the second parameter of POSITION is no longer TOP; instead it is 4.1, which places the first line of text 4.1 inches down the page rather than at the top (Figure 24 on page 45).
The next example shows a third version of the possible formats for the data represented in Figure 22 on page 43.

```
SETUNITS 1 IN 1 IN
LINESP 6 LPI ;
PAGEDEF ABCD
  WIDTH 5
  HEIGHT 7
  LINEONE .1 .1 ;
PRINTLINE CHANNEL 1
  POSITION MARGIN TOP
  REPEAT 8 ;
PRINTLINE CHANNEL 1
  POSITION MARGIN 4.1
  REPEAT 8 ;
```

You also can skip over space using carriage control characters. This example shows how to do this by using a second PRINTLINE command to create a second starting position on the page (as shown in Figure 25 on page 46). The second starting position is vertically 4.1 inches down from the top of the page; see the second POSITION subcommand. The two CHANNEL 1 subcommands take turns mapping the records governed by the successive channel 1 carriage control characters in the data to their specified positions on the page. In this case, the carriage control 1 characters cause printing to alternate between the TOP position (0.1 inch down the page) and 4.1 inches down the page.
The OS/400 Environment

This example shows how you can print a data file developed for a line printer on a page printer without altering the data, within the OS/400 environment.

If, in DDS, the following example were used:

```
Page length=66, LPI=6, OVRFLW=60 (10 inches)
Print 36 lines at 6 LPI (6 inches)
Print 16 lines at 4 LPI (4 inches)
```

you could get the same formatting in PPFA by coding:

```
PAGedef MixLPI
   Width 8 in
   Height 10 in
   Lineone x y
   Setunits linesp 6 lpi;
   Printline position margin top
   Repeat 36;
   Setunits 4 lpi;
   Printline repeat 16;
```

Processing Fields

This section describes the mapping of individual fields to the printed sheets. The technique allows you to print unformatted data according to precise specifications, and these specifications can change without affecting the data file.

The rules for field processing of data files are:

- Each record in your file must correspond to a separate PRINTLINE command because each record is mapped separately. When processing identical fields, you can define a single printline and use the REPEAT subcommand.
- Each FIELD command must follow its associated PRINTLINE command, and more than one FIELD command can be specified for a single PRINTLINE command.
For this field-processing example, the data file shown in Figure 26 on page 47 is used. Figure 27 on page 47 represents an output format that could be used to place data on a form, such as an invoice or an order. The page definition commands to print Figure 27 on page 47 are as follows:

```
PAGEDEF ABCD
    WIDTH 7 IN
    HEIGHT 8 IN ;
    PRINTLINE POSITION 1 IN 1 IN ; /*PROCESSING FOR R1 */
    FIELD START 1 LENGTH 4 ; /*THE PRINTLINE POSITION IS */
    /*THE DEFAULT FOR THE FIRST FIELD*/
    FIELD START 11 LENGTH 4
    POSITION 4 IN 0 IN ;
    PRINTLINE POSITION 3 IN 4 IN ; /*PROCESSING FOR R2 */
    FIELD START 1 LENGTH 4 ; /*DEFAULT POSITION */
    FIELD START 6 LENGTH 4
    POSITION 0 IN 1 IN ;
    FIELD START 13 LENGTH 3
    POSITION 2 IN 3 IN ;
    PRINTLINE POSITION 1 IN 2 IN ; /*PROCESSING FOR R3 */
    FIELD START 1 LENGTH 4 ; /*DEFAULT POSITION */
    FIELD START 11 LENGTH 4
    POSITION 4 IN 0 IN ;
```

![Data File](image1)

*Figure 26. Unformatted Print Data File*

![Data Arranged on the Printed Page](image2)

*Figure 27. Data Arranged on the Printed Page*
POSITION Subcommand as Used in this Example

The POSITION subcommand of each PRINTLINE command specifies the printline position relative to the logical page origin. The POSITION subcommands below FIELD commands specify a field position relative to the governing printline position. Following POSITION subcommands come the horizontal (x) then the vertical (y) offsets from the reference point. They are parallel in structure to the OFFSET subcommand of the form definition.

For example, the final POSITION subcommand places the final field 1 + 4 inches to the right of the left edge of the logical page, combining the x value of 1 in the PRINTLINE command, and the x value of 4 in the nested FIELD command. The 0 in the FIELD command specifies no change to the y value in the PRINTLINE command. Thus, the position of the final field is 5 IN (x), 2 IN (y).

Note: The first FIELD command within each PRINTLINE has no position specification, because the PRINTLINE POSITION value is the default for the first FIELD command nested under it.

Alternate controls for the x and y values of a POSITION subcommand are available. See the description of the POSITION subcommand in “FIELD Command” on page 158 and “PRINTLINE Command” on page 178.

FIELD Command as Used in this Example

In the FIELD command, the START and LENGTH parameters specify the location of the field in the record to be processed. START indicates the starting byte position, and LENGTH specifies the number of bytes in the field.

Because a field can be located independently within the data and on the printed page, more than one page definition or page format can be created for the same data file, each specifying different mapping of the data to the output pages.

Color on the IBM InfoPrint HiLite Color Post Processor

This section provides an example of the use of Highlight color. Figure 29 on page 50 shows where the text is placed on the page. The CALIBRATION setup is as follows:

<table>
<thead>
<tr>
<th>Start pel</th>
<th>Color</th>
<th>Width in pels</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>high 1</td>
<td>288</td>
</tr>
<tr>
<td>488</td>
<td>high 2</td>
<td>288</td>
</tr>
<tr>
<td>776</td>
<td>high 3</td>
<td>192</td>
</tr>
</tbody>
</table>

The page definition commands to print Figure 29 on page 50 are as follows:
Figure 28. Unformatted Print Data File
Figure 29. Data Arranged on the Printed Page with Color

The data ‘AAA’ and ‘1111’ are printed in highlight color 1. The data ‘XXXX’ and ‘YYYY’ are printed in highlight color 2. The data ‘BBBB’, ‘ZZZZ’, and ‘2222’ are printed in highlight color 3.

Setup Verification

You can use the VFYSETUP subcommand to put the name of the printer color setup into the form definition. At print time, PSF compares the setup name in the form definition to verify that the setup was activated in the printer. See the VFYSETUP subcommand in “FORMDEF Command” on page 135 for more information.

Varying Fonts on a Page

This example illustrates a simple font variation within a printout. The task is to print a line-data file having the first line of each page in bold-faced type and the rest in standard type. This requires controls for two fonts in the page definition.

The commands to select a single font for the page, as shown in Figure 30 on page 51, are as follows:

```
PAGEDEF ABCD;
  FONT STANDARD M1/zerodot1;
  PRINTLINE;
```

Note: Fonts cannot be an FGID. Also, all page definitions require a PRINTLINE command.
Figure 30. Data File Printed Using a Single Font

The next command stream changes the font by incorporating a TRCREF command. Assume the data file to be formatted incorporates table reference characters (TRCs) as shown in Figure 31 on page 52.

```
PAGEDEF ABCD ;
   FONT STANDARD M1/0.1 ; //creating local font names /*
   FONT BOLDFACE M1/0.2 ;
   PAGEFORMAT ABCD ;
   TRCREF 0 ; //Defining the TRC values */
      FONT STANDARD ;
   TRCREF 1 ;
      FONT BOLDFACE ;
PRINTLINE CHANNEL 1
   POSITION 1 IN 1 IN
   REPEAT 8 ;
```
The TRCs in the data cause the font switch to be made. The TRCREF command equates a TRC in the data file with the local name of a font specified in the FONT command. The FONT command also contains the user-access name for the font. See Table 6 on page 121 for information on local names and user-access names. Because of the relationship among the user-access name, the local name, and the TRC number that is established in the page definition, the TRCs in the data can cause a font switch automatically.

You can specify fonts within a PRINTLINE command when the data file contains no TRCs. For example:

```
PAGEDEF ABCD;
  FONT M101;
  FONT BOLDFACE M102;
  PRINTLINE CHANNEL 1 /*BOLDFACE LINE */
    POSITION MARGIN TOP
    FONT BOLDFACE;
  PRINTLINE POSITION MARGIN NEXT /*STANDARD-TYPE LINE */
    FONT M101
  REPEAT 7;
```

assume the data file represented in the sample print in Figure 32 on page 53 is to be formatted by this page definition.

This command stream, based on a data file without TRCs, works on the principle that each line of output whose font you want to change from the font in the previous line must be controlled by a separate PRINTLINE command. The FONT
subcommand of the PRINTLINE command names the font desired for that line. In this example, two PRINTLINE commands are used because one font change and two fonts are intended for the output. The user-access font names appear in the two FONT commands immediately below the PAGEDEF command and, optionally, a local name. M101 and M102 in the example are user-access names; BOLDFACE is a local name. Use the local name in the FONT subcommand of PRINTLINE if it is included in the corresponding FONT command, as is done for the first PRINTLINE command.

Two possible defaults apply in case you do not specify a font within a field. If the governing printline has a FONT subcommand, it will contain the font default for the field. If the governing printline has no font specification, PSF will assign a font according to its default rules.

Figure 32. Font Change Using FONT Commands and Subcommands

Changing fonts field by field is similar to changing them in printlines. You map each field individually with a FIELD command; include a FONT subcommand in the FIELD command. If a font change is desired for a field, as with the FONT subcommand of a PRINTLINE command, the font must be previously named in a FONT command.

Two possible defaults apply in case you do not specify a font within a field. If the governing printline has a FONT subcommand, it will contain the font default for the field. If the governing printline has no font specification, PSF will assign a font according to its default rules.
Printing Lines in Two Directions on a Page

Lines can be printed in any of four directions, depending on the type of printer being used. Refer to Advanced Function Presentation: Printer Information for the print directions supported by your printer.

The four parameters for line direction are ACROSS, DOWN, BACK, and UP. The PPFA commands used to format a line-data file with lines printed in more than one direction (as shown in Figure 33) are stated in the following page definition:

```
PAGEDEF ATOG
  DIRECTION ACROSS ;
  PRINTLINE POSITION 1 IN 1 IN /*LINES A-E */
  REPEAT 5 ;
  PRINTLINE POSITION .5 IN 6 IN /*LINE F */
  DIRECTION UP ;
  PRINTLINE POSITION 1 IN 6 IN /*LINE G */
```

![Data File](image)

Figure 33. A Printout with More Than One Line Direction

In this page definition, the logical page direction ACROSS is specified. This is actually the default, but its inclusion clarifies that no direction control is needed for lines A–E. The default direction of a printline is the direction specification of the logical page of which it is part. The PRINTLINE command for the record F has a DIRECTION subcommand because the direction specification changes from that of the previous line. Record G is to be printed in the ACROSS direction again. A direction is not specified, however, because the ACROSS direction is the default for all lines in this page definition.

**Note:** If you are building the page definition for use with the 3800 printer, and if the input data contains table reference characters, you can use the DIRECTION subcommand of the TRCREF command to specify a font that will print UP on the page, as in line F. For more information, see “TRCREF Command” on page 188.

Printing Fields in Two Directions on the Same Page

This example is similar to Printing Lines in Two Directions on a Page, except that you learn how to control direction field by field. This method creates a field-processing page definition and places direction controls in the FIELD commands. This command stream contains a portion of the page definition controls, showing only the PRINTLINE commands:
As expected in field processing, FIELD commands are nested within PRINTLINE commands. Figure 34 shows a simplified portion of an unformatted file and two pages of the printout formatted by the page definition, part of which is shown in the command stream. Two printlines are specified because, as Figure 34 shows, the data file contains two input record formats (1 and 3 are alike; 2 and 4 are alike) and because the fields are mapped to two different positions in the output. The assumption of this sample is that the data file is actually much longer than the portion shown. If, however, the records in the file alternate in format as the first four do, the two PRINTLINEs of this page definition will format as many records as are presented, two to a page, on pages 1 through n.

If more than two mappings are required by the print job, more than two PRINTLINE commands are required in the page definition.

![Data File](image.png)

Figure 34. Field Direction

### Rotating Fonts

Fonts rotate relative to the inline direction of lines (or fields).

This example focuses on a single letter A from FONTA. With PPFA, a single font specified in a page definition can produce letters in any of four rotations. This is accomplished by a FONT command that specifies rotation. If, as in this example, you want to vary the rotation of a font twice within a page, you use two FONT commands, one for each rotation. You also use two PRINTLINE commands to map the data to the printout, using the two rotations of the font. In a field processing application, FIELD commands can be used in the same way. These PRINTLINE commands name the rotated font in a FONT subcommand.

Figure 35 breaks down the elements required for the FONT commands and subcommands. Distinct local names and rotation specifications for each font are placed in a FONT command. These identify a font as rotated within a page definition. The rotation of a character is relative to the inline direction of a printline or field. The characters and rotations shown here assume an inline direction of ACROSS. See “PPFA Basic Terms” on page 8.
You can use up to 16 possible combinations of logical page direction and font rotation for page printers other than the 3800.

The FONT subcommands within PRINTLINE or FIELD commands that name the rotated font in that page definition use only the local name. The following command stream shows the proper specification and nesting of FONT commands and subcommands for rotation.

```
PAGEDEF ABCD ;
  FONT FONTA M103 ; /*NO ROTATION, LOCAL AND USER-ACCESS NAMES.*/
  FONT FONTALD180 M103 /*ROTTED FONT, LOCAL, USER-ACCESS NAMES.*/
    rotation 180 ; /*NAMES PLUS ROTATION SUBCOMMAND AND参数.*/
  PRINTLINE FONT FONTA /*LOCAL NAME*/
    REPEAT 3 ;
  PRINTLINE FONT FONTALD180 /*LOCAL NAME*/
    REPEAT 2 ;
```

Figure 35. Character Rotation

Figure 36. Example of Assumed Data File and Rotation Specifications
FONTA, identified in the first FONT command, requires no rotation parameter because it is printed in the default position (or 0° rotation) for font M103. For the rotated font, the second FONT command identifies FONTARTD180 (the local name) as M103 rotated 180°.

Using Traditional Kanji Formatting

Traditional kanji print presentation, called *tate*, is possible with PSF printers, using a combination of font rotation and logical page direction. A logical page in the DOWN direction and a 270° font rotation provide the right combination to present kanji in *tate* format on an PSF printer.

```
FORMDEF TATE
  OFFSET 1 IN 1 IN ;
PAGEDEF TATE
  HEIGHT 5 IN
  WIDTH 6 IN
  DIRECTION DOWN ;
  FONT KANJIRTD M104
  ROTATION 270 ;
  PRINTLINE FONT KANJIRTD
  REPEAT 3 ;
```

Figure 37 shows the result of formatting with the above page definition. The characters are added to lines down the page. Lines are added right to left.

![Figure 37. 3820 Tate Presentation](image)

Printing Multiple-Up Pages

*Multiple up* is a printer’s term for printing two or more pages of data on one side of a sheet, which is possible with PSF printers and PPFA formatting. The steps used in this example are:

1. Change the print direction of the logical page to one of the landscape presentations.
2. Conceptually divide the sheet of paper into parts, one for each multiple-up page (subpage).
3. Create a printline position at the top of each multiple-up page.

This example assumes the existence of a line-data file with carriage control 1 characters after records 4, 7, and 11. Each carriage control 1 character begins a
new page. Because there are really four pages on the sheet, a skip-to-channel 1 must be used four times. The fifth channel 1 character causes a page eject and the beginning of a new physical sheet. The PPFA commands that follow are for one version of a multiple-up page. This set of commands creates a page layout like the one shown in Figure 38 (the physical sheet is not shown).

```
FORMDEF MULTUP
  OFFSET 1 IN .5 IN ;
  SETUNITS LINESP 4 LPI ;
PAGedef MULTUP1
  WIDTH 10 IN
  HEIGHT 8 IN
  DIRECTION DOWN /*FOR LANDSCAPE PRESENTATION */
  PRINTLINE CHANNEL 1 /*PAGE 1 */
    POSITION 1 IN 1.5 IN
    REPEAT 6 ;
  ENDSUBPAGE ;
  PRINTLINE CHANNEL 1 /*PAGE 2 */
    POSITION 1 IN 5.5 IN
    REPEAT 6 ;
  ENDSUBPAGE ;
  PRINTLINE CHANNEL 1 /*PAGE 3 */
    POSITION 6 IN 1.5 IN
    REPEAT 6 ;
  ENDSUBPAGE ;
  PRINTLINE CHANNEL 1 /*PAGE 4 */
    POSITION 6 IN 5.5 IN
    REPEAT 6 ;
```

```
Page 1
Record 1
Record 2
Record 3
Record 4

Page 3
Record 8
Record 9
Record 10
Record 11

Page 2
Record 5
Record 6
Record 7

Page 4
Record 12
Record 13
Record 14
```

*Figure 38. Multiple-Up Page Layout*

The **DOWN** printline direction creates a page with a landscape presentation typical of multiple-up printing. Individual printlines are specified for the initial lines of the four pages. Ensure that the lines of each page fit in the space designated by the use of a small font.
Note: In this example, no font is specified for the page definition; therefore, the default font for the page printer is used. If you want a different font, write a FONT command naming it.

The next set of commands alters the sequence of pages.

FORMDEF MULTUP
OFFSET 1 IN .5 IN ;
SETUNITS LINESP 4 LPI ;
PAGEDEF MULTUP2
WIDTH 10 IN
HEIGHT 8 IN
DIRECTION DOWN ;
PRINTLINE CHANNEL 1 /* PAGE 1 */
POSITION 1 IN 1.5 IN
REPEAT 4 ;
ENDSUBPAGE ;
PRINTLINE CHANNEL 1 /* PAGE 2 */
POSITION 6 IN 1.5 IN
REPEAT 4 ;
ENDSUBPAGE ;
PRINTLINE CHANNEL 1 /* PAGE 3 */
POSITION 1 IN 5.5 IN
REPEAT 4 ;
ENDSUBPAGE ;
PRINTLINE CHANNEL 1 /* PAGE 4 */
POSITION 6 IN 5.5 IN
REPEAT 4 ;

Here, the upper-right and lower-left pages have been reversed by reversing the position controls for the second and third printlines.

Figure 39 shows the changed printout resulting from the page definition command changes. Once you have set up your basic page definition, changes such as this become easy.
Note: The ENDSUBPAGE command can be used to mark the boundaries between subpages. Without it, the page definition is no different from any other sequence of PRINTLINEs with POSITION commands. Boundaries do not have to be marked unless conditional processing is being performed. The examples given here will print identically with and without ENDSUBPAGE commands. (See “Subpage Description and Processing” on page 77 for more information.)
Chapter 4. Creating Complex Printouts

You are now ready to learn about some formatting tasks that might apply to more complex printouts. The basic form definition and page definition elements have been covered. This chapter describes how these elements are combined to create complete print jobs.

The advanced techniques covered in this section are illustrated in the following examples:

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Example location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Processing with Overlay</td>
<td>“Combining Field Processing and an Electronic Overlay” on page 61</td>
</tr>
<tr>
<td>Suppressing Data</td>
<td>“Using Suppressions to Vary Data Presentation” on page 63</td>
</tr>
<tr>
<td>Including Fixed Text</td>
<td>“Incorporating Fixed Text into a Page Definition” on page 65</td>
</tr>
<tr>
<td>Combining Two Reports</td>
<td>“Combining Two Reports into One Printout” on page 68</td>
</tr>
</tbody>
</table>

The examples in this chapter build on a single sales application, showing different sales reports being formatted by form definitions and page definitions.

Combining Field Processing and an Electronic Overlay

This example involves printing a monthly individual sales report for a specified distribution. The following items are needed to generate the sales report:

- A pre-designed electronic overlay for the sales report
- An unformatted print data file with periodic sales statistics

An example of these is shown in Figure 40 on page 62.
The code example that follows contains a form definition and a page definition. The page definition maps the file to the overlay.

In Figure 40 the 0,0 point is the upper-left corner of the overlay. This means that the logical page origin must coincide with the overlay origin in this example. POSITION subcommands are relative to the logical page origin. The overlay origin point that positions the overlay is specified in the Overlay Generation Language/370 that creates the overlay, not in the page definition. In mapping to an overlay, you should check the input to the overlay creation program so you can coordinate its origin with the logical page origin.
A time-saving device used in the above example is the REPEAT subcommand (line 16), which maps a single printline with its field subsets to records 4 through 7 with all model names and sales statistics. The length values in the repeated fields are 7, 3, 3, and 3—sufficient to accommodate the largest model name, unit value, $(000), and percentage fields mapped by this FIELD command.

Figure 41 shows the report formatted by the resources generated in the command stream of this example.

![Sales Report](image)

**Figure 41. Sales Report**

### Using Suppressions to Vary Data Presentation

PPFA and the PSF printers enable you to produce variations of the same report in a single job. The essential function for this capability is called *suppression*. Suppression involves the coordinated specification of elements in both the page definition and the form definition. You create a suppression in the page definition and turn it on or off in a subgroup within a form definition.
The purpose of the example is to demonstrate how to alter the controls in the previous example ( "Combining Field Processing and an Electronic Overlay" on page 61) in order to generate a second report along with the one already created.

First, change the page definition by adding a SUPPRESSION subcommand to the third field in the repeated printline—the printline that mapped the models and sales figures in “Combining Field Processing and an Electronic Overlay” on page 61. The suppression is, in effect, created by the SUPPRESSION subcommand in the FIELD command. The following example shows the addition at line 23.

```
18 FIELD START 1 LENGTH 7 ;
19 FIELD START 10 LENGTH 3
20 FIELD START 16 LENGTH 3
22 POSITION 2.5 IN * ;
23 SUPPRESSION SALES ; //ADDED LINE */
24 FIELD START 21 LENGTH 3
25 POSITION 3.5 IN * ;
```

The SUPPRESSION subcommand creates the potential for selective suppression of the data in the “$(000)” field of the report.

Then, rewrite the form definition, creating two subgroups within the copy group. Next, write a SUPPRESSION command immediately after the FORMDEF command. Finally, place a SUPPRESSION subcommand in the subgroup in which you want the data suppressed. This names the suppression. The resulting form definition command stream is as follows:

```
FORMDEF SECRPT ;
    SUPPRESSION SALES ; /*NAMING THE SUPPRESSION */
    COPYGROUP SECRPT ;
        OVERLAY SLSRPT ; /*NAMING THE OVERLAY */
        SUBGROUP COPIES 1
            OVERLAY SLSRPT ;
        SUBGROUP COPIES 1
            OVERLAY SLSRPT
                SUPPRESSION SALES ; /*TURNING ON THE SUPPRESSION */
```

The result is shown in Figure 42. The second subgroup creates the second output page of the same data with a second set of modifications; in this case, modifications means a suppression that is not in the first subgroup.

![Figure 42. Selective Suppression](image-url)
Review the steps in this example. To suppress a field, identify the field as suppressible in the page definition under the FIELD command in question. Then create a subgroup, activating this suppression with a SUPPRESSION subcommand in the form definition.

The first subgroup produces an output identical to the report in “Combining Field Processing and an Electronic Overlay” on page 61. It contains no suppression.

**Note:** This example can only be printed simplex.

---

**Incorporating Fixed Text into a Page Definition**

Fixed text can be incorporated into an electronic overlay through the use of programs, such as Overlay Generation Language/370. Having another place (the page definition) to incorporate fixed text permits you to format documents more efficiently.

In “Combining Field Processing and an Electronic Overlay” on page 61, a territory sales report for salesman John Smith is created. Here, the territory sales report is incorporated into a larger format going to ACME’s corporate headquarters in Chicago. Therefore, the identification for the region needs to appear on the report form. An overlay is used as a header for the composite report. This means that two overlays appear in the command stream: one carries over from “Combining Field Processing and an Electronic Overlay” on page 61 and the other is the header.

So, as shown in Figure 43 on page 66, three fixed inputs generate the final report: overlay SLSRPT, overlay HDR, and the fixed regional identification text. (It is the second item that is worked into the page definition in this example.)
The data file used to generate this report is the same as the one shown in Figure 40 on page 62.
In the above command stream, the same basic commands from “Combining Field Processing and an Electronic Overlay” on page 61 are used, although the positions of fields have been changed to accommodate the new layout.

New FIELD commands with TEXT subcommands have been inserted in the first PRINTLINE command to produce the regional text, which is positioned at the bottom of the header form. The 1 is a duplication parameter indicating how many times the fixed text is to be repeated. The C can precede single-byte characters such as those used, for example, to write English or German. Both 1 and C are the default values for a TEXT subcommand. The text you want inserted appears between single quotation marks. Note how the POSITION subcommands change to accommodate both fixed text and record-1 text.

**Note:** Each PRINTLINE command in your PPFA command stream should have a corresponding record in the input data file. If you specify a fixed-text data field and an input data field under the same PRINTLINE command, they will both be associated with the same input data file record. However, if all the FIELD commands under a PRINTLINE command specify fixed text, the corresponding input record will simply be discarded. In that case, you should insert a blank record into the input data file to preserve the correct relationship between records and PRINTLINE commands.

Figure 44 on page 68 shows how the finished output looks.
Combining Two Reports into One Printout

This example combines two data files and two page layouts into one printout, also building on “Combining Field Processing and an Electronic Overlay” on page 61.

Figure 45 on page 69 shows the new data and a new overlay.
Here is the command stream needed to generate both pages of the preceding report:

```
Invoke Medium Map Control Record
Invoke Data Map Control Record
```

**Data File**

Figure 45. Input for a New Report Produced from the Combined Data Files
Although requiring a complex series of commands, the following commission report is handled much like any other field processing problem: the data must be carefully mapped into the overlay exactly where it is wanted. If, as in this example, you change copy groups and page formats, both the Invoke Medium Map structured field and the Invoke Data Map structured field must be inserted into the data file where the changes are desired. Here they occur together.

Figure 46 on page 71 shows both the commission report and the sales report. With page printers and with careful data positioning, such reports look like they were individually prepared with no differences in the presentation of the fixed data.
<table>
<thead>
<tr>
<th>MODEL</th>
<th>UNITS</th>
<th>(000)</th>
<th>% of TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sierra</td>
<td>12</td>
<td>59</td>
<td>6</td>
</tr>
<tr>
<td>Otero</td>
<td>16</td>
<td>70</td>
<td>10</td>
</tr>
<tr>
<td>Agua</td>
<td>60</td>
<td>104</td>
<td>15</td>
</tr>
<tr>
<td>Allegre</td>
<td>71</td>
<td>265</td>
<td>40</td>
</tr>
</tbody>
</table>

Salesman's Commission Report
Acme Corp.
To: John Smith
As your sales for Nov. 1995 were $498,000.00, and your commission rate is 2%, your monthly commission is $9960.00. Should there be any question, please contact Al Jankowski in accounting.
Sincerely,
Acme Corp.

Figure 46. The Sales and the Commission Reports
Chapter 5. Conditional Processing

Conditional processing allows you to test fields within an input line data record (for example, a customer number). Based on the results of the test, you can specify the action to be taken such as to change copy groups or page formats. This section provides:

- An explanation of how conditional processing works
- A detailed list of rules, restrictions, and considerations
- Examples showing how conditional processing can be used to perform some commonly-requested functions

General Description

Conditional processing allows you to:

- Test the input data using the CONDITION command.
- Choose the copy group and page format to be used when printing the data.
- Change to a different copy group or page format after the data has been read. You can specify that the new copy group or page format is to be used:
  - Before printing the current subpage
  - Before printing the current line
  - After printing the current line
  - After printing the current subpage

Table 5 shows the tasks you may perform with conditional processing.

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Location of the Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stack offset from previous jobs</td>
<td>“Jog Output Example” on page 86</td>
</tr>
<tr>
<td>Use different print directions for front and back sides of a sheet</td>
<td>“Duplex Output with Different Front and Back Print Directions” on page 87</td>
</tr>
<tr>
<td>Record reprocessing example</td>
<td>“Record Reprocessing Example” on page 88</td>
</tr>
<tr>
<td>Select different paper sources</td>
<td>“Selecting Paper from an Alternate Bin Example” on page 89</td>
</tr>
<tr>
<td>Multiple CONDITION commands</td>
<td>“Multiple CONDITION Commands” on page 89</td>
</tr>
<tr>
<td>Repeat PRINTLINE commands</td>
<td>“Field Processing When PRINTLINEs Are Repeated” on page 93</td>
</tr>
</tbody>
</table>

Using Conditional Processing versus Normal Line Data Processing

Normal line-data processing consists of:

- Setting up the physical page environment by defining a copy group
- Setting up the logical page environment by defining a page format

Input records correspond to PRINTLINE commands that determine such things as where the input records are to be printed, which font to use and what print direction
to use. Only one copy group and page format can be used for processing each input record.

Conditional processing acts as a preprocessor by allowing you to test the input data before deciding which copy group and page format to use. Furthermore, you can change these specifications based on changes in the input data. Except for record reprocessing (explained on page 78), once the copy group and page-format specifications have been made, conditional processing operates the same as normal line-data processing.

**Note:** The copy group and page format can also be changed by placing Advanced Function Presentation data stream (AFP data stream) Invoke Medium Map (IMM) and Invoke Data Map (IDM) structured fields in the input data. Use of these structured fields within the input print file causes results that differ from what is described in this section. Refer to *Mixed Object Document Content Architecture Reference* for information about these structured fields.

---

**Using Conditional Processing to Set Up the Environment**

Setting up the environment consists of selecting a copy group and a page format.

**Selecting a Copy Group**

Conditional processing can be used to select a copy group; it does not process the copy group.

As described in Chapter 2, “Using Form Definition Commands” on page 17, a form definition contains the controls that govern the physical page on which the print file is to be printed. A form definition can contain one or more copy groups as shown in the following diagram.

<table>
<thead>
<tr>
<th>PPFA Commands</th>
<th>Resulting Form Definition</th>
</tr>
</thead>
</table>
| **FORMDEF FDEFX**<br>...<br>COPYGROUP CGA<br>...<br>OVERLAY ...
SUBGROUP ...
...<br>COPYGROUP CGB<br>...<br>OVERLAY ...
SUBGROUP ...
...<br>COPYGROUP CGC<br>...<br>OVERLAY ...
SUBGROUP ... | **F1FDEFX**
|               | **CGA**                   |
|               | **CGB**                   |
|               | **CGC**                   |

The first copy group within a form definition is always active when processing of a print file begins. To select a different copy group, use the CONDITION command.
Note: By using the BEFORE SUBPAGE and BEFORE LINE parameters with conditional processing, you can change to a different active copy group before any lines have actually been formatted.

Using the previous diagram as a reference, assume copy group CGB is active. The copy-group selections that can be made from a CONDITION command are:

- **condname** which starts the named copy group
- **CURRENT** which restarts copy group CGB
- **=** which restarts copy group CGB (alternate for CURRENT)
- **NEXT** which starts copy group CGC
- **FIRST** which starts copy group CGA
- **NULL** which does not make any change to the current copy group processing
- **/** which does not make any change to the current copy group processing (alternate for NULL)

See “Using the CONDITION Command to Select a Copy Group and a Page Format” on page 84 for more information on each of these options.

### Selecting a Page Format

Conditional processing can be used to select an active page format. Selecting the page format does not change the basic rules for processing a page format:

- PRINTLINE commands are processed sequentially unless skip-to-channel or spacing commands are used.
- When the end of the page format is reached, processing returns to the first PRINTLINE command in the same page format. Processing does not continue with the next page format (if any) in the page definition.

However, conditional processing does involve some additional considerations:

- **Subpages**
  A page format consists of one or more subpages. A subpage is defined by a group of PRINTLINE commands followed by an ENDSUBPAGE command. If an ENDSUBPAGE command is not defined, then the entire page format is one subpage. See “Subpage Description and Processing” on page 77 for more information.

- **Record reprocessing**
  Record reprocessing is used when input records are processed according to one set of copy-group and page-format specifications, and then new specifications are invoked for the same input records. See “Record Reprocessing Description and Processing” on page 78 for more information.

As described in Chapter 3, “Using Page Definition Commands” on page 35, a page definition is a set of controls for formatting line-data and unformatted ASCII files (typically AIX) for printing on a logical page. A page definition can contain one or more page formats as shown in the following diagram.
The first page format in the page definition is always active when processing of the print file begins. To invoke a new page format, use the CONDITION command.

**Note:** By using the BEFORE SUBPAGE and BEFORE LINE parameters, it is possible to change to a different active page format before any lines have actually been formatted.

Using the previous diagram as a reference, assume page format PFMTB is active. The page-format selections that can be made from a CONDITION command are:

- `condname` which starts the named page format
- `CURRENT` which re-starts page format PFMTB
- `=` which re-starts page format PFMTB (alternate for `CURRENT`)
- `NEXT` which starts page format PFMTC
- `FIRST` which starts page format PFMTA
- `NULL` which does *not* make any change to the current page format processing
- `/` which does *not* make any change to the current page format processing (alternate for `NULL`)

See “Using the CONDITION Command to Select a Copy Group and a Page Format” on page 84 for more information on each of these options.
Subpage Description and Processing

A page format consists of one or more subpages. A subpage is defined by a group of PRINTLINE commands followed by an ENDSUBPAGE command. If an ENDSUBPAGE command is not defined, then the entire page format is one subpage. The following considerations apply to subpages:

- Subpages are necessary only with conditional processing.

Multiple-up printing can be done with or without subpages being defined, but to change the page format or copy group at the level of one of the multiple-up pages, the multiple-up pages must be defined as subpages. In the following diagram, pages 1 through 4 can be defined as four separate subpages within one page format, or all defined within one subpage. However, in order to present the data on page 3 (for example) in a format different from that used for pages 1 and 2, the four pages must be defined as subpages.

- A subpage is processed sequentially starting from the beginning of the page format. Moving from one subpage to the next subpage is done by processing all the PRINTLINE commands for a given subpage, or by skipping (by means of the CHANNEL subcommand) or spacing to a PRINTLINE command in a different subpage.

Note: Conditional processing cannot be used to select a subpage except by default. When a page format is started (or the current one is restarted), processing begins with the first PRINTLINE command of the page format. The effect is to select the first subpage in the page format.
Record Reprocessing Description and Processing

Record reprocessing is used when input records are processed according to one set of copy group and page format specifications, and then new specifications are invoked for the same input records. If the new specifications are to be applied using either the BEFORE SUBPAGE or the BEFORE LINE parameter, then the input records must be processed again using the new specifications instead of the original ones.

**Note:** Input records are not printed twice; record reprocessing just changes the specifications used when formatting the records.

The process is shown in the following diagram.

<table>
<thead>
<tr>
<th>PPFA Commands</th>
<th>Input Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAGEFORMAT PFMTA ;</td>
<td></td>
</tr>
<tr>
<td>PRINTLINE POSITION 1 IN 1 IN</td>
<td></td>
</tr>
<tr>
<td>DIRECTION ACROSS</td>
<td></td>
</tr>
<tr>
<td>REPEAT 5 ;</td>
<td></td>
</tr>
<tr>
<td>CONDITION cond1</td>
<td>A</td>
</tr>
<tr>
<td>START 2 LENGTH 1</td>
<td>A</td>
</tr>
<tr>
<td>WHEN EQ &quot;B&quot;</td>
<td>B</td>
</tr>
<tr>
<td>BEFORE SUBPAGE</td>
<td></td>
</tr>
<tr>
<td>NULL PAGEFORMAT PFMTB ;</td>
<td>A</td>
</tr>
<tr>
<td>PAGEFORMAT PFMTB ;</td>
<td></td>
</tr>
<tr>
<td>PRINTLINE POSITION 7 IN 1 IN</td>
<td>A</td>
</tr>
<tr>
<td>DIRECTION DOWN</td>
<td></td>
</tr>
<tr>
<td>REPEAT 5 ;</td>
<td></td>
</tr>
<tr>
<td>CONDITION cond2</td>
<td></td>
</tr>
<tr>
<td>START 4 LENGTH 1</td>
<td></td>
</tr>
<tr>
<td>WHEN EQ &quot;Y&quot;</td>
<td></td>
</tr>
<tr>
<td>BEFORE SUBPAGE</td>
<td></td>
</tr>
<tr>
<td>NULL PAGEFORMAT PFMTA ;</td>
<td></td>
</tr>
</tbody>
</table>

Assume page format PFMTA is active. Under normal processing the first input record would print in the ACROSS direction, starting at a horizontal offset of 1 inch and a vertical offset of 1 inch. However, the third record satisfies the CONDITION statement and causes a new page format (PFMTB) to be started. Since CONDITION cond1 specifies BEFORE SUBPAGE, the first two records must be reprocessed using page format PFMTB. As a result, all of the records will be printed in a DOWN direction, starting at a horizontal offset of 7 inches and a vertical offset of 1 inch.

If allowed to operate without restrictions, record reprocessing could force PSF into an infinite loop. For example:
As in the previous example, page format PFMTA is initially active, and input record 3 results in the selection of page format PFMTB. However, page format PFMTB has a condition that checks position four for the character 'Y', which is satisfied by input record 5. Therefore, if there were no restrictions, page format PFMTA would again be selected, the input data would be reprocessed (starting with input record 1), leading to an infinite loop.

To prevent this situation, after a BEFORE condition has been satisfied, all other BEFORE conditions are ignored until data has actually been formatted. See “Record Reprocessing” on page 80 for detailed information on this restriction.

### Conditional Processing Rules, Restrictions, and Considerations

#### Multiple Conditions

Conditional processing supports:
- Multiple PRINTLINE commands in each subpage
- Multiple CONDITION commands on one PRINTLINE command
- Multiple WHEN statements on one CONDITION command

#### Rule

For all these situations, the rule is the same; the first true condition is the one processed, and any following true conditions are ignored.
Considerations
Conditions are evaluated when they are encountered. For example, if a true condition has not been detected when an OTHERWISE statement is encountered, the OTHERWISE statement always results in a true condition. (An exception to this is explained in “Interaction Between the CONDITION Command and the CHANNEL Subcommand” on page 81.)

See “Multiple CONDITION Commands” on page 89 for an example of multiple CONDITION commands.

Record Reprocessing

Restrictions
To prevent an infinite program loop, be aware that the following restrictions apply:

1. When the conditional action is to take place before the current subpage:
   a. Actions specified as taking place before the current subpage are shut off until the current subpage end.
   b. Actions specified as taking place before the current line are shut off for one line (the first line processed in the subpage).

2. When the conditional action is to take place before the current line, actions specified as taking place before the current subpage or before the current line are shut off for one line.

Considerations
- If a before subpage condition is true and causes a switch to a new page format, all before subpage conditions in the new page format are ignored.
- If a before line condition is true and causes a switch to a new page format, all before subpage and before line conditions in the new page format are ignored until one line has been processed.

The consequence of this is that, after a true condition, at least one line must be processed before the next before condition will be considered. This can be confusing because a condition that would otherwise yield a true result can be ignored.

See “Record Reprocessing Example” on page 88 for an example of record reprocessing.

Interaction Between a CONDITION Command and a REPEAT Subcommand

See “Interaction Between the CONDITION Command and the CHANNEL Subcommand” on page 81 for what can appear to be an exception to the following rules.
Rule for a CONDITION Command and a REPEAT Subcommand

The REPEAT subcommand is used with the PRINTLINE command to specify the number of printlines (usually greater than one) that are to be constructed with the same specifications (font, direction, and so on). The CONDITION command is used to invoke conditional processing based on the data in a particular line. When the REPEAT and CONDITION commands are both specified for the same PRINTLINE command, every line described by the PRINTLINE command is checked for the given condition until either the condition is satisfied or there are no more lines described by the PRINTLINE command.

Note: This is different from the way in which the CHANNEL and POSITION subcommands interact with the PRINTLINE command. These two subcommands apply only to the first line described by the PRINTLINE command.

Rule for a CONDITION Command With an OTHERWISE Subcommand

The REPEAT subcommand is used with the PRINTLINE command to specify the number of printlines (usually greater than one) that are to be constructed with the same specifications (font, direction, and so on). The CONDITION command is used to invoke conditional processing based on the data in a particular line. The CONDITION command includes one or more WHEN subcommands and may include an OTHERWISE subcommand. If an OTHERWISE is coded, and none of the preceding WHEN conditions are true, the OTHERWISE condition is always true. If an OTHERWISE command is not coded, it is treated as a null.

Considerations

For the situation where REPEAT and CONDITION with OTHERWISE are coded for the same PRINTLINE command, the first input line determines the processing to be performed. This happens because either one of the WHEN conditions or the OTHERWISE condition is always true for the very first line.

Interaction Between the CONDITION Command and the CHANNEL Subcommand

Rule

A condition is checked if its associated PRINTLINE command is actually processed.

Note: ANSI carriage controls and machine (EBCDIC) carriage controls are processed differently. See the SPACE_THEN_PRINT subcommand on page “Subcommands” on page 153 for more information.

ANSI  A skip or space occurs before printing the line.
Machine  The line is printed and then skipping or spacing is done.

For a CONDITION to be checked, it must be associated with the PRINTLINE command that is actually used for printing.

ANSI Skipping Consideration

The PRINTLINE command is not processed if a skip-to-channel-n character in the carriage control field causes the given PRINTLINE command not to be processed.

If a data record contains a character ‘1’ (for example) in the carriage control field, and a PRINTLINE command has been specified with CHANNEL 1 subcommand, the data record is processed under the “new” PRINTLINE command (the one that
specified CHANNEL 1). Any CONDITION associated with the “old” PRINTLINE
command is ignored (never even checked). See the following diagram for an
example of this.

The character ‘1’ in the carriage-control field of the fifth input record causes a page
end before condition cond1 is ever checked. Thus, the fifth input record is
processed using the first PRINTLINE command of the current page format.

<table>
<thead>
<tr>
<th>PPFA Commands</th>
<th>Input Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAGEFORMAT PFMTA ;</td>
<td>Carriage Control</td>
</tr>
<tr>
<td>PRINTLINE CHANNEL 1 ;</td>
<td>1 2 3 4 5 6</td>
</tr>
<tr>
<td>PRINTLINE ;</td>
<td>L I N E 1</td>
</tr>
<tr>
<td>PRINTLINE ;</td>
<td>L I N E 2</td>
</tr>
<tr>
<td>PRINTLINE ;</td>
<td>L I N E 3</td>
</tr>
<tr>
<td>PRINTLINE ;</td>
<td>L I N E 4</td>
</tr>
<tr>
<td>CONDITION cond1</td>
<td>L I N E 5</td>
</tr>
<tr>
<td>START 6 LENGTH 1</td>
<td></td>
</tr>
<tr>
<td>WHEN EQ '5'</td>
<td></td>
</tr>
<tr>
<td>AFTER SUBPAGE</td>
<td></td>
</tr>
<tr>
<td>CURRENT NULL;</td>
<td></td>
</tr>
</tbody>
</table>

**Considerations**

The PRINTLINE command is not processed if the PRINTLINE command is spaced
over, for example, when multiple line spacing causes certain PRINTLINE
commands to be bypassed.

If the input-record carriage-control field specifies a double space before print (for
example), and a CONDITION command is specified for the spaced line, the
CONDITION is ignored (never checked). Because the OTHERWISE subcommand
is part of a CONDITION command, the OTHERWISE subcommand is also ignored.

This can be confusing. You might expect an OTHERWISE condition to be true if all
other conditions have failed. In fact, the OTHERWISE condition can be true if it is
associated with a PRINTLINE command that is actually processed. See the
following diagram for an example of this. This assumes ANSI carriage controls
have been specified for this print file. ANSI carriage control ‘0’ means space two
lines before printing.

The fifth input record contains data (character ‘5’ in the sixth position) that would
normally satisfy the condition specified on the fifth PRINTLINE command. However,
the character ‘0’ in the carriage control field of input record 4 causes the fifth
PRINTLINE command to be ignored. The fifth input record is processed by the sixth
PRINTLINE command; therefore, the condition is not satisfied.
**WHEN CHANGE is Always False at Start of a Page Format**

**Rule**
The WHEN CHANGE process compares the contents of a given field with the contents of the same field in the last record that was processed with the current page format and current condition. Whenever a page format is started (either by a condition that changes page formats or when processing of the data file begins), a WHEN CHANGE condition is always false because the previous record was not processed with the current page format.

**Note:** The following meanings apply to the previous statement.
- `changes` switching to a page format that has a different name
- `data file begins` if conditional processing invokes the CURRENT data map, CHANGE information is retained

**Considerations**
Ensure that the WHEN CHANGE statement is processed before the switch to a new page format has been performed. See “Multiple CONDITION Commands” on page 89 for an example of how a combination of WHEN CHANGE BEFORE SUBPAGE and WHEN CHANGE AFTER SUBPAGE can lead to unexpected results.

**Relationship of CC and TRC fields to the START Subcommand**

**Rule**
The position specified by the START subcommand of the CONDITION command is in reference to the start of the data record. The first one or two bytes of an input record may contain either both a carriage-control character (CC) or a table-reference character (TRC). However, these characters are not considered part of the data record and are not to be counted when determining the START subcommand value. In the following example, the field being checked is actually the seventh character of the input record, but is the sixth character of the data record.
Using the CONDITION Command to Select a Copy Group and a Page Format

Rules

1. Within the CONDITION command, a copy group and a page format can be specified by using either a specific name or a parameter (CURRENT or =, FIRST, NEXT) or NULL or / can be specified. The use of the NULL or / parameters differs from the use of the others:

**Others**

When any parameter other than NULL or / is specified, the specifications for the copy group or page format selected replace the current specifications. When the current specifications are replaced, the action is referred to as starting or restarting the copy group or page format. In AFP terminology, an Invoke Medium Map (IMM) command is generated for a copy group and an Invoke Data Map (IDM) command is generated for a page format.

**NULL or /**

When NULL or / is specified, no IMM or IDM is generated and processing continues as if no condition check was present.

2. The COPYGROUP and the PAGEFORMAT parameters are positional. If both parameters are specified, the COPYGROUP parameter must be first. If you want only to specify the copy group, the PAGEFORMAT parameter can be omitted, or specified as NULL or /. However, if you want only to specify the page format, the COPYGROUP parameter must be specified as NULL or /.

Considerations

**Starting or Restarting a Copy Group:** When a copy group is started (or restarted), the remaining input data is forced to the start on the next sheet. Therefore, if duplex output was expected, but the copy group is restarted while processing the front side of a sheet, the remaining data will start on the front side of the next sheet rather than on the back side of the current sheet.

See “Duplex Output with Different Front and Back Print Directions” on page 87 for an example.

Furthermore, note that any copy group action except NULL restarts the page format (see the following item).
**Starting or Restarting a Page Format:** When a page format is started (or restarted), the remaining input data is forced to the start on the next side. Furthermore, that data is processed starting with the first PRINTLINE command in the specified page format. This is true even if CURRENT is specified as the page format parameter.

**Not Restarting a Copy Group:** If the copy group is not to be restarted, specify NULL or /. Do not specify COPYGROUP NULL or COPYGROUP /.

The following example illustrates this point. The command sequence on the left invokes a copy group named NULL. The command sequence on the right leaves the current copy group active.

<table>
<thead>
<tr>
<th>Incorrect Format</th>
<th>Correct Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONDITION condname</td>
<td></td>
</tr>
<tr>
<td>START ...</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>WHEN ...</td>
<td></td>
</tr>
<tr>
<td>COPYGROUP NULL</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>CONDITION condname</td>
<td></td>
</tr>
<tr>
<td>START ...</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>WHEN ...</td>
<td></td>
</tr>
<tr>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
</tbody>
</table>

**Not Restarting a Page Format:** If the page format is not to be restarted, specify NULL or / (or simply omit the specification). Do not specify PAGEFORMAT NULL or PAGEFORMAT /.

The following example illustrates this point. The command sequence on the left invokes a page format named NULL. The command sequence on the right will leave the current page format active.

<table>
<thead>
<tr>
<th>Incorrect Format</th>
<th>Correct Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONDITION condname</td>
<td></td>
</tr>
<tr>
<td>START ...</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>WHEN ...</td>
<td></td>
</tr>
<tr>
<td>COPYGROUP CGA</td>
<td></td>
</tr>
<tr>
<td>PAGEFORMAT NULL</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>CONDITION condname</td>
<td></td>
</tr>
<tr>
<td>START ...</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>WHEN ...</td>
<td></td>
</tr>
<tr>
<td>COPYGROUP CGA</td>
<td></td>
</tr>
<tr>
<td>NULL</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
</tbody>
</table>
Variable Length Records and the CONDITION Command

Considerations
The CONDITION command inspects a field that starts at a particular position and extends for a certain length. If the entire field is not available within the input record, the condition is always false. If the input file contains variable-length records, the record may not extend the full length specified by the START and LENGTH subcommands. In this way, a condition which seems as if it should be satisfied can actually fail.

Truncation of Blanks and the CONDITION Command

Considerations
Truncation occurs when blank characters are removed from the end of records on the spool. If blank truncation is in effect, the result can be the same as if the input file contained variable-length records.

Blank truncation is a consideration at the time the input records are passed to PSF. In the JES2 environment, blank truncation occurs unless the BLNKTRNC=NO parameter is specified. In the JES3 environment, blank truncation occurs unless the TRUNC=NO parameter is specified as part of either the BUFFER or SYSOUT initialization statements. Blank truncation can affect conditional processing since a field could “disappear” by being truncated causing no WHEN/OTHERWISE clause to be executed.

Conditional Processing Examples
This section provides conditional processing examples. The examples are grouped into functionally similar applications and are increasingly complex. The examples provided are:

- Jog output based on a change in the input data
- Duplex output with different front and back print directions
- Record reprocessing
- Select paper from an alternate bin
- Multiple CONDITION commands

Jog Output Example
This example shows how to jog the printed output, based on a change in the input data.

Copy group CGJOG specifies JOG YES. Page format PFJOG contains a CONDITION command that checks for any change in positions 8 through 10. If a change is detected, copy group CGJOG is restarted. Note that the only result is to start printing on a new sheet and to jog that sheet.
Jog Output Example

```
FORMDEF JOG;
COPYGROUP CGJOG JOG YES;

PAGEDEF JOG;
  PAGEFORMAT PFJOG WIDTH 11 IN HEIGHT 8.5;
  PRINTLINE REPEAT 50
    CHANNEL 1;
  CONDITION NEWPAGE START 8 LENGTH 3
    WHEN CHANGE BEFORE SUBPAGE
    COPYGROUP CGJOG;
```

Duplex Output with Different Front and Back Print Directions

This example shows how to establish one print direction on the front side and a different print direction on the back side of a duplex sheet.

The page definition in this example contains two page formats, each of which has a CONDITION statement that always returns a true value. The value is true because the character in position 1 will always have a value greater than or equal to hexadecimal zero. Therefore, every time a page change occurs (front to back, or back to next front) a different page format will be started. The different DIRECTION statements in the two page formats change the layout of the text on the page.

Note that the COPYGROUP parameter is specified as NULL. If a parameter other than NULL or / is specified for COPYGROUP, the copy group restarts every time a page change occurs. Because restarting a copy group forces data to a new sheet, duplex printing does not occur.

```
FORMDEF XMDUP
  DUPLEX NORMAL;

PAGEDEF XMDUP WIDTH 8.5 HEIGHT 11.0;
  PAGEFORMAT P2FRONT DIRECTION ACROSS;
    PRINTLINE CHANNEL 1 POSITION 0.75 TOP;
    CONDITION GOTOBACK START 1 LENGTH 1
      WHEN GE X'00' AFTER SUBPAGE NULL PAGEFORMAT P2BACK;
      PRINTLINE REPEAT 59;
  PRINTFORMAT P2BACK DIRECTION UP;
    PRINTLINE CHANNEL 1 POSITION 0.25 TOP;
    CONDITION GOTOFRNT START 1 LENGTH 1
      WHEN GE X'00' AFTER SUBPAGE NULL PAGEFORMAT P2FRONT;
      PRINTLINE REPEAT 59;
```
Record Reprocessing Example

This example uses the BEFORE SUBPAGE function with record reprocessing because the copy group and page format cannot be determined until input record 3 for each subpage has been read.

Notes:
1. This example includes two subpages.
2. The CONDITION command specifies that the action to be performed is NEWFORM. Therefore, if the condition is satisfied, the data in the current subpage is forced to start on the next form. If the data is already at the start of a new form, no action is performed. In other words, a blank page is not generated.

Record Reprocessing Example

/* Page definition for 2-up printing */
/* Test field in line 3 of each subpage */
/* Eject to new sheet if the field changes. */

PAGEDEF REPROC
  WIDTH 10.6 HEIGHT 8.3 DIRECTION DOWN;

PAGEFORMAT PFREPROC
  /* Definition of first subpage */
  PRINTLINE CHANNEL 1
    REPEAT 2
      POSITION MARGIN TOP;
  PRINTLINE REPEAT 1
    POSITION MARGIN NEXT;
    CONDITION EJECT
      START 5 LENGTH 5
      WHEN CHANGE BEFORE SUBPAGE
      NEWFORM;
  PRINTLINE REPEAT 40
    POSITION MARGIN NEXT;
  ENDSUBPAGE;

  /* Definition of second subpage */
  PRINTLINE CHANNEL 1
    REPEAT 2
      POSITION 5.3 TOP;
  PRINTLINE REPEAT 1
    POSITION 5.3 NEXT;
    CONDITION EJECT;
  PRINTLINE REPEAT 40
    POSITION 5.3 NEXT;
  ENDSUBPAGE;
Selecting Paper from an Alternate Bin Example

This example selects the first sheet from the alternate bin and all other pages from the primary bin. This function is useful when special paper (such as one having the company logo) is to be used for the first page of a document.

**Note:** Bin selection will be overridden by the printer should the form defined to each bin be the same form number. Only the primary bin is selected.

### Alternate Bin Example

/* The form definition contains two copy groups -- */
/* ALTBIN - for the first page */
/* PRIBIN - for all other pages */

FORMDEF BINEX
   DUPLEX NO;
   COPYGROUP ALTBIN BIN 2;
   COPYGROUP PRIBIN BIN 1;

PAGEDEF BINEX
   WIDTH 8.3 HEIGHT 10.6;

   /* Pageformat for first page - bin 2 */
   PAGEFORMAT FIRST;
   PRINTLINE CHANNEL 1
   POSITION MARGIN TOP;
   CONDITION GOTOировки START 1 LENGTH 1
   WHEN GE X'00' AFTER SUBPAGE
   COPYGROUP PRIBIN PAGEFORMAT REST;
   PRINTLINE REPEAT 59;

   /* Pageformat for all other pages - bin 1 */
   PAGEFORMAT REST;
   PRINTLINE CHANNEL 1
   POSITION MARGIN TOP
   REPEAT 60;

### Multiple CONDITION Commands

Two examples are shown here. The first example shows how two CONDITION commands can interact to give unintended results. The second example shows how to use the two CONDITION commands to achieve the correct results.

**Example 1 Multiple CONDITION Command—Incorrect Solution**

The example on page 90 demonstrates how two CONDITION commands can interact to give unintended results. Specifically, one CONDITION command causes a change of page format and then a second CONDITION command inspects a field with a WHEN CHANGE subcommand.

The purpose of condition:

**NEWREP**

Starts a new report on a new sheet of paper whenever the specified field changes and jogs the output so the report can be easily located.
SHIFTB and SHIFTF
Handles the situation where all four subpages of the front (or back) contain data.

In this situation, the objective is to change the print direction of the text on the page.

In the situation where both conditions seem to be true at the same time, the results may be unexpected.

**Note:** Condition SHIFTB (or SHIFTF) takes effect after the current subpage and therefore precedes the before subpage processing defined by condition NEWREP. Because condition SHIFTB results in starting a new page format, condition NEWREP returns a false value, and the expected new report processing is not performed.

**Example 2 Multiple CONDITION Command—Correct Solution**
The example on page 91 differs from the one on page 90 in two significant ways:

- Because the page format for the back side is the first one defined in the page definition, it is the one that is initially active
- Both CONDITION commands (NEWREP and SHIFTIT) specify that the action should happen before the current subpage has been processed

When processing begins, condition NEWREP fails because this is a WHEN CHANGE condition and the page format has just been started. However, condition SHIFTIT will return a true result, and the NEXT page format (PFFRONT) will be started. No lines have been formatted, so condition SHIFTIT has the effect of moving to the page format for the front side.
INCORRECT Solution Example

FORMDEF XMPICO OFFSET 0 0 DUPLEX RTUMBLE JOG YES REPLACE YES;
COPYGROUP CG1;
OVERLAY OVLY1;
OVERLAY OVLY2;
SUBGROUP OVERLAY OVLY1 FRONT;
SUBGROUP OVERLAY OVLY2 BACK;

PAGEDEF XMPICO REPLACE YES;
FONT GT24;
FONT GT12;
/* Definition of pageformat for front side */
PAGEFORMAT PFFRONT WIDTH 11 IN HEIGHT 8.5 IN DIRECTION UP;
SETUNITS 1 PELS 1 PELS LINESP 16 LPI;
PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 75 188;
CONDITION NEWREP START 8 LENGTH 3
WHEN CHANGE BEFORE SUBPAGE COPYGROUP CG1 PAGEFORMAT PFFRONT;
PRINTLINE REPEAT 40 FONT GT24 POSITION 75 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 1377 188;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 40 FONT GT24 POSITION 1377 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 75 1102;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 40 FONT GT24 POSITION 75 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 1377 1102;
CONDITION NEWREP START 8;
CONDITION SHIFTB START 1 LENGTH 1
WHEN GE X'00' AFTER SUBPAGE NULL PAGEFORMAT PFBACK;
PRINTLINE REPEAT 40 FONT GT24 POSITION 1377 NEXT;
ENDSUBPAGE;

/* Definition of pageformat for back side */
PAGEFORMAT PFBACK WIDTH 8.5 IN HEIGHT 11 IN DIRECTION ACROSS;
SETUNITS 1 PELS 1 PELS LINESP 8 LPI;
PRINTLINE REPEAT 1 CHANNEL 1 FONT GT12 POSITION 75 61;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 40 FONT GT12 POSITION 75 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT12 POSITION 75 1335;
CONDITION NEWREP START 8;
CONDITION SHIFTF START 1 LENGTH 1
WHEN GE X'00' AFTER SUBPAGE NULL PAGEFORMAT PFFRONT;
PRINTLINE REPEAT 40 FONT GT12 POSITION 75 NEXT;
ENDSUBPAGE;
FORMDEF XMPCOR OFFSET 0 0 DUPLEX RTUMBLE JOG YES REPLACE YES;
COPYGROUP CG1;
OVERLAY OVLY1;
OVERLAY OVLY2;
SUBGROUP OVERLAY OVLY1 FRONT;
SUBGROUP OVERLAY OVLY2 BACK;

PAGEDEF XMPCOR REPLACE YES;
FONT GT24;
FONT GT12;
/* The pageformat for the back side of the form is */
/* the first pageformat in the PAGEDEF. Therefore, */
/* it will initially be the active pageformat */
PAGEFORMAT PFBACK WIDTH 8.5 IN HEIGHT 11 IN DIRECTION ACROSS;
SETUNITS 1 PELS 1 PELS LINESP 8 LPI;
PRINTLINE REPEAT 1 CHANNEL 1 FONT GT12 POSITION 75 61;
CONDITION NEWREP START 8 LENGTH 3
  WHEN CHANGE BEFORE SUBPAGE COPYGROUP CG1 PAGEFORMAT
  PFFRONT;
CONDITION SHIFTIT START 1 LENGTH 1
  WHEN GE X’00’ BEFORE SUBPAGE NULL NEXT;
PRINTLINE REPEAT 4 GT12 POSITION 75 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 75 188;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 4 GT24 POSITION 75 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 1377 188;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 4 GT24 POSITION 1377 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 75 1102;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 4 GT24 POSITION 75 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 1377 1102;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 4 GT24 POSITION 1377 NEXT;
ENDSUBPAGE;

// This is the pageformat for the front side of the form. */
PAGEFORMAT PFFRONT WIDTH 11 IN HEIGHT 8.5 IN DIRECTION UP;
SETUNITS 1 PELS 1 PELS LINESP 16 LPI;
PRINTLINE REPEAT 1 CHANNEL 1 FONT GT23 POSITION 75 188;
CONDITION NEWREP START 8;
CONDITION SHIFTIT START 1;
PRINTLINE REPEAT 4 GT24 POSITION 75 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 1377 188;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 4 GT24 POSITION 1377 NEXT;
ENDSUBPAGE;

PRINTLINE REPEAT 1 CHANNEL 1 FONT GT24 POSITION 75 1102;
CONDITION NEWREP START 8;
PRINTLINE REPEAT 4 GT24 POSITION 75 NEXT;
ENDSUBPAGE;
**Field Processing When PRINTLINEs Are Repeated**

The following examples show the effect of the [LINE | FIELD ] parameter on **REPEAT n**.

The first PRINTLINE example uses FIELD type repetition. The second PRINTLINE example shows LINE type repetition.

**Note:** When LINE type repetition is used, SETUNITS LINESP may need to be set to a higher value to avoid over printing.

---

**REPEAT n type FIELD Example**

```
PAGEDEF rept01 WIDTH 8.0 IN
     HEIGHT 10.5 IN
     LINEONE 0.2 IN 0.2 IN
     DIRECTION ACROSS
     REPLACE YES;

     FONT normal CR1/zerodot SBCS ROTATION /zerodot;
     FONT italic CI1/zerodot SBCS ROTATION /zerodot;
     FONT bold CB1/zerodot SBCS ROTATION /zerodot;
     
     SETUNITS LINESP 6 LPI;

     PRINTLINE POSITION 1./zerodot IN 1./zerodot IN
     DIRECTION ACROSS
     FONT bold
     REPEAT 3 FIELD;
     FIELD POSITION 0.0 IN 0.0 IN
     DIRECTION ACROSS
     FONT normal
     START * LENGTH 20;
     FIELD POSITION 2.5 IN 0.0 IN
     DIRECTION DOWN
     FONT normal
     START * LENGTH 20;
     FIELD POSITION 2.5 IN 2.5 IN
     DIRECTION BACK
     FONT normal
     START * LENGTH 20;
     FIELD POSITION 0.0 IN 2.5 IN
     DIRECTION UP
     FONT normal
     START * LENGTH 20;
```

---
The next example shows Input Line Data.

### (Input) Line Data

<table>
<thead>
<tr>
<th>Field Type Repeat</th>
<th>Field Type Repeat</th>
<th>Field Type Repeat</th>
<th>Field Type Repeat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field Type Repeat</td>
<td>Field Type Repeat</td>
<td>Field Type Repeat</td>
<td>Field Type Repeat</td>
</tr>
<tr>
<td>Field Type Repeat</td>
<td>Field Type Repeat</td>
<td>Field Type Repeat</td>
<td>Field Type Repeat</td>
</tr>
<tr>
<td>Field Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
</tr>
<tr>
<td>Field Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
</tr>
<tr>
<td>Field Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
</tr>
<tr>
<td>Field Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
</tr>
<tr>
<td>Field Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
<td>Line Type Repeat</td>
</tr>
</tbody>
</table>

Notice that the fields are repeated based on the prior instance of the same field, and not the printline. This has advantages if special effects are desired.

Line Type Repeat is based on the printline. Good for sales tickets.

Generally, this type of repeat needs a SETUNITS LINESP command...

...so that lines won't overlap! This is SETUNITS LINESP 3 IN

### Sample Output

When the previous example is processed by PSF, the following output will be printed.
Field Type Repeat instance of the same special effects are based on the prior has advantages if

fields are repeated printline. This

Notice that the fields, and not the

desired. Generally, this type

of repeat needs a command...

Line Type Repeat

for sales tickets. Generally, this type

printline. Good

This is

SETUNITS LINESP 3 IN

...so that lines

won’t overlap!

SETUNITS LINESP

Good
Chapter 6. N_UP Printing

With N_UP printing, which is defined in the form definition, you can print up to four pages on a sheet of paper in simplex mode and up to eight pages in duplex mode. Each of these pages are independent, allowing use of different page formats and copy groups for each page. This provides significantly more flexibility and function that the traditional multiple-up capability which is defined in the page definition. Refer to “N_UP Compared to Multiple-up” on page 115 for more differences between N_UP printing and multiple-up printing.

There are two levels of N_UP: ²
- basic N_UP supported by older AFP printers: 3825, 3827, 3828, 3829, 3835, and 3900-001.
- enhanced N_UP supported by printers with the new Advanced Function Common Control Unit (AFCCU): IBM 3900 Wide and Duplex printers, 3935, 3130, 3160, and follow-on AFCCU printers.

N_UP Partitions and Partition Arrangement

A key concept in N_UP printing is the partition. In both basic and enhanced N_UP, each sheet of paper is divided into equal size areas called partitions. Pages are placed in these partitions in sequential order in basic N_UP. Pages are placed in relation to one or more of these partitions in enhanced N_UP. Knowing the partition arrangement is critical to designing applications using N_UP.

Note: If you are using basic N_UP printing with PSF set to DATAck=BLOCK, data must fall within the boundary of the partition. Any data placed outside the edge of the partition boundary will not be printed, and no error message will be generated. However, enhanced N_UP printing allows pages to overlap partitions. The only limits are that the pages must not extend beyond the boundaries of the physical sheet, and the pages must not exceed the total number of N_UP partitions specified for the sheet.

The number, size, and position of partitions are determined by three things:
- the N_UP value (1, 2, 3, or 4)
- the size and shape of the sheet of paper
- the form definition presentation options, PRESENT and DIRECTION

When printing in duplex mode, the same number of partitions is also defined for the back of the sheet. For normal duplex, back partitions are placed as if the sheet were flipped around its right side or y-axis. For tumble duplex, they are placed as if the sheet were flipped around its top edge, or x-axis. See Figure 48 on page 99 and Figure 49 on page 100 for illustrations of duplex partitions.

² You must have the correct level of PPFA to generate basic or enhanced N_UP commands and the correct level of Print Services Facility for your operating system to drive the printer in the basic or enhanced N_UP mode.
Figure 47 on page 98 through Figure 50 on page 101 show the partition arrangement that results from every combination of N_UP value, paper size, and presentation option.

Use these figures to determine how your N_UP application will be formatted by the printer. In the figures, each equal-sized partition has a number indicating its default presentation sequence. The origin for each partition is in the same relative position as the origin point shown for the medium. This point serves as the top left corner for a page printed in the ACROSS (or 0°) printing direction. Figure 47 through Figure 50 on page 101 also show the way data formatted in the DOWN printing direction is printed when you use a DOWN copy group.

---

**Figure 47. N_UP 1 Partition Arrangement**
### Figure 48. N_UP 2 Partition Arrangement

<table>
<thead>
<tr>
<th>Cut Sheet</th>
<th>Across</th>
<th>Down with Down Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Portrait</td>
<td>Landscape</td>
<td>Portrait</td>
</tr>
<tr>
<td><img src="image1" alt="Diagram" /></td>
<td><img src="image2" alt="Diagram" /></td>
<td><img src="image3" alt="Diagram" /></td>
</tr>
<tr>
<td>Wide Fanfold</td>
<td><img src="image4" alt="Diagram" /></td>
<td><img src="image5" alt="Diagram" /></td>
</tr>
<tr>
<td>Narrow Fanfold</td>
<td><img src="image6" alt="Diagram" /></td>
<td><img src="image7" alt="Diagram" /></td>
</tr>
</tbody>
</table>

Legend:
- Medium origin
- Orientation of X axis on the medium
Figure 49. N_UP 3 Partition Arrangement
**Basic N_UP Printing**

You can specify the N_UP subcommand on either the FORMDEF or COPYGROUP commands in the form definition. Figure 51 on page 102 shows the subcommands and parameters enabled with basic N_UP printing.

The N_UP subcommand divides the medium into one, two, three, or four partitions, as described in “N_UP Partitions and Partition Arrangement” on page 97. The OVERLAY subcommand prints a page overlay in each partition at a specified offset from the page origin or the partition origin. For more information about page overlays, see “Medium Overlays and Page Overlays” on page 114.

The INVOKE subcommand controls the action that occurs if you invoke a new copy group. You can invoke copy groups using conditional processing in the page definition or by including an Invoke Medium Map (IMM) structured field in the print data. The default action is to eject to a new sheet. By specifying an INVOKE subcommand on a COPYGROUP command, you can instead eject to a new N_UP

---

**Figure 50. N_UP 4 Partition Arrangement**

<table>
<thead>
<tr>
<th>Across</th>
<th>Down with Down Data</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Portrait</strong></td>
<td><strong>Landscape</strong></td>
</tr>
<tr>
<td><img src="image" alt="Cut Sheet" /></td>
<td><img src="image" alt="Cut Sheet" /></td>
</tr>
<tr>
<td><img src="image" alt="Narrow Fanfold" /></td>
<td><img src="image" alt="Narrow Fanfold" /></td>
</tr>
</tbody>
</table>

Legend:  
- Medium origin
- Orientation of X axis on the medium
partition, which may be on the same sheet. If printing in duplex mode, you can specify whether to eject to a partition on the front or back side of the sheet.

You must use page overlays instead of medium overlays if you want to change overlays while ejecting to a new partition. PSF honors the NEXT, FRONT, and BACK values of the INVOKE subcommand only if the new copy group has the same medium modifications as the previous copy group. Medium modifications include duplexing, bin, page offset, N_UP values, presentation, direction, and medium overlays. If any of these modifications differ, PSF ejects to a new sheet when the copy group is invoked.

By combining INVOKE with the N_UP OVERLAY subcommand, you can place different page overlays in different partitions when you change copy groups. This is illustrated in “Basic N_UP Example 1: Using INVOKE and OVERLAY.”

The following examples show the use of basic N_UP. Because each example builds on the previous one, read them in sequential order to better understand basic N_UP. All the pages used in the examples are formatted in the ACROSS printing direction. Their orientation on the media is the result of using the available PRESENT and DIRECTION combinations in the FORMDEF command.

**Basic N_UP Example 1: Using INVOKE and OVERLAY**
Figure 52. Basic N_UP Example 1: Using INVOKE and OVERLAY

```
FORMDEF TWOUPS ;
  COPYGROUP A
    N_UP 2
    OVERLAY A
    INVOKE NEXT ;
  COPYGROUP B
    N_UP 2
    OVERLAY B
    INVOKE NEXT ;
```

Figure 53. Form Definition for Basic N_UP Example 1

Figure 52 shows the INVOKE and OVERLAY functions of basic N_UP printing. Specifying INVOKE NEXT on the COPYGROUP command ensures that when the copy group is invoked by an Invoke Medium Map (IMM) structured field with conditional processing, the next page will be placed in the next partition of the N_UP form.

The OVERLAY subcommand specifies a page overlay, which can be positioned relative to the page origin or relative to the partition origin. In basic N_UP, the
OVERLAY subcommand prints the overlay with the page data in every partition on the sheet. However, as you will see in this example, using INVOKE NEXT allows the application to use different overlays in different partitions.

Example 1 has been defined as N_UP 2 simplex with the default PORTRAIT ACROSS presentation, which results in the partitions illustrated in Figure 52 on page 103. The application uses different page formats on different application pages. With N_UP, changing page formats ejects to the next partition, just as it ejects to a new page in applications without N_UP.

The application also needs different overlays on different pages. Because the overlays are specified on N_UP in the COPYGROUP subcommand, the application accomplishes this by changing copy groups. Without the INVOKE subcommand, changing the copy group forces an eject to a new physical sheet. However, because INVOKE NEXT is specified, the eject will be to the next partition. Changing to copy group B after page 1 is written places page 2 in partition 2 of the same physical sheet. If the change were made after a page is placed in partition 2, the eject to the next partition would be to partition 1 of the next sheet. The page will be printed with the overlay specified in the new copy group.

Notes:

1. The pages in this example are line-format print data, formatted using a page definition. The example would be the same for MO:DCA data, except that page formats would not be used.

2. You can select the copy groups and page formats by including IMM and IDM structured fields in the print data or by using conditional processing in the page formats.

3. Overlays can be defined as page overlays in the page definition or in the form definition N_UP or PLACE subcommands. Overlays can also be defined as medium overlays in the form definition SUBGROUP command. If you want to change overlays when ejecting to a new partition, use page overlays instead of medium overlays. See “Medium Overlays and Page Overlays” on page 114 for information about page and medium overlays.

Basic N_UP Example 2: Normal Duplex
Figure 54. Basic N_UP Example 2: Normal Duplex

```
FORMDEF NUPDUP
N_UP 2
PRESENT PORTRAIT
DIRECTION ACROSS
DUPLEX NORMAL ;
```

Figure 55. Form Definition for Basic N_UP Example 2: Normal Duplex

Figure 54 shows the partition order for duplexed pages. This figure also shows the partitions into which the sheet is divided for N_UP 2 with PORTRAIT presentation and ACROSS direction. With normal duplex, the sheet is rotated around its “y-axis,” which is the right edge of the sheet. The result is that partition 2 for the back side of the sheet is on the back of partition 1 for the front side, and page 4 is on the back of page 1. The tops of pages 3 and 4 are aligned with the tops of pages 1 and 2.
Basic N_UP Example 3: Tumble Duplex

Figure 56. Basic N_UP Example 3: Tumble Duplex

```
FORMDEF NUPTUM
N_UP 2
PRESENT PORTRAIT
DIRECTION ACROSS
DUPLEX TUMBLE ;
```

Figure 57. Form Definition for Basic N_UP Example 3: Tumble Duplex

Figure 56 shows the partition order for tumble duplex pages. This figure also shows the partitions into which the sheet is divided for N_UP 2 with PORTRAIT presentation and ACROSS direction. With tumble duplex, the sheet is rotated around its x-axis, which is the top of the sheet. The result is that partition 1 of the back of the sheet falls on the back of partition 1 for the front, and page 3 falls on the back of page 1. The tops of pages 3 and 4 are aligned with the bottoms of pages 1 and 2. For more information about normal and tumble duplex printing, refer to “Normal Duplex and Tumble Duplex” on page 12.

Enhanced N_UP Printing

Enhanced N_UP is supported on the newest family of AFP printers, which includes the 3900-0W1 Wide, the 3900 Duplex, the 3935, 3130, and 3160. In addition to all the function of basic N_UP, enhanced N_UP includes the powerful PLACE subcommand.
Using the PLACE subcommand, you can place pages in the partitions in any sequence, specify unique overlays for each page, and rotate both the page and the overlays in the partitions. You can place multiple pages in the same partition and no pages in other partitions, and you can extend pages across partition boundaries. In short, you can place pages of any size at any location on the front or back of the sheet, in any orientation. The only limits are that the pages must not extend outside the boundaries of the physical sheet, and the pages must not exceed the total number of N_UP partitions specified for the sheet.

You use a single PLACE command to place each page of data on the sheet. You must specify the same number of PLACE commands as the number of N_UP partitions for the sheet. This is required for error recovery and restart integrity. If you do not want to place as many pages as partitions, you can specify CONSTANT on a PLACE command to indicate that no data is to be placed in the partition. You can specify the OVERLAY subcommand with the CONSTANT subcommand to place overlays without user’s data. Figure 58 shows the subcommands and parameters enabled with enhanced N_UP printing.

For most applications, place constant overlays before placing data on the sheet. This is because the overlay is not actually placed until the next page of data is placed. If your application changes copy groups or runs out of pages on the sheet before reaching the constant overlay PLACE subcommand, the constant overlay will not be printed. However, if you do not want the overlays to print in these cases, place the constant overlay after placing the page data.

```
FORMDEF Subcommands
[N_UP { 1 | 2 | 3 | 4 }] [ { OVERLAY name [ x-pos y-pos ] } ... ]
  { PLACE { n [ FRONT | BACK ] } [ CONSTANT ]
    [ OFFSET x [ unit ] y [ unit ] ]
    [ OVERLAY name [ x [ unit ] y [ unit ] ] &\r
      k. PARTITION ]
  } ... ]
[ INVOKE { SHEET | NEXT | FRONT | BACK } ]

COPYGROUP Subcommands
[N_UP { 1 | 2 | 3 | 4 }] [ { OVERLAY name [ x-pos y-pos ] } ... ]
  { PLACE { n [ FRONT | BACK ] } [ CONSTANT ]
    [ OFFSET x [ unit ] y [ unit ] ]
    [ OVERLAY name [ x [ unit ] y [ unit ] ] &\r
      k. PARTITION ]
  } ... ]
[ INVOKE { SHEET | NEXT | FRONT | BACK } ]
```

Figure 58. Subcommands for Enhanced N_UP Printing

The following examples show enhanced N_UP printing. Read these examples in sequence to better understand enhanced N_UP printing.
**Enhanced N_UP Example 1: Using PLACE**

![Diagram showing Enhanced N_UP Example 1: Using PLACE]

**Figure 59. Enhanced N_UP Example 1: Using PLACE**

```plaintext
FORMDEF BOOKLT DUPLEX NORMAL
    N_UP 2
    /* Page 1 */ PLACE 2 FRONT
    /* page 2 */ PLACE 1 BACK
    /* Page 3 */ PLACE 2 BACK
    /* Page 4 */ PLACE 1 FRONT ;
```

**Figure 60. Form Definition for Enhanced N_UP Example 1**

Figure 59 shows the function of the PLACE subcommand in specifying the sequence of partitions into which pages will be placed. This example is N_UP 2 duplex. The default partition sequence is from left to right. Notice that when printing in normal duplex, partition 1 on the back of the sheet aligns with partition 2 on the front of the sheet. See “Basic N_UP Example 2: Normal Duplex” on page 104 and “Basic N_UP Example 3: Tumble Duplex” on page 106 for information on N_UP duplex partitions.

For this booklet, you do not want to print pages in the default order: partitions 1 and 2 on the front, followed by partitions 1 and 2 on the back. Instead, print the pages so that when the sheet is folded, you will have a booklet, with page 1 on the front outside of the booklet, pages 2 and 3 inside the folded booklet, and page 4 on the back outside of the booklet. The form definition shown in Figure 60 uses the PLACE subcommand of enhanced N_UP to place pages in the partitions in the order needed to accomplish this. The application writes the pages in order, page 1 through page 4, and the N_UP form definition provides the correct sequencing in the partitions.
Enhanced N_UP Example 2: Using CONSTANT and OVERLAY

**Figure 61. Enhanced N_UP Example 2: Using CONSTANT and OVERLAY**
Figure 61 on page 109, introduces the CONSTANT subcommand of enhanced N_UP and shows the functions of the PLACE subcommand, which was described in “Enhanced N_UP Example 1: Using PLACE” on page 108 and the INVOKE subcommand, which was described in “Basic N_UP Example 1: Using INVOKE and OVERLAY” on page 102. This figure represents a user application that is printing customer statements using the values N_UP 2 duplex. The PLACE subcommand places the pages in the correct order for post-processing equipment to cut the sheets into individual pages and interleave them to produce sequential pages. The INVOKE subcommand guarantees that one customer's statement will never be printed on the back side of another customer's statement. The N_UP 2 subcommand, combined with the default PORTRAIT ACROSS presentation, divides the sheet into the two partitions illustrated in Figure 61 on page 109.

In Figure 61 on page 109, page 1 of each customer's statement is printed with overlay A. The back side of page 1 is a constant overlay, with no user's data. The remaining pages of each customer's statement are printed with overlay B.

The copy groups place the required overlays on both the right and left halves of the sheet, so that a new customer statement can begin on either half of the sheet. COPYGROUP PAGEN assigns overlay B to all partitions on the sheet. COPYGROUP PAGE1 assigns overlay A to all front partitions and overlay C to all back partitions. The CONSTANT parameter used with OVERLAY C means that no user's data will be printed in the partition with the overlay. To guarantee that the constant overlay prints whenever page 1 is printed, the PLACE subcommand for the constant overlay is specified before the PLACE subcommand for page 1 print data. The INVOKE subcommand specifies BACK to ensure that the overlay is printed on the back of the partition.

In the application shown in Figure 61 on page 109, the copy group is changed to PAGEN after page 1 is printed. Because the constant overlay and page 1 were printed with the first two PLACE commands of copy group PAGE1, the third PLACE command in new copy group will be used for the next page. Page 2 of statement 1
will be placed in partition 2 front, as specified in the third PLACE subcommand of copy group PAGEN.

After the fourth and last page of statement 1, the copy group is changed back to PAGE1 to print page 1 of statement 2. Page 4 of statement 1 printed in front partition 1 using the first PLACE subcommand of copy group PAGEN. N_UP selects the second PLACE subcommand of copy group PAGE1: PLACE 1 FRONT. But the INVOKE subcommand for copy group PAGE1 specifies BACK. N_UP continues sequentially through the PLACE subcommands of copy group PAGE1 until it finds a BACK partition. This is the third PLACE subcommand: PLACE 1 BACK CONSTANT OVERLAY C. The constant overlay is placed in partition 1 on the back of the sheet, then page 1 of the new customer’s statement is printed using the next PLACE subcommand: PLACE 2 FRONT on the front side of the constant overlay.

**Note:** You can use NEXT, FRONT, or BACK on the INVOKE subcommand only when switching between copy groups that have identical medium modifications. This includes identical N_UP values and an identical number of PLACE subcommands. If the copy groups have different values, the INVOKE command causes an eject to a new physical sheet.

**Enhanced N_UP Example 3: Asymmetric Pages**

![Diagram](image)

**Figure 63. Enhanced N_UP Example 3: Asymmetric Pages**
Figure 63 shows the flexibility and power of enhanced N_UP printing. With enhanced N_UP printing, you can place pages relative to any partition on the sheet, front or back, in any sequence. Pages are not limited by partition boundaries. The only limitations are that pages must not print outside the physical form boundaries, and you cannot place more pages on a sheet than the number specified in the N_UP subcommand. For an N_UP 4 duplex page, the limit is eight pages total on front and back sides combined. For N_UP 3 duplex, the limit is six pages on the front and back combined.

```plaintext
FORMDEF ASYMET DUPLEX NORMAL

COPYGROUP FIRST
  PRESENT LANDSCAPE DIRECTION ACROSS
  N_UP 4
  /* Constant*/ PLACE 1 BACK OFFSET 4 0 CONSTANT OVERLAY B
  /* Page 1 */ PLACE 1 FRONT OFFSET 0 0 Overlay A
  /* Page 2 */ PLACE 1 FRONT OFFSET 12 0
  /* Page 3 */ PLACE 1 BACK OFFSET 0 0
  /* Page 4 */ PLACE 1 FRONT OFFSET 12 4
  /* Page 5 */ PLACE 1 BACK OFFSET 0 4
  /* Page 6 */ PLACE 1 FRONT OFFSET 12 8
  /* Page 7 */ PLACE 1 BACK OFFSET 0 8

COPYGROUP SECOND
  PRESENT PORTRAIT DIRECTION ACROSS
  N_UP 3
  /* Constant*/ PLACE 1 BACK OFFSET 0 0 CONSTANT OVERLAY C
  /* Page 8 */ PLACE 1 FRONT OFFSET 0 0
  /* Page 9 */ PLACE 1 FRONT OFFSET 6 0
  /* Page 10 */ PLACE 1 FRONT OFFSET 6 4
  /* Page 11 */ PLACE 1 FRONT OFFSET 6 4
  /* 6th place */ PLACE 1 BACK OFFSET 0 0 CONSTANT
```

Figure 64. Form Definition for Enhanced N_UP Example 3

To achieve the asymmetrical page placement shown in this example, place all the pages relative to the origin of partition 1 on the front or the back side of the sheet. You can place the pages relative to the origin of any of the partitions, but using partition 1 simplifies the calculations for page positions.

With N_UP 4, the default PORTRAIT presentation and ACROSS direction place the origin at the top right of the partition on wide, continuous-form paper. In this example, specifying LANDSCAPE ACROSS sets the origin at the top-left corner, to achieve the correct page arrangement.

The coding of the form definition for example 3 is shown in Figure 64. Copy group FIRST specifies N_UP 4, which requires eight PLACE subcommands for the duplex page. Note that the constant overlay B on the back of the sheet represents one of the eight PLACE subcommands. COPYGROUP SECOND used for the second sheet specifies N_UP 3. You must use six PLACE subcommands. Four pages are placed on the front side, and a constant overlay is placed on the back, using five of the six PLACE subcommands. A CONSTANT page is specified without an overlay to fill the sixth PLACE subcommand. Nothing will be printed with this PLACE
subcommand, but it is required to ensure a correct internal page count for recovery and restart.

**Note:** In each copy group, the PLACE subcommand for the constant overlay is placed in front of all the PLACE subcommands for page data. This placement ensures that the constant overlay prints if any pages are printed on the sheet. Otherwise, if you change copy groups or run out of pages before the PLACE command for the constant overlay, the overlay will not print.

---

### Additional N_UP Considerations

N_UP can affect the scope of other PPFA commands that operate on a page or a medium.

**COPIES**
The COPIES subcommand in the SUBGROUP of the form definition operates on the physical medium. When you specify five copies using N_UP 2, you will get five sheets of the N_UP 2 data.

**SUPPRESSION**
The SUPPRESSION subcommand in the SUBGROUP of the form definition operates on the physical medium. The suppression names in the SUBGROUP operate on all N_UP pages on the sheet.

**OVERLAY**
You can specify an OVERLAY subcommand in multiple places in the form definition and can also specify an overlay in the page definition. The result will be either a page overlay or a medium overlay. See “Medium Overlays and Page Overlays” on page 114 for a description of the differences between these commands and the uses of these overlays.

**PRESENT DIRECTION**
You use the PRESENT and DIRECTION subcommands of the form definition with the N_UP subcommand to determine partition arrangement. These commands, which are described in this update guide, now affect all N_UP printers, including cut-sheet printers.

**CONDITION**
You can use the CONDITION command of the page definition with N_UP just as you use it with non N_UP jobs. However, the NEWSIDE and NEWFORM parameters may operate differently than you expect. NEWSIDE, which is equivalent to invoking a new page format, will eject to the next partition, which may not be on a new side of an N_UP sheet. NEWFORM, which is equivalent to invoking a new copy group, will eject to a new sheet with basic N_UP. The effect with enhanced N_UP depends on the coding of the INVOKE subcommand.
Medium Overlays and Page Overlays

An AFP overlay can be used as a page overlay or as a medium overlay. Different actions are performed on these two different types of overlays. Page overlays apply to the page and are placed relative to the page origin. Medium overlays always apply to the entire medium and are placed at the medium origin. When used with N_UP, the medium overlay still applies to the entire sheet of paper, not to the individual partitions.

The same overlay can be either a page overlay or a medium overlay, depending on the method used to invoke it for printing. An overlay invoked by a page definition or by an Include Page Overlay (IPO) structured field is always a page overlay. An overlay invoked by a form definition without N_UP is always a medium overlay. When N_UP is specified in the form definition, you can specify commands to invoke a page overlay. The examples below show the ways in which overlays can be invoked.

```
PAGEDEF EXMPL1;
PAGEFORMAT P2EXMPL1;
OVERLAY EXMPL1; /* Allows this page overlay to be invoked by an IPO structured field */
PRINTLINE REPEAT 60; /* coded in the print data */
```

Figure 65. Page Overlay Invoked by an IPO Structured Field

```
PAGEDEF EXMPL2;
PAGEFORMAT P2EXMPL2;
OVERLAY EXMPL2; /* Optional. Stores overlay for reuse */
PRINTLINE REPEAT 1
   POSITION 1 IN 1 IN
   OVERLAY EXMPL2 /* Prints overlay if data prints on printline */
   -1 IN -1 IN /* Positions overlay relative to printline */
   PRINTLINE REPEAT 50;
```

Figure 66. Page Overlay Invoked by a PRINTLINE Command

```
FORMDEF EXMPL3;
COPYGROUP F2EXMPL3
   DUPLEX NORMAL;
OVERLAY XMPL3F; /* Allows SUBGROUP to invoke overlay */
OVERLAY XMPL3B; /* Allows SUBGROUP to invoke overlay */
SUBGROUP FRONT
   OVERLAY XMPL3F; /* Prints overlay on front of every form */
SUBGROUP BACK
   OVERLAY XMPL3B; /* Prints overlay on back of every form */
```

Figure 67. Medium Overlay Invoked by a Form Definition
N_UP Compared to Multiple-up

With the addition of the N_UP capability, AFP now provides two methods to format multiple application pages on a single sheet:

- N_UP as defined in a form definition
- Multiple-up as defined in a page definition

The multiple-up function has long been available for line-format data printed on AFP printers. Multiple-up achieves the appearance of multiple pages on a sheet by formatting multiple groups of print lines as a single AFP page. The output is still a single AFP page on a side of a sheet, and the entire output is formatted by a single page format. If the application pages within that sheet require different print layouts, you must design a different page format for all possible arrangements of data. For example, if one side of a 2-up sheet has ten different print layouts, you need 100 different page formats to cover all the possible combinations.

In contrast, N_UP enables you, for the first time in AFP, to place multiple AFP pages on a side of a sheet. This means that each of the N_UP pages can be formatted using a different page format. You can change page formats between each N_UP page without ejecting to a new side of the sheet. For the same example with N_UP, you need only ten page formats for a 2-up sheet with ten different print layouts.

N_UP also means you can place multiple pages of fully-composed AFP data (or MO:DCA data) on a single sheet. This was not possible using the multiple-up function defined in the page definition, because AFP data does not use a page definition.
# Part 3. PPFA Commands and Syntax

## Chapter 7. PPFA Command Syntax

- Rules for Creating a PPFA Command Stream ........................................ 119
- Token Rules ......................................................................................... 120
- Character Set ...................................................................................... 120
- Command Delimiters .......................................................................... 121
- Blanks and Blank Lines ...................................................................... 121
- Names .................................................................................................. 121
- Comments ........................................................................................... 122
- Literals ................................................................................................. 122
- Numeric Values ................................................................................... 123
- Units of Measurement ......................................................................... 123
- Notational Conventions ...................................................................... 124

## Chapter 8. Form Definition Command Reference

- Sequence of Commands for Form Definitions .................................. 125
- COPYGROUP Command ...................................................................... 126
  - Subcommands .................................................................................. 127
- FORMDEF Command .......................................................................... 135
  - Subcommands ................................................................................ 135
- OVERLAY Command ......................................................................... 144
  - Subcommand .................................................................................. 145
- SETUNITS Command ......................................................................... 145
  - Subcommand .................................................................................. 146
- SUBGROUP Command ......................................................................... 146
  - Subcommands ................................................................................ 147
- SUPPRESSION Command ................................................................... 150

## Chapter 9. Page Definition Command Reference

- Sequence of Commands for Page Definitions .................................. 151
- CONDITION Command ....................................................................... 152
  - Subcommands ................................................................................ 153
- ENDSUBPAGE Command .................................................................... 158
- FIELD Command .............................................................................. 158
  - Subcommands ................................................................................ 160
- FONT Command ................................................................................ 170
  - Subcommands ................................................................................ 171
- OVERLAY Command ......................................................................... 172
- PAGEDEF Command .......................................................................... 173
  - Subcommands ................................................................................ 173
- PAGEFORMAT Command .................................................................... 175
  - Subcommands ................................................................................ 176
- PRINTLINE Command ....................................................................... 178
  - Subcommands ................................................................................ 179
- SEGMENT Command .......................................................................... 186
- SETUNITS Command ......................................................................... 187
  - Subcommand .................................................................................. 188
- TRCREF Command ............................................................................. 188
  - Subcommands ................................................................................ 189
Chapter 7. PPFA Command Syntax

PPFA controls are made up of four elements: commands, subcommands, parameters, and literals.

- **Commands** are controls representing the major functions of PPFA and are separated from other commands by semicolons. Each command has its own entry in Chapter 8, “Form Definition Command Reference” on page 125 and in Chapter 9, “Page Definition Command Reference” on page 151.
- **Subcommands** fall within commands and specify the function of that command.
- **Parameters** specify the values for one subcommand.
- **Literals** consist of fixed text included in a field definition or as constant data for comparison in a conditional processing definition.

Rules for Creating a PPFA Command Stream

When you create a PPFA command stream, follow these rules:

- You cannot intermix uppercase and lowercase characters to write commands, subcommands, and literals. Before processing the commands, PPFA converts lowercase characters into uppercase characters, except those in literals. Thus, it does not discriminate between uppercase and lowercase characters. For example, `OVERLAY abc` and `overlay ABC` produce the same results because both `overlay` and `abc` are converted to uppercase.
- Commands and subcommands can be abbreviated to the first five characters, which are always unique. For example, PRINTLINE can be abbreviated to PRINT, FORMDEF to FORMD, CHANNEL to CHANN, and so forth.
- User names for form definitions and page definitions must not be the same as PPFA command names and subcommand names. These are reserved words. For example, REPEAT or CHANNEL must not be form-definition names.
- The subcommands governed by a command can be entered in any order; however, the name of a font or form definition, for example, must come immediately after the object being named. Parameters defined in a subcommand must be entered immediately after the subcommand.
- Commands must end with a semicolon.
- A command or subcommand can start in any column and can continue on the next line without a continuation indicator.
- More than one form definition and page definition can be specified in a job stream.
- PPFA neither checks nor sets default values for items that depend on printer hardware.
Token Rules

Tokens are character strings, within a set of PPFA commands, that PPFA recognizes as units. Tokens include:

- Both local names and user-access names for fonts, form definitions, page definitions, overlays, and suppressions
- Commands
- Subcommands
- Parameters
- Literals
- Special characters

The only PPFA element that is not a token is a blank.

To create a token, you must separate a string from the previous token by either a special character or a blank. See the list of special characters in “Character Set.” Thus, A+B is the same as A + B, because + is a special character. But AB is not the same as A B. The blank in A B creates two tokens.

Character Set

The four types of characters are alphabetic, numeric, blank, and special. Characters of each type are as follows:

- The following are PPFA alphabetic characters:
  
  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
  a b c d e f g h i j k l m n o p q r s t u v w x y z
  # @ $

- The following are PPFA numeric characters:
  
  0 1 2 3 4 5 6 7 8 9

- The blank character has a character code of X'20' in ASCII (which is the data stream used for creating the form definition or page definition

  **Note:** In EBCDIC data, the blank character has a character code of X'40'.

- The following are PPFA special characters:
  
  . ( + * ) = \% = ; / &

- The following are EBCDIC shift-out and shift-in codes:
  
  X'0E', the shift-out (SO) code
  X'0F', the shift-in (SI) code

Other character codes are also allowed within comments and literals. See “Comments” on page 122 and “Literals” on page 122 for details of what can be included.
Command Delimiters
A command always ends with a semicolon. One command can extend over several lines and does not end until a semicolon appears.

Blanks and Blank Lines
Blanks and blank lines can occur anywhere and have no effect on the processing of PPFA. The “;” is the command delimiter.

Names
The maximum number of alphanumeric characters in a PPFA name varies. Table 6 shows the number of characters allowed in the PPFA names.

<table>
<thead>
<tr>
<th>Type of Name</th>
<th>Number of Characters Allowed</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Form Definition</strong></td>
<td></td>
</tr>
<tr>
<td>COPYGROUP</td>
<td>1–8</td>
</tr>
<tr>
<td>FORMDEF</td>
<td>1–6</td>
</tr>
<tr>
<td>OVERLAY (local name)</td>
<td>1–16</td>
</tr>
<tr>
<td>OVERLAY (user-access name)</td>
<td>1–6</td>
</tr>
<tr>
<td>SUPPRESSION</td>
<td>1–8</td>
</tr>
<tr>
<td><strong>Page Definition</strong></td>
<td></td>
</tr>
<tr>
<td>CONDITION</td>
<td>1–8</td>
</tr>
<tr>
<td>FONT (local name)</td>
<td>1–16</td>
</tr>
<tr>
<td>FONT (user-access name)</td>
<td>1–6</td>
</tr>
<tr>
<td>OVERLAY</td>
<td>1–6</td>
</tr>
<tr>
<td>PAGEDEF</td>
<td>1–6</td>
</tr>
<tr>
<td>PAGEFORMAT</td>
<td>1–8</td>
</tr>
<tr>
<td>SEGMENT</td>
<td>1–6</td>
</tr>
</tbody>
</table>

Note: The name can consist of the characters shown under the “Alphabetic Characters” and the “Numeric Characters” in “Character Set” on page 120. A local name identifies a font or an overlay only within a set of PPFA commands. A user-access name is the name the operating system uses to find the data. PPFA adds the appropriate prefix to the user-access name (for example, F1 for form definitions, P1 for page definitions, and O1 for overlays) to match the library resource name.

Two PPFA commands can be used to equate the local name and the user-access name.

OVERLAY     Within a form definition
FONT        Within a page definition
Comments

Programmer comments used to document PPFA command streams are allowed anywhere within the command stream. Comments must be enclosed with the delimiters /* and */. A comment is allowed anywhere a blank is allowed and can continue for any number of lines.

**Note:** For VSE, however, a comment must not start at the beginning of the line. A /* specified as the first two bytes of a record in PPFA running under VSE is interpreted as the end of system input.

The following example shows the available variations in comment formats:

```plaintext
FIELD /* comment */ FONT GT10 /* comment,
multiline comment,
more comment */ START * + 10 LENGTH 5;
FIELD LENGTH 10; FIELD START * + 10 LENGTH 15;
```

**Notes:**
1. A comment must end with the closing delimiter (*/) .
2. Double-byte character codes in comments must be enclosed within SO (X'0E') and SI (X'0F') on EBCDIC platforms.

Literals

A literal is any material specified in single quotation marks. Literals can be used within a:

- TEXT subcommand to create fixed text for a page definition
- WHEN subcommand to define constant text for comparison

Literals can contain any characters in any position, except those that have special syntactic meanings. Single quotation marks may be used within a literal only if they are entered in pairs ('). PPFA translates a pair of single quotation marks into one quotation mark. For example, 'JOAN''S' yields JOAN'S.

A literal can continue for any number of lines. For example:

```plaintext
TEXT 'THIS IS ' 'A LITERAL' /* The four separated */
   'THE TEXT SPANS' /* text elements will produce*/
   'THREE LINES' ; /* one sequence of text */
TEXT 'X'0101'
   'X'ABAB' /* spanning three lines */
   'X'BABB' ;
TEXT 'K'100,200' /* kanji numbers */
   'K'321,400' ; /* specified sequentially */
```

Invalid:

```plaintext
TEXT 'THIS IS'
   'K'100,200' ; /* Mixing single-byte and double-byte characters in one field is not allowed */
```

A double-byte literal must be enclosed within apostrophe shift-out (X'7D0E') and shift-in apostrophe (X'0F7D').
Numeric Values

Numeric variables are specified as decimal numbers; up to three decimal places can be specified.

Units of Measurement

Numbers used to specify dimensions in form definitions and page definitions can be in any of five units of measurement. They are specified in a command stream as follows:

- IN (inches)
- MM (millimeters)
- CM (centimeters)
- POINTS — Points are a common measurement in printing used to measure character height, as in 20-point type. A point is approximately 1/72 inch.
- PELS (equates to L-units under the SETUNITS command)—There are 240 L-units to the inch.

Two additional measurement units can be used in the SETUNITS command; the measurement units are:

- LPI (lines per inch)
- CPI (characters per inch)

The parameters in PPFA that define a measurement can include any of the first five units of measurement shown in the previous list. For example:

POSITION 1 IN 1 IN ;

or

POSITION 1 MM 1 MM ;

However, PPFA converts all measurements to logical units (L-units) as the common measurement (one inch equals 240 L-units). If a fraction exists, the first decimal point is truncated. A SETUNITS command defines a unit of measurement that is to be used as the default for any parameter that does not specify a given dimension. This default is in effect until another SETUNITS command is encountered. This example:

SETUNITS 1 IN 1 IN ;
.
.
.
POSITION (or OFFSET or LINEONE) 1 1 ;

shows part of a PPFA command stream in which a SETUNITS command sets the units of measurement to one inch for a subsequent POSITION (or OFFSET or LINEONE) subcommand.

SETUNITS can be used as a multiplier:
In this example, the SETUNITS command sets two-inch x and y default values. The POSITION subcommand values are multiplied by the default values creating a position four inches horizontally and four inches vertically from a given reference point. See “SETUNITS Command” on page 145 for a more detailed explanation.

---

**Notational Conventions**

These symbols are used in the command definitions:

- [ ] Brackets enclose optional subcommands or parameters
- Logical OR symbol separates mutually exclusive parameters: THIS ─ THAT
- {} Braces enclose mutually exclusive parameters: { THIS ─ THAT }
- Underscores (in boxed syntax) indicate defaults for commands and subcommands
- An ellipsis following a command or a set of commands indicates the command or set of commands can be repeated.
Chapter 8. Form Definition Command Reference

This section includes:

- Sequence of commands for form definitions
- Form definition commands listed alphabetically
- Detailed information on each command
- Descriptions of the applicable subcommands and parameters for each command

Sequence of Commands for Form Definitions

```
[SETUNITS ]
FORMDEF
[SUPPRESSION ...]
[COPYGROUP ]
[OVERLAY ...]
[SUBGROUP ...]
[COPYGROUP ]
[OVERLAY ...]
[SUBGROUP ...]...
```

1. SUPPRESSION commands must be specified immediately after FORMDEF commands. The exception is the SETUNITS command (see item 4).

2. OVERLAY and SUBGROUP commands must be specified under their associated COPYGROUP command. The OVERLAY commands must be specified immediately after a COPYGROUP command.
   - The OVERLAY command is required only to designate an overlay that is to be kept in the 3800 printer as raster data, or to specify a local name for referencing an overlay in a SUBGROUP command. If you do not code the OVERLAY command, you can still specify an overlay in a SUBGROUP command using its user-access name.
   - Overlays also may be specified using the N_UP subcommand of the FORMDEF or COPYGROUP command, or using the PRINTLINE command in the page definition. If the overlay is specified in one of these ways, it should also not be coded on the OVERLAY or SUBGROUP commands shown here. For more information, see “Medium Overlays and Page Overlays” on page 114.

   **Note:** If the form definition has only one copy group, the COPYGROUP command can be omitted. The OVERLAY command then follows any SUPPRESSION command.

3. The first COPYGROUP command can be omitted in a form definition if it contains only one copy group and no OVERLAY commands. If it is omitted, the FORMDEF command parameters are used to define the copy group.

4. A SETUNITS command can be placed before any PPFA command. The values set are in effect until the next SETUNITS command.

5. Each command can appear more than once under one FORMDEF command.
COPYGROUP Command

COPYGROUP name
[ADJUST n ]
[ BIN { n | MANUAL } ]
[ CONSTANT { NO | BACK | FRONT | BOTH } ]
[ DIRECTION { ACROSS | DOWN } ]
[ DUPLEX { NO | NORMAL | TUMBLE | RNORMAL | RTUMBLE } ]
[ INVOKE { SHEET | NEXT | FRONT | BACK } ]
[ JOG { YES | NO } ]
[ OFFSET x [ unit ] y [ unit ] ]
[ PRESENT { PORTRAIT | LANDSCAPE } ]
[ QUALITY { n } ]
[ PROCESSING ]
{ [ MEDIA_INFO { n } ... ]
[ PERFORATE ]
[ CUT ] } ]
[ N_UP { 1 | 2 | 3 | 4 } ]
[ [ OVERLAY name [ x-pos y-pos ] [ PARTITION ] ... ] ]
* [ PLACE { n [ FRONT | BACK ] } ]
[ CONSTANT ]
[ OFFSET x [ unit ] y [ unit ] ]
[ OVERLAY name [ x [ unit ] y [ unit ] ] [ PARTITION ] ]
[ ROTATION { 0 | 90 | 180 | 270 } ] ] ... ] ;

* The use of the PLACE subcommand indicates enhanced N_UP printing.

Copy groups are subsets of a form definition. A form definition can contain one or several copy groups. Copy groups are nested within a form definition following any SUPPRESSION command. COPYGROUP subcommands have no fixed defaults; if any subcommand is omitted, its value is selected from the corresponding subcommand in the FORMDEF command.

Notes:
1. Subsets of copy groups are called subgroups.
2. If you specified DUPLEX NO anywhere in the copy group, output will be simplex regardless of any other DUPLEX subcommand within the same copy group.
3. If a form definition has only one copy group, the COPYGROUP command can be omitted. If omitted, a name is automatically assigned by PPFA to the copy group, using the form definition resource name, including the F1 prefix. All values for the copy group are given the values from the FORMDEF command and subcommands. You need to know this name should you use conditional processing and need to invoke this copy group by name. Copy groups are placed within the form definition in the order in which they are generated.

To change copy groups during formatting, use conditional processing.
Note: Another way to change copy groups after the resource is stored is to insert an Invoke Medium Map structured field into your print data file (copy groups are known to PSF as medium maps). If no Invoke Medium Map structured field is found and no conditional processing is being performed, the first copy group in the form definition is used for the job.

**COPYGROUP name**
Defines an alphanumeric name of 1–8 characters. This name must be unique in a single form definition. If any names are duplicated, PPFA issues an error message and does not create the form definition.

### Subcommands

**[ ADJUST n ] (3800 Printers Only)**
Establishes the range of horizontal adjustment for the print area on the sheet.

- **n**
The adjustment range can be set from 0 to 20 L-units. After a value is set, it is the maximum amount available in both directions, plus and minus.

  **Note:** If you specify ADJUST, the maximum logical page size (in the horizontal direction) is reduced by the amount you specified here.

**[ BIN { n | MANUAL } ]**
Specifies the paper source. This subcommand should be used only for printers that have more than one paper source.

- **1**
  Selects the primary paper source.

- **2–255**
  Selects another paper source. If the specified bin does not exist on your printer, the default paper source for that printer will be used. For more information about paper sources on your printer, refer to your printer publications. Using a value of 100 is the same as specifying MANUAL.

**MANUAL**
Selects manual feed as a paper source on those printers that support manual feed. For more information, refer to your printer publications.

### Notes:

1. BIN selection is overridden by the printer if the form defined to each bin is the same form number. Only the primary bin is selected.

2. The primary source usually contains either letter-size (U.S.) or A4 (I.S.O.) paper. Other paper sources are used for less common paper sizes (such as legal-size) and for special paper (such as colored stock or preprinted letterhead on heavy bond).

3. If duplexing is requested and you select from the front side from one bin and the back side from another bin, a warning message will be issued and the printer will take the paper from the bin specified on the front side.
[ CONSTANT { NO | BACK | FRONT | BOTH } ]
Specifies whether the constant-forms function is on or off and whether constant form is to be printed on the front or back sides of a sheet.

NO Specifies that the constant forms function is off.
BACK Specifies that a constant form is to be printed on the back side without variable data.
FRONT Specifies that a constant form is to be printed on the front side without variable data.
BOTH Specifies that a constant form is to be printed on both sides without variable data.

[ DIRECTION { ACROSS | DOWN } ]
Determines, along with the PRESENT subcommand, how data is oriented on printers whose media origin can be changed. See the list of printers under the PRESENT subcommand.

If you are printing line data, you usually specify the same value for the DIRECTION subcommand as is specified for the DIRECTION subcommand in the page definition.

ACROSS Specifies that the pages are formatted in the ACROSS printing direction.
DOWN Specifies that the pages are formatted in the DOWN printing direction.

If the DIRECTION subcommand is specified, you must specify the PRESENT subcommand. The default for DIRECTION is determined by the value specified for PRESENT.

The direction default of PORTRAIT is ACROSS; the direction default of LANDSCAPE is DOWN. If neither PRESENT nor DIRECTION is specified, the default is PRESENT PORTRAIT and DIRECTION ACROSS.

[ DUPLEX { NO | NORMAL | TUMBLE | RNORMAL | RTUMBLE } ]
Specifies whether printing is done on both sides of the sheet. This subcommand should be used only for page printers that have duplex capability.

NO Duplex printing is not performed.
NORMAL Duplex printing is performed, with the tops of both sides printed along the same edge for side binding.
TUMBLE Duplex printing is performed with the top of one side and the bottom of the other printed along the same edge of the sheet for top binding.
RNORMAL Rotated normal. Duplex printing is performed with the tops of both sides printed along the same edge. Used with landscape pages, N_UP 2, and N_UP 3.
RTUMBLE Rotated tumble. Duplex printing is performed with the top of one side printed along the same edge of the sheet as the
bottom of the other. Used with landscape pages, N_UP 2, and N_UP 3.

[ INVOKE { SHEET | NEXT | FRONT | BACK } ]

Specifies where the next page of data is placed when this copy group is activated by conditional processing or by an Invoke Medium Map structured field.

INVOKE SHEET, which is the default, places the next page of data on a new sheet. The NEXT, FRONT, and BACK parameters place the next page in a subsequent partition on the same sheet or, if no partitions are available, on the next sheet. If FRONT or BACK is specified, INVOKE selects only partitions on the front or back, respectively.

PSF honors the NEXT, FRONT, and BACK values of the INVOKE subcommand only if the new copy group has the same medium modifications as the previous copy group. Medium modifications include duplexing, bin, page offset, N_UP values, presentation, direction, and medium overlays.

If any of these modifications differ, PSF ejects to a new sheet when the copy group is invoked. If you want to change overlays when ejecting to a new partition, use page overlays instead of medium overlays. See “Medium Overlays and Page Overlays” on page 114 for information about page and medium overlays.

When you use PLACE subcommands, the NEXT, FRONT, and BACK parameters place the next page using the next sequential PLACE subcommand that matches the requirement (next, front, or back). For example, if you print using the second PLACE subcommand of copy group A, and then you change to copy group B, you will start with the third PLACE subcommand of copy group B.

A CONSTANT parameter on the PLACE subcommand does not alter the selection process. The selection is complete, even though the selected PLACE subcommand does not place the data. N_UP performs the constant modification and continues until it finds a PLACE subcommand that does not specify CONSTANT. The data is placed with this subcommand. Note that this PLACE subcommand need not match the FRONT or BACK specifications of the INVOKE subcommand.

SHEET Specifies that data be placed in the first selected partition of the sheet.

NEXT Specifies that data be placed in the next selected partition.

FRONT Specifies that data be placed in the next selected front partition.

BACK Specifies that data be placed in the next selected back partition.

[ JOG { YES | NO } ]

Specifies whether output pages of this copy group are stacked offset from those of a previous copy group. Some cut-sheet page printers automatically offset stack the output of complete data files from each other within one print job.
YES  Specifies offset stacking between copy groups on cut-sheet page printers and on continuous-forms printers with a burster, trimmer, stacker feature. For all other continuous-form printers, this allows for printing marks on the sheets to indicate breaks between copy groups.

NO   Do not offset stack or mark sheets.

[ OFFSET x [ unit ] y [ unit ] [ x [ unit ] y [ unit ] ] ]

Specifies the offset of the logical page for both the front and back pages in reference to the media origin. The media origin is printer dependent. For more information about media origin, see your printer publications or Advanced Function Presentation: Printer Information.

If you specify offset values for the back of the page, you must also specify the front offset values.

Note: The OFFSET subcommand does not affect the position of medium overlays.

x  Specifies the horizontal offset of the logical page on the front or back side of the copy group relative to the media origin. The offset values may vary according to the type of printer. For more information, refer to your printer publications.

The x is a decimal number with up to three decimal places. It must be specified if unit is specified.

The default for the first x parameter is 1; no default value is set for the second x parameter.

unit  Specifies a unit of measurement for the x parameter. The choices are: IN, MM, CM, POINTS, or PELS.

If no unit is specified, a default setting is:

- Taken from the last SETUNITS command
- IN (inch) if no SETUNITS command has been issued

y  Specifies the vertical offset for the logical page for the front or back side of the page. The offset values may vary according to the type of printer.

The y is a decimal number with up to three decimal places. It must be specified if unit is specified.

The default for the first y parameter is 1; no default value is set for the second y parameter.

Note: The vertical offset for the 3800 must be 0.5 inch or greater.

For more information, refer to your printer publications.

unit  Specifies a unit of measurement for the y parameter. The choices are: IN, MM, CM, POINTS, or PELS.

If no unit is specified, a default setting is:
• Taken from the last SETUNITS command
• \textit{IN} (inch) if no SETUNITS command has been issued

\textbf{Note:} If OFFSET is not specified, the OFFSET default is 0.1 IN 0.1 IN.

\begin{verbatim}
[ PRESENT { PORTRAIT | LANDSCAPE } ]
\end{verbatim}

Specifies, along with the DIRECTION subcommand, how the data will be oriented on printers whose media origin can be changed. PRESENT and DIRECTION are supported on the following continuous-forms printers:

- IBM 3835 Page Printer Model 001
- IBM 3835 Advanced Function Printer Model 002
- IBM 3900 Advanced Function Printer Model 001
- IBM 3900 Advanced Function Duplex Printing System
- IBM 3900 Wide Advanced Function Printer Model 0W1

The PRESENT and DIRECTION subcommands are also supported by N_UP cut-sheet printers when you specify the N_UP subcommand. You must specify N_UP in the COPYGROUP or FORMDEF command to activate the PRESENT and DIRECTION subcommands on cut-sheet printers. See Figure 47 on page 98 through Figure 50 on page 101 to determine the effect of the PRESENT and DIRECTION subcommands when you use them with the N_UP subcommand.

\textbf{PORTRAIT}

Specifies that the pages are printed in the portrait page presentation, with their short edges at the top and bottom and their long edges at the sides.

\textbf{LANDSCAPE}

Specifies that the pages are printed in the landscape page presentation, with their long edges at the top and bottom and their short edges at the sides.

\begin{verbatim}
[ QUALITY { 1 | 2 | 3 | 4 | ... | 10 } ]
\end{verbatim}

Specifies the print quality. This subcommand is recognized only on printers that can produce more than one level of print quality. The default is determined by the printer model. (On some printers, the default may be set at the printer itself.) For more information, refer to your printer publications.

\textbf{1–10}

You can select a level of print quality by entering any whole number from 1 to 10. Higher numbers correspond to higher levels of print quality; lower numbers correspond to lower levels. For more information, refer to your printer publications.

Print quality is determined by a numerical code in the range of 1 to 254 (hexadecimal X'01'–X'FE'). The codes corresponding to the possible QUALITY parameters are:

\begin{itemize}
  \item 1 = 15 (X'0F')
  \item 2 = 40 (X'28')
  \item 3 = 65 (X'41')
  \item 4 = 90 (X'5A')
\end{itemize}
[ PROCESSING { [ MEDIA_INFO [ [ 0...254 ] ] | 255 ] [ PERFORATE ] [ CUT ]

Specifies additional postprocessing capabilities for selected printers and attached equipment. This subcommand expects 1 to 3 of the following keywords:

{ [ MEDIA_INFO [ [ 0...254 ] ] | 255 ]

  This parameter specifies the ID of fixed medium information that a printer or printer–attached device applies to a sheet. Examples such as color plates logos, letter heads, and other fixed images.

  The numeric values that can be included are:

  0–254

  These numeric values select a particular fixed medium local ID that the printer or printer–attached device applies to a sheet. One or more IDs can be specified within this range.

  255

  This value selects all the current fixed medium local IDs that the printer or printer–attached devices applies to a sheet.

[ PERFORATE ]

  Specifies a perforation cut at one or more fixed locations on the sheet according to the printer or printer–attached device.

[ CUT ]

  Specifies a separation cut at one or more fixed locations on the sheet according to the printer or printer–attached device.

[ N_UP { 1 | 2 | 3 | 4 } ]

  Specifies the number (1, 2, 3, or 4) of equal-size partitions into which the sheet will be divided. See page 97 for a list of printers that support the N_UP subcommand.

  If you do not specify the N_UP subcommand in the COPYGROUP command, the N_UP subcommand from the FORMDEF command is the default for the COPYGROUP command. You can mix N_UP printing and non-N_UP printing by specifying or not specifying the N_UP subcommand in each copy group and by not specifying N_UP in the FORMDEF command.

[ { OVERLAY } name [ x-pos y-pos ] [ PARTITION ] ]

  Specifies the name of an overlay to be placed with every page in each of the N_UP partitions. The overlay is placed relative to the page origin, or if the PARTITION parameter is specified, relative to the partition origin. You can specify a maximum of 127 OVERLAY subcommands in a copy group.

  [x-pos y-pos ]

  Specifies the horizontal and vertical adjustment to the position of the overlay. This is in addition to any offset...
values built into the overlay. The x and y values must be positive (+). You can specify them in inches (IN), millimeters (MM), centimeters (CM), POINTS, or PELS. If you do not specify a unit value, PPFA uses the unit value specified in the last SETUNITS command or uses a default unit value of inches.

Note: This OVERLAY subcommand cannot be specified if the PLACE subcommand is specified.

PLACE { n [ FRONT | BACK ] }
Places a page of data or a constant modification relative to a partition. Each PLACE subcommand specifies the number n of a partition on either the front or back side of the sheet. FRONT is the default, if you do not specify this subcommand. You must specify the same number of PLACE subcommands as the number of partitions on the sheet. The sequence of the PLACE subcommands is the sequence in which incoming pages will be placed in the partitions.

Note: The PLACE subcommand is valid only on printers that support enhanced N_UP printing. If PLACE is not specified, pages are placed in partitions in the default partition sequence.

n Specifies the numbered partition (1–4) into which the page of data is placed. See Figure 47 on page 98 through Figure 50 on page 101 for the locale of each numbered partition.

FRONT Specifies that this partition be placed on the front side of the sheet.

BACK Specifies that this partition be placed on the back side of the sheet.

CONSTANT Specifies that no page data will be placed by this PLACE subcommand.

Use CONSTANT when you are placing overlays without user's data or are placing fewer data pages on the sheet than the number of partitions specified in the N_UP subcommand.

For an example of using the CONSTANT parameter with overlays and to understand how the ordering of the PLACE subcommand affects overlays, see “Enhanced N_UP Example 3: Asymmetric Pages” on page 111.

OFFSET x [ unit ] y [ unit ]
Specifies a positive offset of the page horizontally (x) and vertically (y) from the partition origin. If OFFSET is not coded, PPFA uses the value of 0.1 inch for both x and y offsets. This OFFSET parameter overrides any other OFFSET parameters specified on the FORMDEF or COPYGROUP command. You can specify the units in inches (in), millimeters (mm), centimeters (cm), points, or
pels. If you do not specify a unit value, PPFA uses the unit value specified in the last SETUNITS command or uses a default unit value of inches.

\[ \text{OVERLAY name } [ x [ \text{unit} ] y [ \text{unit} ] ] ] \]

Specifies the name of an overlay to be placed with this PLACE subcommand. The overlay is placed relative to the page origin or, if the PARTITION keyword is specified, to the partition origin. You can specify multiple OVERLAY parameters in each PLACE subcommand.

\[ x [\text{unit}] y [\text{unit}] \]

Specifies the horizontal and vertical adjustment to the position of the overlay. This is in addition to any offset values built into the overlay. The \(x\) and \(y\) values must be positive (+). You can specify them in inches (in), millimeters (mm), centimeters (cm), points, or pels. If you do not specify a unit value, PPFA uses the unit value specified in the last SETUNITS command or uses a default value of inches.

\[ \text{PARTITION} \]

Specifies that the previous offset is from the partition origin. If not present, the offset is from the page origin, which is subject to the OFFSET parameter.

\[ \text{ROTATION} \{ 0 | 90 | 180 | 270 \} \]

Specifies the clockwise rotation of the page and associated page overlays placed by this PLACE command.

Rotation turns the page and its associated page overlays around their fixed origin points. If you rotate the page without moving its origin point, you might rotate it off the physical medium. To prevent this, always offset the page origin to the place you want it to be for the rotated page, as shown in Figure 70.

![Figure 70. Offsetting the Page Origin for Rotated Pages](Image)
A form definition is a resource that contains all the controls relating to the physical sheet. A FORMDEF command must be specified when you define a new form definition. When subcommands (except for the REPLACE, PRESENT, and DIRECTION subcommands) are specified, they become the defaults for all COPYGROUP commands nested within this form definition.

**FORMDEF name**

Defines an alphanumeric name of 1–8 characters for the form definition. When you create a form definition, PPFA assigns a prefix of F1 to the name you specify. F1nnnnnn is the resource name in the form-definition library.

### Subcommands

**[ ADJUST n ] (3800 Printers Only)**

Establishes the range of horizontal adjustment for the print area on the sheet.

\[ n \]

The adjustment range can be set from 0 to 20 L-units. After a value is set, it is the maximum amount available in both directions, plus and minus.
Note: If you specify ADJUST, the maximum logical page size (in the horizontal direction) is reduced by the amount you specified here.

[ BIN { n | MANUAL } ]

Specifies the paper source. This subcommand should be used only for printers that have more than one paper source.

1  Selects the primary paper source.

2–255  Selects another paper source. If the specified bin does not exist on your printer, the default paper source for that printer will be used. For more information about paper sources on your printer, refer to your printer publications. Using a value of 100 is the same as specifying MANUAL.

MANUAL

Selects manual feed as a paper source on those printers that support manual feed. For more information, refer to your printer publications.

Notes:

1. BIN selection is overridden by the printer if the form defined to each bin is the same form number. Only the primary bin is selected.

2. The primary source usually contains either letter-size (U.S.) or A4 (I.S.O.) paper. Other paper sources are used for less common paper sizes (such as legal-size) and for special paper (such as colored stock or preprinted letterhead on heavy bond).

3. If duplexing is requested and you select from the front side from one bind and the back side from another bin, a warning message will be issued and the printer will take the paper from the bin specified on the front side.

[ CONSTANT { NO | BACK | FRONT | BOTH } ]

Specifies whether the constant-forms function is on or off and whether constant form is to be printed on the front or back sides of a sheet.

NO  Specifies that the constant forms function is off.

BACK  Specifies that a constant form is to be printed on the back side without variable data.

FRONT  Specifies that a constant form is to be printed on the front side without variable data.

BOTH  Specifies that a constant form is to be printed on both sides without variable data.

[ DIRECTION { ACROSS | DOWN } ]

Determines, along with the PRESENT subcommand, how data is oriented on printers whose media origin can be changed. See the list of printers under the PRESENT subcommand.

If you are printing line data, you usually specify the same value for the DIRECTION subcommand as is specified for the DIRECTION subcommand in the page definition.
ACROSS  Specifies that the pages are formatted in the ACROSS printing direction.

DOWN   Specifies that the pages are formatted in the DOWN printing direction.

If the DIRECTION subcommand is specified, you must specify the PRESENT subcommand. The default for DIRECTION is determined by the value specified for PRESENT.

The direction default of PORTRAIT is ACROSS; the direction default of LANDSCAPE is DOWN. If neither PRESENT nor DIRECTION is specified, the default is PRESENT PORTRAIT and DIRECTION ACROSS.

[ DUPLEX { NO | NORMAL | TUMBLE | RNORMAL | RTUMBLE } ]

Specifies whether printing is done on both sides of the sheet. This subcommand should be used only for page printers that have duplex capability.

NO      Duplex printing is not performed.

NORMAL  Duplex printing is performed, with the tops of both sides printed along the same edge for side binding.

TUMBLE  Duplex printing is performed with the top of one side and the bottom of the other printed along the same edge of the sheet for top binding.

RNORMAL  Rotated normal. Duplex printing is performed with the tops of both sides printed along the same edge. Used with landscape pages, N_UP 2, and N_UP 3.

RTUMBLE  Rotated tumble. Duplex printing is performed with the top of one side printed along the same edge of the sheet as the bottom of the other. Used with landscape pages, N_UP 2, and N_UP 3.

[ INVOKE { SHEET | NEXT | FRONT | BACK } ]

Specifies where the next page of data is placed when this copy group is activated by conditional processing or by an Invoke Medium Map structured field.

INVOKE SHEET, which is the default, places the next page of data on a new sheet. The NEXT, FRONT, and BACK parameters place the next page in a subsequent partition on the same sheet or, if no partitions are available, on the next sheet. If FRONT or BACK is specified, INVOKE selects only partitions on the front or back, respectively.

PSF honors the NEXT, FRONT, and BACK values of the INVOKE subcommand only if the new copy group has the same medium modifications as the previous copy group. Medium modifications include duplexing, bin, page offset, N_UP values, presentation, direction, and medium overlays.
If any of these modifications differ, PSF ejects to a new sheet when the copy group is invoked. If you want to change overlays when ejecting to a new partition, use page overlays instead of medium overlays. See “Medium Overlays and Page Overlays” on page 114 for information about page and medium overlays.

When you use PLACE subcommands, the NEXT, FRONT, and BACK parameters place the next page using the next sequential PLACE subcommand that matches the requirement (next, front, or back). For example, if you print using the second PLACE subcommand of copy group A, and then you change to copy group B, you will start with the third PLACE subcommand of copy group B.

A CONSTANT parameter on the PLACE subcommand does not alter the selection process. The selection is complete, even though the selected PLACE subcommand does not place the data. N_UP performs the constant modification and continues until it finds a PLACE subcommand that does not specify CONSTANT. The data is placed with this subcommand. Note that this PLACE subcommand need not match the FRONT or BACK specifications of the INVOKE subcommand.

**SHEET**

Specifies that data be placed in the first selected partition of the sheet.

**NEXT**

Specifies that data be placed in the next selected partition.

**FRONT**

Specifies that data be placed in the next selected front partition.

**BACK**

Specifies that data be placed in the next selected back partition.

**[JOG { YES | NO }]**

Specifies whether output pages of this copy group are stacked offset from those of a previous copy group. Some cut-sheet page printers automatically offset stack the output of complete data files from each other within one print job.

**YES**

Specifies offset stacking between copy groups on cut-sheet page printers and on continuous-forms printers with a burster, trimmer, stacker feature. For all other continuous-form printers, this allows for printing marks on the sheets to indicate breaks between copy groups.

**NO**

Do not offset stack or mark sheets.

**[OFFSET x [ unit ] y [ unit ] [ x [ unit ] y [ unit ] ]]**

Specifies the offset of the logical page for both the front and back pages in reference to the media origin. The media origin is printer dependent. For more information about media origin, see your printer publications or Advanced Function Presentation: Printer Information.

If you specify offset values for the back of the page, you must also specify the front offset values.

**Note:** The OFFSET subcommand does not affect the position of medium overlays.
\textit{x} Specifies the horizontal offset of the logical page on the front or back side of the copy group relative to the media origin. The offset values may vary according to the type of printer. For more information, refer to your printer publications.

The \textit{x} is a decimal number with up to three decimal places. The \textit{x} must be specified if \textit{unit} is specified.

The default for the first \textit{x} parameter is 1; no default value is set for the second \textit{x} parameter.

\textit{unit} Specifies a unit of measurement for the \textit{x} parameter. The choices are: \textsc{in}, \textsc{mm}, \textsc{cm}, \textsc{points}, or \textsc{pel}. If no unit is specified, a default setting is:

- \textit{in} if no SETUNITS command has been issued

\textit{y} Specifies the vertical offset for the logical page for the front or back side of the page. The offset values may vary according to the type of printer.

The \textit{y} is a decimal number with up to three decimal places. It must be specified if \textit{unit} is specified.

The default for the first \textit{y} parameter is 1; no default value is set for the second \textit{y} parameter.

\textbf{Note:} The vertical offset for the 3800 must be 0.5 inch or greater.

For more information, refer to your printer publications.

\textit{unit} Specifies a unit of measurement for the \textit{y} parameter. The choices are: \textsc{in}, \textsc{mm}, \textsc{cm}, \textsc{points}, or \textsc{pel}. If no unit is specified, a default setting is:

- \textit{in} if no SETUNITS command has been issued

\textbf{Note:} If OFFSET is not specified, the OFFSET default is \textbf{0.1 IN 0.1 IN}.

\textbf{[ PRESENT \{ PORTRAIT | LANDSCAPE \} ]}

Specifies, along with the \textsc{direction} subcommand, how the data will be oriented on printers whose media origin can be changed. \textsc{present} and \textsc{direction} are supported on the following continuous-forms printers:

- IBM 3835 Page Printer Model 001
- IBM 3835 Advanced Function Printer Model 002
- IBM 3900 Advanced Function Printer Model 001
- IBM 3900 Advanced Function Duplex Printing System
- IBM 3900 Wide Advanced Function Printer Model 0W1
The PRESENT and DIRECTION subcommands are also supported by N_UP cut-sheet printers when you specify the N_UP subcommand. You must specify N_UP in the COPYGROUP or FORMDEF command to activate the PRESENT and DIRECTION subcommands on cut-sheet printers. See Figure 47 on page 98 through Figure 50 on page 101 to determine the effect of the PRESENT and DIRECTION subcommands when you use them with the N_UP subcommand.

**PORTRAIT**

Specifies that the pages are printed in the portrait page presentation, with their short edges at the top and bottom and their long edges at the sides.

**LANDSCAPE**

Specifies that the pages are printed in the landscape page presentation, with their long edges at the top and bottom and their short edges at the sides.

**QUALITY { 1 | 2 | 3 | 4 | ... | 10 }**

Specifies the print quality. This subcommand is recognized only on printers that can produce more than one level of print quality. The default is determined by the printer model. (On some printers, the default may be set at the printer itself.) For more information, refer to your printer publications.

1–10 You can select a level of print quality by entering any whole number from 1 to 10. Higher numbers correspond to higher levels of print quality; lower numbers correspond to lower levels. For more information, refer to your printer publications.

Print quality is determined by a numerical code in the range of 1 to 254 (hexadecimal X'01'–X'FE'). The codes corresponding to the possible QUALITY parameters are:

1 = 15 (X'0F')  
2 = 40 (X'28')  
3 = 65 (X'41')  
4 = 90 (X'5A')  
5 = 115 (X'73')  
6 = 140 (X'8C')  
7 = 165 (X'A5')  
8 = 190 (X'BE')  
9 = 215 (X'D7')  
10 = 240 (X'F0')

**REPLACE { YES | NO }**

Specifies whether this form definition is to replace an existing one with the same resource name in the library.

**YES** Replace an existing form definition of the same name in the library if there is one. If a form definition with the same name does not exist in the library, then store this form definition.

**NO** Do not replace an existing form definition of the same name. If a form definition with the same name does not exist in the library, then store this form definition.
This is the default.

[ PROCESSING
  { [ MEDIA_INFO
    [ [ 0...254 ] ] | 255 ]
  ][ PERFORATE ]
[ CUT ]

Specifies additional postprocessing capabilities for selected printers and attached equipment. This subcommand expects 1 to 3 of the following keywords:

{ [ MEDIA_INFO
    [ [ 0...254 ] ] | 255 ]

This parameter specifies the ID of fixed medium information that a printer or printer–attached device applies to a sheet. Examples such as color plates logos, letter heads, and other fixed images.

The numeric values that can be included are:

0–254
These numeric values select a particular fixed medium local ID that the printer or printer–attached device applies to a sheet. One or more IDs can be specified within this range.

255
This value selects all the current fixed medium local IDs that the printer or printer–attached devices applies to a sheet.

[ PERFORATE ]
Specifies a perforation cut at one or more fixed locations on the sheet according to the printer or printer–attached device.

[ CUT ]
Specifies a separation cut at one or more fixed locations on the sheet according to the printer or printer–attached device.

[ N_UP { 1 | 2 | 3 | 4 } ]
Specifies the number (1, 2, 3, or 4) of equal-size partitions into which the sheet will be divided. See page 97 for a list of printers that support the N_UP subcommand.

If you do not specify the N_UP subcommand in the COPYGROUP command, the N_UP subcommand from the FORMDEF command is the default for the COPYGROUP command. You can mix N_UP printing and non-N_UP printing by specifying or not specifying the N_UP subcommand in each copy group and by not specifying N_UP in the FORMDEF command.

[ { OVERLAY
    name [ x-pos y-pos ]
  [ PARTITION ] ]

Specifies the name of an overlay to be placed with every page in each of the N_UP partitions. The overlay is placed relative to the page origin, or if the PARTITION parameter is specified, relative to the partition origin. You can specify a maximum of 254 OVERLAY subcommands in a copy group.

[x-pos y-pos ]
Specifies the horizontal and vertical adjustment to the position of the overlay. This is in addition to any offset values built into the overlay. The x and y values must be positive (+). You can specify them in inches (in), millimeters (mm), centimeters (cm), points, or pels. If you do not specify a unit value, PPFA uses the unit value specified in Chapter 8. Form Definition Command Reference
the last SETUNITS command or uses a default unit value of inches.

Note: This OVERLAY subcommand cannot be specified if the PLACE subcommand is specified. Use the OVERLAY parameter of the PLACE subcommand instead.

[ PLACE { n [ FRONT | BACK] } ]
Places a page of data or a constant modification relative to a partition. Each PLACE subcommand specifies the number \( n \) of a partition on either the front or back side of the sheet. FRONT is the default, if you do not specify this subcommand. You must specify the same number of PLACE subcommands as the number of partitions on the sheet. The sequence of the PLACE subcommands is the sequence in which incoming pages will be placed in the partitions.

Note: The PLACE subcommand is valid only on printers that support enhanced N_UP printing. If PLACE is not specified, pages are placed in partitions in the default partition sequence.

\( n \) Specifies the numbered partition (1–4) into which the page of data will be placed.

FRONT Specifies that this partition be placed on the front side of the sheet.

BACK Specifies that this partition be placed on the back side of the sheet.

[ CONSTANT ]
Specifies that no page data will be placed by this PLACE subcommand.

Use CONSTANT when you are placing overlays without user's data or are placing fewer data pages on the sheet than the number of partitions specified in the N_UP subcommand.

For an example of using the CONSTANT parameter with overlays and to understand how the ordering of the PLACE subcommand affects overlays, see “Enhanced N_UP Example 3: Asymmetric Pages” on page 111.

[ OFFSET x [ unit ] y [ unit ] ]
Specifies a positive offset of the page horizontally (x) and vertically (y) from the partition origin. If OFFSET is not coded, PPFA uses the value of 0.1 inch for both the x and y offsets. This OFFSET parameter overrides any other OFFSET parameters specified on the FORMDEF or COPYGROUP command. You can specify the units in inches (in), millimeters (mm), centimeters (cm), points, or pels. If you do not specify a unit value, PPFA uses the
unit value specified in the last SETUNITS command or uses a default unit value of inches.

[ OVERLAY name [ x [ unit ] y [ unit ] ]
  [ PARTITION ] ]
Specifies the name of an overlay to be placed with this PLACE subcommand. The overlay is placed relative to the page origin or, if the PARTITION keyword is specified, to the partition origin. You can specify multiple OVERLAY parameters in each PLACE subcommand.

x [ unit ] y [ unit ]
Specifies the horizontal and vertical adjustment to the position of the overlay. This is in addition to any offset values built into the overlay. The x and y values must be positive (+). You can specify them in inches (in), millimeters (mm), centimeters (cm), points, or pels. If you do not specify a unit value, PPFA uses the unit value specified in the last SETUNITS command or uses a default value of inches.

[PARTITION]
Specifies that the previous offset is from the partition origin. If not present, the offset is from the page origin, which is subject to the OFFSET parameter.

[ ROTATION {0 | 90 | 180 | 270 } ] ]
Specifies the clockwise rotation of the page and associated page overlays placed by this PLACE command.

Rotation turns the page and its associated page overlays around their fixed origin points. If you rotate the page without moving its origin point, you might rotate it off the physical medium. To prevent this, always offset the page origin to the place you want it to be for the rotated page, as shown in Figure 71 on page 144.
OVERLAY Command

OVERLAY [ name1 ] name2  
[ RASTER | NORASTER ] ;

This OVERLAY command identifies an electronic overlay to be used in one or more subgroups of a copy group. When using the OVERLAY command, follow these guidelines:

[ VFYSETUP verificationID ... ]

Use specifically for the IBM InfoPrint 4000 Highlight Color post processor to propagate the setup IDs to all medium maps (copygroups) in the form definition. Do not specify VFYSETUP on the COPYGROUP command. Before using the VFYSETUP subcommand, verify that your version of PSF supports FORMDEF setup verification.

To use VFYSETUP, specify one or more 4-character identifier sets that match the Setup Verification IDs defined at the printer operator's console for the specific print job. For example, if the Setup Verification IDs defined at the printer were X'012F', X'0521', and X'938A', specify the following:

FORMDEF vfy7 REPLACE YES VFYSETUP 012F 0521 938A;

When PSF processes the print job, it compares the setup verification IDs in the form definition to the IDS that are active in the printer. If the active IDs in the printer do not match the IDs required by the form definition, or if the printer does not support FORMDEF setup verification IDs, the job is held.

Figure 71. Offsetting the Page Origin for Rotated Pages

1 Inch

Page offset 1 in., 1 in.

Page

1 Inch

Page offset 9 in., 1 in.

Page origin
Partition origin

90°
• An OVERLAY command comes after the COPYGROUP command.
• A separate OVERLAY command must be specified for each electronic overlay used in a subgroup.
• A maximum of 127 OVERLAY commands can be specified for coded overlays per copy group.
• The overlay named here must be referenced in a SUBGROUP command in order to be printed (see page 149).

Note: Overlays contain their own positioning controls.

OVERLAY [ name1 ] name2

Identifies an electronic overlay to be used in one or more subgroups of a copy group.

name1 Specifies an alphanumeric name of 1 to 16 characters (local name) for the overlay. It must conform to the token rules and must be unique within a copy group.

Note: If name1 is omitted, name2 is used as the local name and is the name used in the subgroup command.

name2 Specifies an alphanumeric name of 1 to 6 characters (user-access name) for this overlay. A prefix of O1 is added by PPFA to identify the overlay resource.

Subcommand

[ RASTER | NORASTER ] (3800 printers only)

Specifies overlays as raster or not raster data.

RASTER Specifies this overlay is to be kept in the printer as raster data. If this overlay is to be used several times, the printer does not need to recompile it each time.

Note: This function is ignored by PSF for AIX. One raster overlay can be specified per copy group.

NORASTER Specifies this is a coded overlay. A maximum of 127 coded overlays can be specified per copy group.

SETUNITS Command

] SETUNITS { x [ unit ] y [ unit ] [LINESP n unit ] [ LINESP n unit ] } ;

The SETUNITS command specifies the value and the unit of measurement that will be the default for any subsequent measurement parameter in all of the commands and subcommands. These values remain the default values until another SETUNITS command is specified. The SETUNITS command
should be specified as the first command in a form definition. If neither this command nor a measurement parameter is specified, the defaults identified within the following description are used.

**SETUNITS** \( x \{ \text{unit} \} \ y \{ \text{unit} \} \)

Specifies the value and the unit of measurement that will be the default for any subsequent measurement parameter in all of the commands and subcommands.

- **x** Specifies the number used for horizontal measurement. A number with up to three decimal places may be used. The default is 1.
  - **Note:** This value affects subsequent OFFSET subcommands.
- **unit** Specifies a unit of measurement for the \( x \) parameter. The choices are IN, MM, CM, POINTS, PELS, or CPI. The default is IN.
- **y** Specifies the number used for vertical measurement. A number with up to three decimal places may be used. The default is 1.
  - **Note:** This value affects subsequent OFFSET subcommands.
- **unit** Specifies a unit of measurement for the \( y \) parameter. The choices are IN, MM, CM, POINTS, PELS, or LPI. The default is IN.

**Using CPI and LPI Units of Measurement**

The CPI and LPI units of measurement make it possible to write the following command:

```
SETUNITS 10 CPI 6 LPI ;
```

This command sets the units of measurement for horizontal and vertical spacing in terms of characters per inch and lines per inch. You can then use the OFFSET subcommand specifications to increment the spacing one character or one line at a time. The distance specified by \( n \) characters over and by \( n \) lines down is defined in the governing SETUNITS command. In this example, there are 10 characters per inch (CPI) and 6 lines per inch (LPI).

**Subcommand**

**LINESP** \( n \text{unit} \)

This subcommand is to be used within a page definition; it serves no purpose when used within a form definition.
The SUBGROUP command specifies the number of copies of a single page that are to be printed and any modifications (consisting of overlays, suppressions, type of duplexing, and forms flash) to be made to the copies. A SUBGROUP command follows a COPYGROUP command; a maximum of 127 SUBGROUP commands can be specified within each copy group.

Notes:

1. The BOTH subcommand causes two subgroups to be generated. Thus, a maximum of 63 subgroups can be specified when the BOTH subcommand is used.

2. When you specify the DUPLEX subcommand (with a parameter other than NO) in the COPYGROUP command, you must include one SUBGROUP command for each side of a sheet, or you may specify the BOTH subcommand in a single SUBGROUP command.

Subcommands

[ COPIES n ]

 Specifies how many copies of each page are to be printed.

 n Defines the number of copies (the maximum number is 255). When BACK is specified within a SUBGROUP command, the system counts the front pages printed (the actual number of sheets) not copies made (front and back). The default is 1.

[ FLASH { NO | YES } ] (3800 printers only)

 Specifies whether to use forms flash.

 Note: When forms flash is used, its name must be specified in the job control language for the print job. The operator must place the correct negative into the 3800 when the job is ready to print.

 NO Specifies that forms flash does not occur.

 YES Specifies that forms flash occurs for the number of copies designated.

[ { BACK | FRONT | BOTH } ]

 These optional subcommands specify whether the subgroup is for both sides of a sheet or for only the front or the back side.

 Rules:
1. Subgroups must specify FRONT and BACK if an overlay, suppression, or forms flash appears on one side but not on the other.

2. The FRONT and BACK subgroups must have the same number of copies.
   If the number of copies differs, the COPIES parameter of the BACK subgroup is ignored, and a warning message is issued.

3. The FRONT and BACK subcommands must occur in pairs.

4. If the FRONT and BACK subcommands are specified with DUPLEX NO (in the FORMDEF or COPYGROUP commands), PPFA issues an error message and does not create the form definition.

**BACK**

Specifies this SUBGROUP command is for the back sides of the sheets.

A subgroup with a BACK subcommand must have a FRONT subcommand in the preceding subgroup.

**FRONT**

Specifies this subgroup is for the front sides of the sheets.

If a DUPLEX subcommand in a FORMDEF or COPYGROUP command is specified with a parameter other than NO and the FRONT subcommand is specified in a SUBGROUP command, the next SUBGROUP command must have a BACK subcommand.

**BOTH**

Specifies this subgroup is used for both sides of the sheet.

This is the default when DUPLEX is specified in the copy group.

If BOTH is specified with DUPLEX NO (in a FORMDEF or COPYGROUP command), PPFA issues a warning message and ignores the BOTH subcommand.

**[ BIN name { n | MANUAL } ]**

Specifies the paper source. This subcommand should be used only for printers that have more than one paper source.

1 Selects the primary paper source.

2–255 Selects another paper source. If the specified bin does not exist on your printer, the default paper source for that printer will be used. For more information about paper sources on your printer, refer to your printer documentation. Using a value of 100 is the same as specifying MANUAL.
MANUAL
Selects manual feed as a paper source on those printers that support manual feed. For more information, refer to your printer documentation.

Notes:
1. BIN selection is overridden by the printer if the form defined to each bin is the same form number. Only the primary bin is selected.
2. The primary source usually contains either letter-size (U.S.) or A4 (I.S.O.) paper. Other paper sources are used for less common paper sizes (such as legal-size) and for special paper (such as colored stock or preprinted letterhead on heavy bond).
3. If duplexing is requested and you select from the front side from one bin and the back side from another bin, a warning message will be issued and the printer will take the paper from the bin specified on the front side.
4. Bin changes at the SUBGROUP level are not supported on all printers. Check your printer documentation to see if your printer supports “media source by copy.”

[ OVERLAY name [ name ] ... ]
Specifies the electronic overlay that is to be used with this subgroup.

name Specifies either the local or user-access name. A maximum of eight names can be specified within a subgroup.

Notes:
1. If the local name is used, it must be defined in an OVERLAY command before it can be referenced.
2. PPFA does not check for duplicate user-access names.

[ SUPPRESSION name [ name ] ... ]
Specifies that the named field is suppressed.

name Specifies a alphanumeric name of 1 to 8 characters (local name) of the field to be suppressed. A maximum of eight names can be specified within a subgroup.

The suppression field named here must be defined in a SUPPRESSION command following the FORMDEF command before it can be referenced. See page 150.
SUPPRESSION Command

SUPPRESSION name ;

A SUPPRESSION command, if used, must immediately follow the FORMDEF command. It names the suppression that is specified in the FIELD command of a page definition associating the form definition and the page definition.

SUPPRESSION name

Identifies an alphanumeric name of 1 to 8 characters (local name). The name must conform to the token rules.

You must specify the area to be suppressed in a FIELD command or a SUBGROUP command using one of the names specified within this series of SUPPRESSION commands for the suppression to be effective.

Note: A maximum of eight suppressions can be specified for one SUBGROUP command, and a maximum of 127 suppressions can be specified within one form definition.
Chapter 9. Page Definition Command Reference

This section includes:

- Sequence of commands for page definitions
- Page definition commands listed alphabetically
- Detailed information on each command
- Descriptions of the applicable subcommands and parameters for each command

Sequence of Commands for Page Definitions

PAGEDEF
[ FONT...]
[ PAGEFORMAT]
 [ TRCREF...]
 [ SEGMENT...]
 [ OVERLAY...]
 PRINTLINE [ FIELD | CONDITION ...]
 [ ENDSUBPAGE] |
 [ PRINTLINE [ FIELD | CONDITION ...] ...]
[ PAGEFORMAT ]
 [ TRCREF...]
 [ SEGMENT...]
 [ OVERLAY...]
 PRINTLINE [ FIELD | CONDITION ...]
 [ ENDSUBPAGE] |
 [ PRINTLINE [ FIELD | CONDITION ...] ...] ...

- FONT commands must be specified immediately after a PAGEDEF command. The exception is the SETUNITS command.
- A SETUNITS command can be placed before any other PPFA command. The values set are in effect until the next SETUNITS command.
- TRCREF, SEGMENT, and OVERLAY commands must be specified under their associated PAGEFORMAT command.
- The first PAGEFORMAT command can be omitted in a page definition, if the page definition contains only one page format. If the PAGEFORMAT command is omitted, the PAGEDEF command parameters are used to define the page format.
- At least one PRINTLINE command is required.
- An ENDSUBPAGE command can occur anywhere in a page definition that a PRINTLINE command can occur, except it can not occur between a PRINTLINE command and its associated FIELD and CONDITION commands.
The CONDITION command examines data in an input record and specifies actions to be taken based on the result of the examination.

- The `condname` parameter must come before any subcommands
- No WHEN subcommand can follow an OTHERWISE subcommand in the same CONDITION command

**CONDITION**

Names the condition. The name must contain 1 to 8 alphanumeric characters.

PPFA allows cross-referencing to the `condname`. The cross-reference is done by using the short form of the CONDITION command (second format in the syntax table). By specifying a previously defined `condname`, PPFA uses the specifications from that command. When the condition is reused, the point where you want the comparison to begin may be at a different point in the record. By specifying the optional START subcommand, you can change the starting point of the comparison but not the field length. If the START subcommand is not specified, the starting point is the same as defined in the original CONDITION command.
Subcommands

START $n$

Specifies the starting byte of the comparison field within the data record where the comparison is to be done.

$n$

Specifies the number of bytes from the first data byte in the record as the starting point of the comparison field. The first data byte position of an input record is 1.

Note: The carriage-control character and the table-reference character are not considered data.

LENGTH $n$

Specifies the length of the comparison field.

$n$

Specifies the number of bytes in the data record to be compared, beginning with the position specified in START. Valid values are numbers from 1 to 8000. The length of the constant text must be the same as defined in this parameter or the results are invalid.

Comparisons are done on a byte-by-byte basis. Because the comparison field and the constant text must have the same lengths, padding is not necessary.

Note: If any part of the comparison field specified by the combination of START and LENGTH is outside the boundaries of the data record, all conditional processing is not performed. No WHEN is executed. If an OTHERWISE is present, it is not executed either.

SPACE_THEN_PRINT { YES | NO }

Specifies whether ANSI carriage controls for spacing are enabled for the first record on the new logical page following the execution of the CONDITION command. The abbreviation of this command is SPACE.

Note: This subcommand is effective for print files that contain ANSI carriage controls. It is not used for data files containing machine carriage controls, or a mixture of ANSI and machine carriage controls.

YES

Specifies that the ANSI carriage-control character in the first print record of the new page is enabled for spacing. The spacing action specified in the carriage control is performed after the eject to the new page. For example, if the carriage-control byte in the first record of the new page is a blank (skip one line before printing), then the first record skips the first line of the new page and prints at the second printline position.

NO

Specifies the ANSI carriage-control character spacing action is suppressed for the first print record of the new page. If this
record contains a carriage-control spacing value, such as “blank,” “0,” or “–,” the spacing is ignored and the record will print at the first printline position on the new page. Channel code values will not be ignored. If the first print record contains a valid channel code value of 1–9, or A–C, then the first record on the new page will print at the printline defined with that channel code.

WHEN parameters

Marks the start of the conditional comparison parameters. At least one WHEN subcommand is required.

comparisontype= { EQ | NE | GT | GE | LT | LE }  

Specifies the type of comparison that is to be performed between the data in the comparison field (the portion of the record specified by START and LENGTH) and the constant text defined in the text parameter.

The choices are:

   EQ equal to
  NE not equal to
   GT greater than
   GE greater than or equal to
   LT less than
   LE less than or equal to

   text

Specifies constant text for comparison with the comparison field text. The constant text length must be the same as the value on the LENGTH subcommand, with a maximum length of 8000 bytes. Examples of valid text are:

   2C(3)'AB'  
   K'321,400'  
   X'41FE7799' 2 'CHARS'

Any values or parameters that are valid for the TEXT subcommand within the FIELD command may be used as text; see the TEXT subcommand in “Subcommands” on page 160.

CHANGE

Specifies that the contents of the comparison field in this record are to be compared with the field in the record last processed by the same CONDITION command.

This parameter is an alternative to the comparisontype and text parameter combination but can be specified only once in a CONDITION command.

The results of the comparison is either TRUE or FALSE.
TRUE
When the contents of the comparison field have changed from one record to the next.

FALSE
When PSF processes the data, if the comparison field lies outside the boundary of the current record, which may occur with variable-length records or with truncated trailing blanks, the current record will not be used in future comparisons.

CHANGE is always false if used with the first WHEN subcommand of a series (no previous record to compare against). Whenever a new data map (one with a different name) is invoked, all the CHANGE comparisons are reset. Field values in the previous data map are not retained.

BEFORE
Specifies that the conditional action takes place before the current line or subpage is processed. This is the default.

AFTER
Specifies that the conditional action takes place after the current line or subpage is processed.

LINE
Specifies that the conditional action takes place either before or after the current line.

SUBPAGE
Specifies that the conditional action takes place either before or after the current subpage. This is the default.

For a description of subpages, see “Logical Page” on page 7.

`copygroup { CURRENT | = } | FIRST | { NULL | / } | NEXT | COPYGROUP cgname`
Specifies a copy group to be invoked if the condition is true.

Note: Any copy group action (except NULL) restarts the page format.

`{ CURRENT | = }`
Invoke the current copy group again. This results in ending printing on the current sheet and resuming on the front side of a new sheet. This is the default.

The page format is restarted. This means that the first input record to go on the new page is printed using the first PRINTLINE command of the current page format, and so on. For example, data that was to be printed as subpage 4 on the sheet might be printed on subpage 1 on the new sheet.
**CONDITION**

**Note:** The character "=" can be used for CURRENT.

**FIRST**
Invokes the first copy group in the current form definition.

{ **NULL** | / }
Retains the current copy group, taking no action. The character "/" can be used for NULL.

**NEXT**
Invokes the next copy group in the current form definition.

**Note:** If NEXT is specified from the last copy group in the form definition, the first copy group in the form definition will be used.

**COPYGROUP** *cgname*
Uses the named copy group defined in the current form definition. The name must contain 1 to 8 alphanumeric characters.

**pageformat** { **CURRENT** | = } | **FIRST** | { **NULL** | / } | **NEXT** | **PAGEFORMAT** *pfname*
Specifies a page format to be invoked if the condition is true.

{ **CURRENT** | = }
Invokes the current page format again. This results in ending printing on the current sheet and resuming on the front side of a new sheet.

The page format is restarted. This means that the first input record to go on the new page is printed using the first PRINTLINE command of the current page format, and so on.

The character "=" can be used for CURRENT.

**FIRST**
Invokes the first page format in the current page definition.

{ **NULL** | / }
Retains the current page format, taking no action. The character "/" can be used for NULL. This is the default.

**NEXT**
Invokes the next page format in the current page definition.

**Note:** If NEXT is specified from the last page format in the page definition, the first page format in the page definition will be used.
PAGEFORMAT \textit{pfname}

Uses the named page format defined in the current page definition. The name must contain 1 to 8 alphanumeric characters.

NEWFORM

Specifies that the only action to be taken is skipping to the front of a new form (sheet) and restarting the page format.

\textbf{Note:} This parameter is an alternative to using the parameters \texttt{[copygroup [pageformat]]} and is equivalent to specifying CURRENT for the copy group parameter and NULL for the pageformat parameter. CURRENT NULL are the respective defaults for copy group and pageformat parameters; therefore, NEWFORM is the default action.

NEWSIDE

Specifies that the only action to be taken is skipping to a new side (either the back of the current sheet or the front of a new sheet) and restarting the page format.

\textbf{Note:} This parameter is an alternative to using the parameters \texttt{[copygroup [pageformat]]} and is equivalent to specifying NULL for the copy group parameter and CURRENT for the pageformat parameter.

\textbf{Note:} Conditional processing does not result in unnecessary blank pages.

If the line currently being processed is the first line on a side, then:

- A \texttt{copygroup} or NEWFORM action taking effect BEFORE LINE does not force an additional new form.
- A \texttt{pageformat} or NEWSIDE action taking effect BEFORE LINE does not force an additional new side.

Similarly, additional sides or forms are not forced by BEFORE SUBPAGE if the line currently being processed is in the first subpage on a side or a form.

OTHERWISE \textit{parameters}

Marks the start of a conditional action to be taken if all preceding WHEN comparisons have proved false. The syntax is the same as the WHEN subcommand, except that the comparison parameters (\texttt{comparisontype text} or ‘CHANGE’) are not used. See the WHEN parameters starting with BEFORE on page 155 for a description of the parameters.

If the OTHERWISE subcommand is not used within the sequence, no action is taken. This is the same as if an OTHERWISE NULL NULL had been entered.
Note: OTHERWISE is not executed if any part of the comparison field specified by the combination of START and LENGTH is outside the boundaries of the data record.

ENDSUBPAGE Command

ENDSUBPAGE ;

The ENDSUBPAGE command is used to identify the end of a subpage for conditional processing.

You can specify the ENDSUBPAGE command at any point in a page definition command stream where a PRINTLINE command can occur. However, you must not enter the ENDSUBPAGE command between a PRINTLINE command and its associated FIELD or CONDITION command.

If an ENDSUBPAGE command is not specified, the entire page format is treated as one subpage.

FIELD Command
FIELD
[START byte ]
[LENGTH n ]
[TEXT [ duplication ] texttype [ L(m) ] ‘text’ ]
[FONT name1 [ , name2 ] ]
[POSITION horizontal vertical ]
[DIRECTION { ACROSS | DOWN | BACK | UP & rbrc. ]]
[SUPPRESSION name ]
[COLOR colorme ]
[RGB rvalue gvalue bvalue ]
[HIGHLIGHT hvalue { COVERAGE=cvalue BLACK=bvalue } ]
[CMYK=cvalue mvalue yvalue kvalue ]
[CIELAB=Lvalue (-)c1value (-)c2value ]
[BARCODE [name] ]
[TYPEN | CODE39 (same as 1)
 | MSI (same as 2)
 | UPCB (same as 3)
 | UPCE (same as 5)
 | UPC2SUPP (same as 6)
 | UPC5SUPP (same as 7)
 | EANB (same as 8)
 | EAN13 (same as 9)
 | IND2OF5 (same as 10)
 | MAT2OF5 (same as 11)
 | ITL2OF5 (same as 12)
 | CDB2OF7 (same as 13)
 | CODE128 (same as 17)
 | EAN2SUP (same as 22)
 | EAN5SUP (same as 23)
 | POSTNET (same as 24)
 | RM4SCC (same as 26)
 | MOD { n }
 | HRI { ON | ABOVE | BELOW | OFF } ]
SSASTERISK { ON | OFF }
HEIGHT { n [unit] }]
MODWIDTH { n }
RATIO { n } ]
where
duplication = Dn
texttype = [ C | X ] [ X | G | K ]
horizontal = { [ - ] x [ unit ] | { CURRENT | * } }
vertical = { [ - ] y [ unit ] | NEXT | { CURRENT | * } }

The FIELD command identifies a field in a data record or supplies a field of constant text, and positions where the field is on the page. More than one position on the page can be specified.

FIELD commands:
- Are subordinate to a PRINTLINE command
- Must follow a PRINTLINE command
- Must contain either a LENGTH subcommand or a TEXT subcommand
The FONT, DIRECTION, and COLOR subcommands do not have fixed defaults. If any of these subcommands is omitted, the value for the omitted subcommand is obtained from corresponding subcommand in the PRINTLINE command.

Subcommands

```
[ START n | [ * [ { + | - } n ] ] ]
```

Specifies the starting byte in the data record for the desired field.

- **n** Specifies the number of bytes from the first data byte in the record to be used as the starting point of the field. The first data byte position of an input record is 1.

  **Note:** The carriage-control character and the table-reference character are not considered data.

- ***** Denotes the next byte after the field identified in the previous FIELD command, excluding FIELD commands with constant TEXT.

  If START * was specified in the previous FIELD command, byte 1 is assumed.

- **+ n** Adds the value of n to the * byte position.

- **− n** Subtracts the value of n from the * byte position.

  If START is omitted and LENGTH is specified, then START * is assumed.

```
[ LENGTH n ]
```

Specifies the number (n) of bytes to process from the data record, beginning with the position specified in START.

```
[ TEXT [ duplication ] texttype [ L(m) ] 'text' ]
```

Specifies the constant text that is to be printed in the output. A maximum of 65535 bytes of text can be provided in one page format.

  **Note:** This text is considered constant in that the same text is printed each time. In reference to the CONSTANT command within a form definition, this text is considered variable because the text prints only where variable data is allowed to print.

- **duplication=Dn** Specifies the number of times the text is to be repeated (use a decimal number). The maximum times the text is repeated varies depending on the size of the text. The default is 1.

- **texttype=[ C | X ] | [ X | G | K ]** Specifies the type of text.

  - **C** Indicates that the text contains single-byte code characters, which includes all Roman alphabetic characters (for example, those
used for English). Any valid character code can be specified, including blanks. This is the default.

\**X** Indicates that the text contains hexadecimal codes (in groups of two hexadecimal codes) that specify values from \( \text{X'}00' \) through \( \text{X'}\text{FE}' \).

\**G** Indicates that the text contains double-byte code characters (for example, kanji characters).

Characters in type G text must start with shift-out (SO \( \text{X'}0E' \)) and end with shift-in (SI \( \text{X'}0F' \)) characters within opening and closing apostrophes (\( \text{X'}7D' \)).

\**K** Indicates that the text contains kanji numbers enclosed in apostrophes. Kanji numbers are separated by commas:

\[ \text{K}'321,400' \]

Valid double-byte character set (DBCS) codes are from \( \text{X'}41' \) through \( \text{X'}\text{FE}' \) for each byte. PPFA checks this range. Code \( \text{X'}4040' \) (blank) is the only exception.

\textbf{Valid}: \( \text{X'}4040' \), \( \text{X'}4041' \), \( \text{X'}41\text{FE}' \), \( \text{X'}\text{FE41}' \), \( \text{X'}\text{FEFE}' \)

\textbf{Invalid}: \( \text{X'}2040' \), \( \text{X'}413E' \), \( \text{X'}4100' \), \( \text{X'}7F00' \), \( \text{X'}\text{FE3E}' \)

\**L(m)** Specifies the length of text (use a decimal number in parentheses). When the actual length of the text is different from \( m \), the \( m \) specification is honored. That is, the text is either padded with blanks to the right or truncated.

\'text\'

 Specifies the text.

Examples:

- When TEXT 2C(3)'AB' is specified, 'AB AB ' is generated. The blanks are generated because of the (3) specification.
- TEXT 2C(1)'AB' generates 'AA', truncating the Bs.

\[ \text{[ FONT name1 [, name2 ]]} \]

Defines the font to be used for the field.

\textit{name1} Specifies the name of a font used to print the data. This font must have been defined in a previous FONT command in this page definition.
If Shift-Out, Shift-In (SOSI) processing is used, *name1* must be the single-byte font.

*name2* Specify only when using Shift-Out, Shift-In (SOSI) processing to dynamically switch between a single-byte font and a double-byte font within the field. *name2* must be the double-byte font.

**Notes:**

1. If this subcommand is not specified, the font specified in the preceding PRINTLINE command is used. If neither has been specified, PSF assigns a font.

2. When selecting a font in AIX, you could consider that the text will be selected in EBCDIC, not ASCII. Therefore, an EBCDIC font and code page 500 (also called International #5) should be used for *name1*.

\[
\text{\{ POSITION horizontal vertical \}}
\]

Specifies the starting position of the field in the printout.

**horizontal=** \{ \[ − \] \} \( x \) \{ \text{ \text{CURRENT} | *} \} \}

Do not mix \( x \) \text{unit} specifications with \text{CURRENT} or * except in ACROSS fields.

− Specifies that the \( x \) value is negative.

\( x \) Specifies the horizontal offset for the starting print position relative to the printline starting position. The value can be a number with up to three decimal places.

This value must be within the logical page boundary.

**unit** Specifies a unit of measurement for the \( x \) parameter. The choices are IN, MM, CM, POINTS, or PELS.

The default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

The PELS measurement equals one L-unit or 1/240 of an inch.

**CURRENT**

Specifies that the inline offset (relative to the field’s direction) is the end of the previous field. For the first field, use the PRINTLINE offset. This is the default.

**Note:** The meaning of CURRENT differs from the meaning of the PRINTLINE command parameter SAME.

* Alternate for CURRENT.

**vertical=** \{ \[ − \] \} \( y \) \{ \text{ \text{CURRENT} | *} \} \}

Do not mix \( y \) \text{unit} specifications with CURRENT or * except in ACROSS fields.
- Specifies that the \( y \) value is negative.

\( y \) Specifies the vertical offset for the starting print position relative to the \textit{printline starting position}. The value can be a number with up to three decimal places.

This value must be within the logical page boundary.

\textit{unit} Specifies a unit of measurement for the \( y \) parameter. The choices are IN, MM, CM, POINTS, or PELS.

The default is the most recent \texttt{SETUNITS} command value or IN (inch) if a \texttt{SETUNITS} command has not been issued.

\textbf{NEXT}

Specifies a field that is positioned down one line in the baseline direction (as defined in the \texttt{SETUNITS} command \texttt{LINESP} subcommand) from the previous field.

Use NEXT only in ACROSS fields.

\textbf{CURRENT}

Specifies that the inline offset (relative to the field’s direction) is the same as the previous field. That is, the baseline position does not change. For the first field, use the \texttt{PRINTLINE} offset. This is the default.

\textbf{Note:} The meaning of CURRENT differs from the meaning of the \texttt{PRINTLINE} command parameter \texttt{SAME}.

* Alternate for CURRENT.

\[ \textbf{[ DIRECTION \{ ACROSS \mid DOWN \mid BACK \mid UP \} ]} \]

Specifies the print direction of the field, relative to the upper-left corner as you view the logical page. If this subcommand is omitted, the direction specified in the governing \texttt{PRINTLINE} command is used.

\textbf{Note:} Not all printers can print in all directions. Refer to your printer documentation for more information.

\textbf{ACROSS} The page is printed with the characters added from \textit{left to right} on the page, and the lines are added from the top to the bottom.

\textbf{DOWN} The page is printed with the characters added from \textit{top to bottom} on the page, and the lines added are from the right to the left.

\textbf{BACK} The page is printed with the characters added from \textit{right to left} on the page, and the lines are added from the bottom to the top.
The page is printed with the characters added from bottom to top on the page, and the lines are added from the left to the right.

[ SUPPRESSION name ]
Specifies that this field can be suppressed.

name Specifies the name of a field to be suppressed.

Printing of this field is suppressed if this name is identified by a SUPPRESSION command within the form definition.

The same name can be used in one or more fields to suppress these fields as a group.

[ COLOR colorname ]
Specifies the OCA color of print for this field. This subcommand is recognized only by printers that support multiple-color printing. Refer to your printer publication for more information.

colorname

Values for colorname are NONE, DEFAULT, BLACK, BLUE, BROWN, GREEN, PINK, RED, TURQ (turquoise), or YELLOW. The color choices depend on the printer and, in some cases, the ribbon that is installed in the printer.

Note: In some printer publications, the color turquoise (TURQ) is called “cyan,” and the color pink (PINK) is called “magenta.”

NONE is the color of the medium. DEFAULT is the printer default color.

[ RGB rvalue gvalue bvalue ] { HIGHLIGHT hvalue
{ COVERAGE=cvalue BLACK=bvalue } } [ CMYK=cvalue mvalue yvalue kvalue ]
{ CIELAB=Lvalue (-)c1value (-)c2value }]

Specifies the color of print for this field supported in MO:DCA for the Red/Green/Blue color model (RGB), the highlight color space, the Cyan/Magenta/Yellow/Black color model (CMYK), and the CIELAB color model.

[ RGB rvalue gvalue bvalue ]

Three RGB integer values are used. The first (rvalue) represents a value for red, the second (gvalue) represents a value for green, and the third (bvalue) represents a value for blue. Each of the three integer values may be specified as a percentage from 0 to 100.

Note: An RGB specification of 0/0/0 is black. An RGB specification of 100/100/100 is white. Any other value is a color somewhere
between black and white, depending on the output device.

```
[ HIGHLIGHT hvalue {COVERAGE cvalue BLACK bvalue } ]
```

Indicates the highlight color model. Highlight colors are device dependent, and can be specified for the IBM InfoPrint Hi-Lite Color Printer Model 4005-HCI.

You can use an integer within the range of 0 to 65535 for the `hvalue`.

**Note:** An `hvalue` of 0 indicates that there is no default value defined; therefore, the default color of the presentation device is used.

**COVERAGE** indicates the amount of coverage of the highlight color to be used. You can use an integer within the range of 0 to 100 for the `cvalue`. If less than 100 percent is specified, the remaining coverage is achieved with the color of the medium.

**Note:** Fractional values are ignored. If **COVERAGE** is not specified, a value of 100 is used as a default.

**BLACK** indicates the percentage of black to be added to the highlight color. You can use an integer within the range of 0 to 100 for the `bvalue`. The amount of black shading applied depends on the **COVERAGE** percentage, which is applied first. If less than 100 percent is specified, the remaining coverage is achieved with black.

**Note:** If **BLACK** is not specified, a value of 0 is used as a default.

See “Color on the IBM InfoPrint HiLite Color Post Processor” on page 48 for more information.

```
[ CMYK=cvalue mvalue yvalue kvalue ]
```

Defines the cyan/magenta/yellow/black color model. **Cvalue** specifies the cyan value. **Mvalue** specifies the magenta value. **Yvalue** specifies the yellow value. **Kvalue** specifies the black value. You can use an integer percentage within the range of 0 to 100 for any of the CMYK values.
FIELD

[CIELAB=Lvalue (-)c1value (-)c2value]
Defines the CIELAB model. Use a range of 0.00 to 100.00 with Lvalue to specify the luminance value. Use signed integers from −127 to 127 with C1value and c2value to specify the chrominance differences.

Lvalue, c1value, c2value must be specified in this order. There are no defaults for the subvalues.

Note: Do not specify both an OCA color with the COLOR subparameter and an extended color model on the same FIELD or PRINTLINE command. The output is device dependent and may not be what you expect.

[BARCODE [name] [ TYPE {n | CODE39 | MSI | UPCA]
| UPCE | UPC2SUPP | UPCS2SUPP |
| EAN8 | EAN13 | IND2OF5 | MAT2OF5 |
| ITL2OF5 | CDB2OF7 | CODE128 | EAN2SUP |
| EAN5SUB | POSTNET | RM4SCC }
| MOD {n} | HRI {ON | ABOVE | BELOW | OFF }
| SSASTERISK {ON | OFF} | HEIGHT {n [unit]}
| MODWIDTH {n} | RATIO {n } ] ];
Specifies a bar code in a page definition.

The bar code name can be 1-8 characters long. Refer to your printer documentation for additional information about bar code support. Ensure that the bar code fits on the page or you will get errors at print time.

Please read your printer hardware documentation before using bar codes. The documentation will indicate which bar code types, modifiers, modwidth, element heights, and ratio values are valid for the printer.

PPFA does minimal verification of the bar code values. If you use the MOD, HEIGHT, MODWIDTH and RATIO parameters, ensure that the values you specify are valid for your printer.

For printer optimization, specify BARCODE name options in the first instance of a specific type of bar code. If this type is used again, position it as usual with START, LENGTH, and POSITION, but specify the barcode information using only BARCODE same-name-as-previously. The BARCODE subcommand is recognized only by printers that support BCOCA bar code printing; refer to Advanced Function Presentation: Printer Information (G544-3290) for more information.

For more information about bar codes, see Appendix D, “More About Bar Code Parameters” on page 223 and refer to Data Stream and Object Architectures: Bar Code Object Content Architecture Reference, S544-3766.
Note: Support for the BARCODE command is not available for OS/400.

TYPE Specifies the type of bar code symbol to be generated.

Note: If a type indicates "(same as n)," you may substitute the number given for the character name.

The following bar code types are supported:

name
Specifies a specific bar code name to be included in a page definition.

CODE39 (same as 1)

MSI (same as 2)
Specifies a bar code type of modified Plessey code.

UPCA (same as 3)
Specifies a bar code type of Universal Product Code (United States) and the Canadian Grocery Product Code, Version A

UPCE (same as 5)
Specifies a bar code type of Universal Product Code (United States) and the Canadian Grocery Product Code, Version E

UPC2SUPP (same as 6)
Specifies a bar code type of Universal Product Code (United States) two-digit Supplemental (periodicals).

UPC5SUPP (same as 7)
Specifies a bar code type of Universal Product Code (United States) five-digit Supplemental (paperbacks).

EAN8 (same as 8)
Specifies a bar code type of European Article Numbering 8 (includes Japanese Article Numbering-short).

EAN13 (same as 9)
Specifies a bar code type of European Article Numbering 13 (includes Japanese Article Numbering-standard).

IND2OF5 (same as 10)
Specifies a bar code type of Industrial 2-of-5.

MAT2OF5 (same as 11)
Specifies a bar code type of Matrix 2-of-5.
ITL2OF5 (same as 12)
  Specifies a bar code type of Interleaved 2-of-5,
  Automatic Identification Manufacturers Uniform
  Symbol Specification-I 2/5.

CDB2OF7 (same as 13)
  Specifies a bar code type of Codabar, 2-of-7,
  Automatic Identification Manufacturers Uniform
  Symbol Specification-Codabar.

CODE128 (same as 17)
  Specifies a bar code type of Code 128, Automatic
  Identification Manufacturers Uniform Symbol
  Specification-128.

EAN2SUP (same as 22)
  Specifies a bar code type of European Article
  Numbering, Two-digit Supplemental.

EAN5SUB (same as 23)
  Specifies a bar code type of European Article
  Numbering, Five-digit Supplemental.

POSTNET (same as 24)
  Specifies a bar code type of POSTal Numeric
  Encoding Technique (United States Postal
  Service), and defines specific values for the BSD
  module width, element height, height multiplier,
  and wide-to-narrow ratio fields.

RM4SCC (same as 26)
  Specifies a 4-state customer code defined by the
  Royal Mail Postal Service of England for bar
  coding postal code information.

MOD
  Specifies additional processing information about
  the bar code symbol to be generated (for example,
  MOD specifies whether a check-digit 4 should be
  generated for the bar code symbol).

\[ n \]
  The meaning of \( n \) differs between the types.
  For more information, see Table 15 on page 228.

If MOD is not specified, the MOD value defaults as
follows, depending on the bar code type specified:

---

4 Check digits are a method of verifying data integrity during the bar coding reading process.
HRI Specifies the human-readable interpretation (text characters) to be generated and placed above or below the bar code symbol, as directed.

ON Specifies that HRI should be generated at the default location for the barcode type.

ABOVE Specifies that HRI should be placed above the bar code symbol.

BELOW Specifies that HRI should be placed below the bar code symbol.

OFF Specifies that HRI should not be generated. If HRI is requested, the printer default font is used to render the HRI, instead of the font specified on the FIELD FONT subcommand.

SSASTERISK Specifies whether an asterisk is to be generated as the HRI for CODE39 bar code start and stop characters.

Note: SSASTERISK is ignored by all bar code types except CODE39.

ON Specifies that start and stop characters should be generated in the HRI.

OFF Specifies that start and stop characters should not be generated in the HRI.

HEIGHT Specifies the height of bar code element. For UPC and EAN bar codes, the total height includes the bar code and the HRI characters.

If HEIGHT is not specified, the printer default height is used.

Note: HEIGHT is ignored by bar code types that explicitly specify the element heights (for example, POSTNET or RM4SBC).

\[ n \] Specifies the height of the bar code.
[\text{unit}]

Specifies a unit of measurement for the \text{HEIGHT} parameter. The choices are IN, MM, CM, POINTS, or PELS.

\textbf{Note:} If no unit is specified, the default is the most recent \text{SETUNITS} command value or IN (inch) if a \text{SETUNITS} command has not been issued.

\textbf{MODWIDTH}

Specifies the width of the smallest defined bar code element, using mils (thousandths of an inch). For bar code types that explicitly specify the module width (for example, \text{POSTNET} and \text{RM4SCC}), this field is ignored. The range of values allowed is 1-254. If \text{MODWIDTH} is not specified, the printer default MODWIDTH is used.

\textit{n} Specifies the width of each module, using thousandths of an inch (1/1000) as the unit of measurement.

\textbf{RATIO}

Specifies the ratio between the width of the wide and the narrow bar code elements. The range of values allowed is 100-500, but you must specify a value appropriate for your printer and bar code type or you will get errors at print time.

If \text{RATIO} is not specified, the printer default ratio is used.

\textit{n} The \text{RATIO} is specified as a percent value. For example, form \texttt{n}nn. For example, 200 represents a ratio of 2 to 1; 250 represents a ratio of 2.5 to 1. For most bar code symbols, the \text{RATIO} value should be between 200 and 300. For bar code types that explicitly specify the module width (for example, \text{POSTNET} and \text{RM4SCC}, this field is ignored. If \text{RATIO} is not specified, the default ratio for the bar code symbol is used.

\textbf{FONT Command}

\begin{verbatim}
FONT name1
   { name2 | CS character-set-name CP code-page-name | GRID hex-grid }
   [SBCS – DBCS ]
   [HEIGHT points ]
   [RATIO percent ]
   [ROTATION { 0 | 90 | 180 | 270 }]
};
\end{verbatim}
The FONT command is used to identify the fonts that are to be specified in the PRINTLINE, FIELD, and TRCREF commands. A maximum of 127 font names for each page definition can be identified.

**Note:** Naming a font with the FONT command does not, by itself, affect your output. You must specify the font in a PRINTLINE, FIELD, or TRCREF command for the font to become effective. If you do not name a font, the default font will be used.

FONT commands immediately follow the PAGEDEF command. A separate FONT command is required:
- For each font used within a page definition
- For each rotation of the same font

**Note:** See the TRCREF command for the exception.

```
FONT name1 { name2 | CS character-set-name CP code-page-name | GRID hex-grid }
```

Identifies the fonts to be specified in the PRINTLINE, FIELD, and TRCREF commands.

- **name1** Specifies an alphanumeric name of 1 to 16 characters (local name) of the font to be used in this page definition. The name must conform to the token rules and must be unique within this page definition.
- **name1** is used in the PRINTLINE, FIELD, or TRCREF commands of a page definition.
- **name1** is optional if **name2** is specified.
- **name2** Specifies an alphanumeric name of 1 to 6 characters (user-access name) of the font to be used in this page definition.
- **character-set-name** Specifies an alphanumeric name of 1 to 6 characters of the character set to be used in this page definition.
- **code-page-name** Specifies an alphanumeric name of 1 to 6 characters of the code page to be used in this page definition.
- **hex-grid** Specifies the 16-character hexadecimal GRID.

**Subcommands**

[ **SBCS** | **DBCS** ]

- Specifies single-byte or double-byte fonts.
- **SBCS** Specifies that the font is a single-byte character set. This is the default.
- **DBCS** Specifies that the font is a double-byte character set.
OVERLAY

[ HEIGHT points ]
Specifies the height of the outline font.
points Each point is equal to 1/72 of one inch.

[ RATIO percent ]
Specifies the ratio of scaling the width relative to the height in an outline font.
percent Represents the percent of the “normal” width of the character that will be printed. For example, specifying RATIO 50 yields a font with characters half as wide as normal, and specifying RATIO 200 yields a font with characters twice as wide (200% as wide) as normal. If RATIO is specified, you must also specify the HEIGHT.

[ ROTATION { 0 | 90 | 180 | 270 } ]
Specifies the rotation of characters in degrees. The specified value is relative to the inline direction of a printline or field. Valid rotations are 0°, 90°, 180°, or 270°; 0° is the default.

OVERLAY Command

OVERLAY name ;

This OVERLAY command is used to identify the overlay that will be positioned on a page at some spot other than the position defined within the overlay. This function is similar to the SEGMENT command. A separate OVERLAY command is required for each overlay. A maximum of 127 OVERLAY commands (each of the 127 names must be unique) can be specified for each page format.

The OVERLAY commands are nested within the PAGEFORMAT command.

PAGEFORMAT
[ TRCREF]
[SEGMENT]
[ OVERLAY ]
...
[ OVERLAY ]

For the overlay to be used, the end-user must embed an Include Page Overlay (IPO) structured field within the line data or unformatted ASCII to be printed. The same name must appear within the structured field as identified by this command, and the page origin must be stated.

To include page overlays without using the IPO structured field, see the “PRINTLINE Command” on page 178.

OVERLAY name;
Identifies the overlay that will be positioned on the page.
name Specifies the user-access name of an overlay.
Note: The prefix ‘O1’ is not part of the six-character user-access name. The overlay name can be an alphanumeric.

PAGEDEF Command

```
PAGEDEF name
  [WIDTH n [ unit ]]
  [HEIGHT n [ unit ]]
  [LINEONE x [ unit ] y [ unit ]]
  [DIRECTION { ACROSS | DOWN | BACK | UP }]
  [REPLACE { YES | NO }]
```

A page definition is a resource used to define how data is to be formatted on a logical page. When generated by PPFA, a page definition is stored as a resource in the page-definition library.

This command must be specified when you define a page definition. All of the PAGEDEF subcommands are optional; defaults are assumed.

Note: Values assigned within the subcommands or the default values become the values for any PAGEFORMAT subcommand not specified. REPLACE is not a PAGEFORMAT subcommand, so its default is not carried forward.

**PAGEDEF name**

Identifies the page definition to be used with the print job.

`name` Defines an alphanumeric name of 1 to 6 characters for the page definition. When page definitions are generated, PPFA assigns the prefix ‘P1’ to this name as the resource name.

**Subcommands**

```
[ WIDTH n [ unit ]]
```

Defines the width of the logical page.

`n` A number with up to three decimal places is used. The width may vary according to the type of printer being used. For more information, refer to your printer documentation. The default is **8.3 IN**.

`unit` Specifies a unit of measurement for the WIDTH subcommand. The choices are IN, MM, CM, POINTS, or PELS.

**Note:** If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.
[HEIGHT n [ unit ]]
Defines the height of the logical page.

n A number with up to three decimal places is used. The height may vary according to the type of printer being used. For more information, refer to your printer documentation. The default is 10.8 IN.

unit Specifies a unit of measurement for the HEIGHT subcommand. The choices are IN, MM, CM, POINTS, and PELS.

Note: If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

[LINELINE x [ unit ] y [ unit ]]
Specifies the values for the MARGIN and TOP parameters used in the POSITION subcommand of the PRINTLINE command.

x Specifies the offset from the left edge of the logical page (margin position). A number with up to three decimal places is used. This value must be within the logical page boundary. The default is 0.

unit Specifies a unit of measurement for the x parameter. The choices are IN, MM, CM, POINTS, or PELS.

Note: If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

y Specifies the offset from the top of the logical page (top line position). A number with up to three decimal places is used. This value must be within the logical page boundary.

The default is 1 L-unit less than 80% of one line space down from the top of the logical page. The size of one line space is specified in the SETUNITS command.

unit Specifies a unit of measurement for the y parameter. The choices are IN, MM, CM, POINTS, or PELS.

Note: If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

[DIRECTION { ACROSS | DOWN | BACK | UP }]
Specifies the print direction of the logical page. Not all printers can print in all print directions. For more information, refer to your printer documentation.
**Note:** Some printers such as the IBM 3835 Page Printer and the IBM 3900 Advanced Function Printer have a different media origin and require different direction settings than most page printers. For printing in the landscape page presentation when using wide forms, the PRESENT subcommand must be specified on the FORMDEF command to produce readable output. Alternatively, if you have existing page definitions, the UP direction can be used in the page definition without changes to the form definition to produce the same result.

**ACROSS** The page is printed with the characters added *left to right* in each line, and the lines added from the top to the bottom.

**DOWN** The page is printed with the characters added to the page from *top to bottom*, and the lines added from the right to the left.

**BACK** The page is printed with the characters added to the page from *right to left*, and the lines added from the bottom to the top.

**UP** The page is printed with the characters added to the page from *bottom to top*, and the lines added from the left to the right.

[ **REPLACE** { **NO** | **YES** } ]

Specifies whether this page definition is to replace an existing one with the same resource name in the library.

**NO** This page definition does not replace one with the same resource name in the library.

If a page definition with the same resource name does not exist in the library, this page definition is stored.

**YES** If a page definition with the same resource name already exists in the library, this page definition replaces it.

If a page definition with the same resource name does not exist in the library, this page definition is stored.
Page formats are subsets of page definitions. If you want to use more than one set of specifications to format a page within a single print job, you must use more than one page format. To change page formats, use conditional processing or insert an Invoke Data Map structured field in your print file. (Page formats are known to PSF as data maps.) If you do not use conditional processing or if you do not insert an Invoke Data Map structured field, PSF uses only the first page format in the page definition. Page formats are placed in the page definition in the order in which they are generated.

PAGEFORMAT subcommands have no fixed defaults. The entire PAGEFORMAT command and all of its subcommands can assume defaults. If any PAGEFORMAT subcommand is omitted, its value is selected from the corresponding subcommand in the governing PAGEDEF command.

This command can be omitted for the first page format in a page definition if only one page format is used. If omitted, PPFA assigns a page format name by using the page-definition name, including the ‘P1’ prefix.

**PAGEFORMAT name**

- **[WIDTH \( n \) [ unit ]]**
  - Defines the width of the logical page.
  - \( n \): A number with up to three decimal places is used. The width may vary according to the type of printer being used. For more information, refer to your printer documentation.
  - **unit**: Specifies a unit of measurement for the WIDTH subcommand. The choices are IN, MM, CM, POINTS, or PELS.

  **Note:** If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

**Subcommands**

- **[ HEIGHT: \( n \) [ unit ]]**
- **[LINEONE \( x \) [ unit ] \( y \) [ unit ]]**
- **[DIRECTION { ACROSS | DOWN | BACK | UP }]**
[**HEIGHT** \( n \) \[ *unit* \]]

Defines the height of the logical page.

\( n \quad \) A number with up to three decimal places is used. The height may vary according to the type of printer being used. For more information, refer to your printer documentation.

*unit* \quad \) Specifies a unit of measurement for the HEIGHT parameter. The choices are IN, MM, CM, POINTS, or PELS.

**Note:** If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

[**LINEONE** \( x \) \[ *unit* \] \( y \) \[ *unit* \]]

Specifies the values for the MARGIN and TOP parameters used in the POSITION subcommand of the PRINTLINE command.

\( x \quad \) Specifies the offset from the left edge of the logical page (margin position). A number with up to three decimal places is used. This value must be within the logical page boundary. The default is 0.

*unit* \quad \) Specifies a unit of measurement for the \( x \) parameter. The choices are IN, MM, CM, POINTS, or PELS.

**Note:** If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

\( y \quad \) Specifies the offset from the top of the logical page (top line position). A number with up to three decimal places is used. This value must be within the logical page boundary.

The default is one L-unit less than 80% of one line space down from the top of the logical page. The size of one line space is specified in the LINESP subcommand of the SETUNITS command.

*unit* \quad \) Specifies a unit of measurement for the \( y \) parameter. The choices are IN, MM, CM, POINTS, or PELS.

**Note:** If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

[**DIRECTION** \{ ACROSS | DOWN | BACK | UP \}]

Specifies the print direction of the logical page. Not all printers can print in all print directions. For more information, refer to your printer documentation.
Note: Some printers such as the IBM 3835 Page Printer and the IBM 3900 Advanced Function Printer have a different form origin and require different direction settings than most page printers. For printing in the landscape page presentation when using wide forms, the PRESENT subcommand must be specified on the FORMDEF command to produce readable output. Alternatively, if you have existing page definitions, the UP direction can be used in the page definition without changes to the form definition to produce the same result.

**ACROSS** The page is printed with the characters added to the page from *left to right*, and the lines added from the top to the bottom.

**DOWN** The page is printed with the characters added to the page from *top to bottom*, and the lines added from the right to the left.

**BACK** The page is printed with the characters added to the page from *right to left*, and the lines added from the bottom to the top.

**UP** The page is printed with the characters added to the page from *bottom to top*, and the lines added from the left to the right.

---

**PRINTLINE Command**

```
PRINTLINE
[REPEAT n [ FIELD | LINE ] ]
[CHANNEL n ]
[FONT name1 [ , name2 ] ]
[POSITION horizontal [ RELATIVE ] vertical ]
[OVERLAY name [ relative_xpos relative_ypos ] ]
[SEGMENT name [ relative_xpos relative_ypos ] ]
[DIRECTION { ACROSS | DOWN | BACK | UP } ]
[COLOR colorname ]
[ RGB rvalue gvalue bvalue ]
[ HIGHLIGHT hvalue { COVERAGE=cvalue BLACK=bvalue } ]
[ CMYK=cvalue mvalue yvalue kvalue ]
[ CIELAB=Lvalue (-)c1value &paren;(-)c2value ]
```

where

- horizontal = x [ unit ] | MARGIN | { SAME | = }
- vertical = y [ unit ] | TOP | NEXT | { SAME | = }

---

**PRINTLINE**

The PRINTLINE command specifies the printing of one data record on a line. If a formatted printline is to be printed, one or more FIELD commands must follow the governing PRINTLINE command; at least one is required. If this is not
done, field processing is not performed and the unformatted data is printed.

Subcommands

[ REPEAT n [ FIELD | LINE ] ]

Specifies the number of printlines that are to be printed on a logical page. The direction and font specified within this printline applies to all lines printed. By using this command, you do not have to write specifications for each line.

Note: If the REPEAT subcommand is omitted, only one line is printed for this PRINTLINE command.

\( n \)

This value specifies the number of printlines for a logical page; the maximum value is 65535.

REPEAT 0  Not valid
REPEAT 1  Only one line is printed

If the CHANNEL or POSITION subcommands are specified within this PRINTLINE command, they apply only to the first line.

If this PRINTLINE is followed by several FIELD commands, the related field controls are also repeated.

FIELD

 Specifies that fields associated with repetitions of this PRINTLINE are to be positioned based on the first instance of the same field.

This parameter has no affect in fields with the same direction as the PRINTLINE of which they are a part.

This parameter specifies that the direction of repetition—for a given field—will be the direction of the first instance of this field, plus 90°. Therefore, every field of an ACROSS PRINTLINE will be repeated down the page, regardless of the direction of the FIELD.

LINE

 Specifies that fields associated with repetitions of this printline are to be positioned based on the repetition of the PRINTLINE itself.

This parameter has no effect in fields with the same direction as the PRINTLINE of which they are a part.

This parameter specifies that the direction of repetition—for a given field—will be the direction of the associated PRINTLINE plus 90°. Therefore, every field of an ACROSS
PRINTLINE will be repeated down the page, regardless of the direction of the FIELD.

[ CHANNEL \( n \) ]

Used to specify line spacing, skipping within a logical page, or page ejection (skipping to a new page). This subcommand is equivalent to the Forms Control Buffer (FCB) channel.

\( n \) The range of channels is 1 to 12. These correspond to carriage-control characters in the data. There is no default.

[ FONT \( \text{name1} [ , \text{name2} ] \) ]

Defines the font to be used for the printline.

\( \text{name1} \) Specifies the name of a font used to print the data. This font must have been defined in a previous FONT command in this page definition.

If Shift-Out, Shift-In (SOSI) processing is used, \( \text{name1} \) must be the single-byte font.

\( \text{name2} \) Specify only when using Shift-Out, Shift-In (SOSI) processing to dynamically switch between a single-byte font and a double-byte font within the printline. \( \text{name2} \) must be the double-byte font.

Notes:

1. If this subcommand is not specified and TRC (Table Reference Character) bytes are specified in the print data, PSF will use the font indicated by the TRC byte. Otherwise, PSF will select a default font.

2. When selecting a font in AIX, you should consider that the text will be selected in EBCDIC, not ASCII. Therefore, an EBCDIC font and code page 500 (also called International #5) should be used for \( \text{name1} \).

[ POSITION horizontal [ RELATIVE ] vertical ]

Specifies the starting position of the printline in the printout.

\( \text{horizontal} \times [ \text{unit} ] \) | MARGIN | { SAME | = }

\( \text{x} \) Specifies the horizontal offset from the left side of the logical page. The value is a number with up to three decimal places. This value must be within the logical page boundary.

\text{Note: } \text{x} \text{ must be specified if } \text{unit} \text{ is specified.}

\text{unit} Specifies a unit of measurement for the \( \text{x} \) parameter. The choices are IN, MM, CM, POINTS, or PELS.

\text{Note: } \text{If no unit is specified, the default is the most recent SETUNITS command value or IN}
PRINTLINE

(inch) if a SETUNITS command has not been issued.

MARGIN Specifies this line starts at the position specified as the horizontal \((x)\) value in the previous LINEONE subcommand within this page definition.

SAME Specifies this line starts at the same horizontal offset position as the previous printline. If applied to the first printline of a logical page, the horizontal position is 0, which is the default.

= Alternate for SAME.

RELATIVE

Specifies that the following vertical position value is to be processed as a relative value. The printline is positioned relative to the last printline placed on the page.

If a set of printlines were skipped over in the page definition because of a skip-to-channel carriage control, and the new active printline contains a relative vertical position, the output line will be positioned relative to the location of the last line printed on the page.

Note: If both TOP and RELATIVE are requested for the Y position value, the RELATIVE request is ignored.

When using RELATIVE positioning, PPFA does not flag off-the-page conditions for the position of a printline or for any overlays or segments placed relative to that printline. Printlines that fall outside the bounds of the logical page are flagged by PSF at run time.

When specifying RELATIVE, use the minus sign to indicate any negative values for the PRINTLINE vertical position; you may use the plus sign to indicate positive values. If no sign is used, a positive value is assumed.

The DIRECTION for a relative printline must be ACROSS. Fields associated with a relative printline must have the same DIRECTION as the printline and must match the pageformat DIRECTION.

If RELATIVE is specified with “SAME” or “=” as the “y” value, the relative value in the printline will be +0.

Relative positioning is allowed on a PRINTLINE command only if the PRINTLINE and all its associated FIELD commands are formatted to print in the same direction as the PAGEFORMAT. That is, the DIRECTION parameter in the PRINTLINE and any associated FIELD commands must specify (or default to) ACROSS. The DIRECTION in the PAGEFORMAT or
The PRINTLINE command in which relative positioning is used must specify a CHANNEL parameter. The “n” value specified for the CHANNEL parameter cannot be used for any other PRINTLINE in the same PAGEFORMAT.

```plaintext
setunits linesp 6 lpi;
PAGEDEF rel9 replace yes
direction across width 8.5 in height 11.0 in;
PRINTLINE channel 1 repeat 7 position 0 IN 1.0 IN;

/* The fields will be placed at +120 pels, +24 pels (next) +
/* and +48 pels (.20 IN) from lines previously placed on page */

setunits linesp 10 lpi;
PRINTLINE channel 2 repeat 2 position 0 relative next;
FIELD START 1 LENGTH 3 position 0 IN .5 IN;
FIELD START 4 LENGTH 3 position 0 IN next;
FIELD START 7 LENGTH 3 position current .20 IN;
```

**vertical y [ unit ] | TOP | NEXT | { SAME | = }**

- **y** Specifies the vertical offset from the top side of the logical page. The value is a number with up to three decimal places. This value must be within the logical page boundary.

  **Note:** y must be specified if unit is specified.

- **unit** Specifies a unit of measurement for the y parameter. The choices are IN, MM, CM, POINTS, or PELS.

  **Note:** If no unit is specified, the default is the most recent SETUNITS command value or IN (inch) if a SETUNITS command has not been issued.

- **TOP** Specifies that the printline is placed in the position specified as the vertical (y) value in the previous LINEONE subcommand within this page definition.

- **NEXT** Specifies the printline is to be positioned down (on the logical page) one line (as defined in the LINESP subcommand of the last SETUNITS command) from the previous field. The LINESP subcommand of the SETUNITS command establishes the distance from one line to the next.

  When NEXT is specified for the first printline of a logical page, the starting position of the line is one line down from the top of the logical page, which is the default.
Note: The “down” direction is determined by the direction of the logical page (as specified in the page format), not the printline direction. NEXT is, therefore, mainly useful in ACROSS printlines.

**SAME**

Specifies this printline starts at the same vertical position as the previous printline.

= Alternate for SAME.

**OVERLAY [name [ relative_xpos relative_ypos ]]**

Specifies the name of an overlay that is to be positioned relative to the location specified in the PRINTLINE command in which the OVERLAY subcommand was named. The PAGEFORMAT OVERLAY command may contain the named overlays. The maximum number of overlays specified for a PAGEFORMAT including the PRINTLINE OVERLAY subcommand is 127.

Specifies the electronic overlay that is to be used with this subgroup.

*name*

Specifies either the local or user-access name.

**Notes:**

1. PPFA checks for duplication of local names. If there is a duplication, the page definition is generated, but a warning message is issued.

2. PPFA does not check for duplicate user-access names.

*relative_xpos relative_ypos*

Specifies the number of units (inches, mm, and so on) that are added to the position of the printline to position the top-left corner of the overlay. The values for horizontal and vertical may be (+) or (-). The maximum value is + or - 32760 L-units. For example:

- OVERLAY NAME1 2 in 1 in
- OVERLAY NAME2 5 mm 1 mm

**Note:** Any offset coded in the overlay itself is added to this offset.

**SEGMENT [name [ relative_xpos relative_ypos ]]**

Specifies the name of a segment that is to be positioned relative to the location specified in the PRINTLINE command in which the SEGMENT subcommand was named. The PAGEFORMAT SEGMENT command may contain the named segments. The maximum number of segments specified for a PAGEFORMAT including the PRINTLINE SEGMENT subcommand is 127.

Specifies the page segment that is to be used with this subgroup.
PRINLINE

name Specifies either the local or user-access name.

Notes:

1. PPFA checks for duplication of local names. If there is a duplication, the page definition is generated, but a warning message is issued.
2. PPFA does not check for duplicate user-access names.

relative_xpos relative_ypos

Specifies the number of units (inches, mm, and so on) that are added to the position of the printline to position the top-left corner of the page segment. The values for horizontal and vertical may be (+) or (-). The maximum value is + or - 32760 L-units. For example:

- SEGMENT MYSEG1 2 in 1 in
- SEGMENT MYSEG1 5 mm 1 mm

[DIRECTION { ACROSS | DOWN | BACK | UP }]

Specifies the print direction of the line relative to the upper-left corner as you view the logical page. Not all printers can print in all print directions. For more information about your printer, refer to your printer documentation.

If DIRECTION is not specified, the direction specified in the PAGEFORMAT command is used. Note that this direction is additive to the direction specified in the PAGEFORMAT command. See Appendix B, “More about Direction” on page 219.

ACROSS The printline direction is rotated 0 degrees relative to the direction specified in the PAGEFORMAT (the printlines are oriented in the same direction as the page).

DOWN The printline direction is rotated 90 degrees relative to the direction specified in the PAGEFORMAT.

BACK The printline direction is rotated 180 degrees relative to the direction specified in the PAGEFORMAT.

UP The printline direction is rotated 270 degrees relative to the direction specified in the PAGEFORMAT.

[ COLOR colorname ]

Specifies the printline data color. This subcommand is recognized only by printers that support multiple-color printing. Refer to your printer publication for information about the colors that can printed.

Values for colorname are NONE, DEFAULT, BLACK, BLUE, BROWN, GREEN, PINK, RED, TURQ (turquoise), or YELLOW. The color choices
 depend on the printer and, in some cases, the ribbon that is installed in the printer.

If you do not enter one of these colors, the default color for that printer is used.

Note: In some printer manuals, the color turquoise (TURQ) is called “cyan,” and the color pink (PINK) is called “magenta.”

NONE is the color of the medium. DEFAULT is the printer default color.

[ RGB rvalue gvalue bvalue ] [ HIGHLIGHT hvalue { COVERAGE=cvalue BLACK=bvalue } ] [ CMYK=cvalue mvalue yvalue kvalue ] [ CIELAB=Lvalue (−c1value (−c2value ) ]

Specifies the color of print for this field supported in MO:DCA for the Red/Green/Blue color model (RGB), the highlight color space, the Cyan/Magenta/Yellow/Black color model (CMYK), and the CIELAB color model.

[ RGB rvalue gvalue bvalue ]

Three RGB integer values are used. The first (rvalue) represents a value for red, the second (gvalue) represents a value for green, and the third (bvalue) represents a value for blue. Each of the three integer values may be specified as a percentage from 0 to 100.

Note: An RGB specification of 0/0/0 is black. An RGB specification of 100/100/100 is white. Any other value is a color somewhere between black and white, depending on the output device.

[ HIGHLIGHT hvalue { COVERAGE cvalue BLACK bvalue } ]

Indicates the highlight color model. Highlight colors are device dependent, and can be specified for the IBM InfoPrint Hi-Lite Color Printer Model 4005-HCI.

You can use an integer within the range of 0 to 65535 for the hvalue.

Note: An hvalue of 0 indicates that there is no default value defined; therefore, the default color of the presentation device is used.

COVERAGE indicates the amount of coverage of the highlight color to be used. You can use an integer within the range of 0 to 100 for the cvalue. If less than 100 percent is specified, the remaining coverage is achieved with the color of the medium.

Note: Fractional values are ignored. If COVERAGE is not specified, a value of 100 is used as a default.

BLACK indicates the percentage of black to be added to the highlight color. You can use an integer within the range of 0 to 100 for the bvalue. The
amount of black shading applied depends on the 
**COVERAGE** percentage, which is applied first. If less than 100 percent is specified, the remaining coverage is achieved with black.

**Note:** If **BLACK** is not specified, a value of 0 is used as a default.

```
[ CMYK=cvalue mvalue yvalue kvalue ]
```

Defines the cyan/magenta/yellow/black color model. *cvalue* specifies the cyan value. *mvalue* specifies the magenta value. *yvalue* specifies the yellow value. *kvalue* specifies the black value. You can use an integer percentage within the range of 0 to 100 for any of the CMYK values.

```
[ CIELAB=Lvalue (-)c1value (-)c2value ]
```

Defines the CIELAB model. Use a range of 0.00 to 100.00 with *Lvalue* to specify the luminance value. Use signed integers from −127 to 127 with *c1value* and *c2value* to specify the chrominance differences.

*Lvalue*, *c1value*, *c2value* must be specified in this order. There are no defaults for the subvalues.

**Note:** Do not specify both an OCA color with the **COLOR** subparameter and an extended color model on the same **FIELD** or **PRINTLINE** command. The output is device dependent and may not be what you expect.

---

### SEGMENT Command

```
SEGMENT name ;
```

Use the SEGMENT command only if you want page segments to be loaded to the printer before the page begins printing. If segments are used repeatedly and need to be available in the printer, this eliminates the need to load them each time. However, they do take up raster-pattern storage. If the segments are included on a page but not in the SEGMENT command, they are loaded to the printer as they are used in the print data.

A separate SEGMENT command is required for each page segment with a maximum of 127 SEGMENT commands within a single page format.

```
PAGEFORMAT
TRCREF
SEGMENT...

SEGMENT
```

A SEGMENT command is nested within the page format and follows the PAGEFORMAT command.
To include a page segment on a page without using an IPS structured field within the user data, see the “PRINTLINE Command” on page 178.

**SEGMENT name**

Specifies the alphanumeric name of 1 to 6 characters (user-access name) of the page segment. Each name must be unique within a single page format.

**Note:** The prefix ‘S1’ is not part of the six-character user-access name.

### SETUNITS Command

```plaintext
SETUNITS { x [ unit ] y [ unit ] [ LINESP n unit ] | LINESP n unit } ;
```

The SETUNITS command specifies the value and the unit of measurement that will be the defaults for any subsequent measurement parameter in all of the commands and subcommands. These values remain the default values until another SETUNITS command is specified. The SETUNITS command should be specified as the first command in a page definition. If neither this command nor a measurement parameter is specified, the defaults identified within the following description are used.

**SETUNITS x [ unit ] y [ unit ]**

Specifies the value and the unit of measurement that will be the defaults for any subsequent measurement parameter in all of the commands and subcommands.

**x** Specifies the number used for horizontal measurement. A number with up to three decimal places is used. The default is 1.

**Note:** This value affects subsequent OFFSET subcommands.

**unit** Specifies a unit of measurement for the x parameter. The choices are IN, MM, CM, POINTS, PELS, or CPI. The default is IN.

**y** Specifies the number used for vertical measurement. A number with up to three decimal places is used. The default is 1.

**Note:** This value affects subsequent OFFSET subcommands.

**unit** Specifies a unit of measurement for the y parameter. The choices are IN, MM, CM, POINTS, PELS, or LPI. The default is IN.

### Using CPI and LPI Units of Measurement

The CPI and LPI units of measurement make it possible to write the following command:

```plaintext
SETUNITS 10 CPI 6 LPI ;
```
This command sets the units of measurement for horizontal and vertical spacing in terms of characters per inch and lines per inch. You can then use the OFFSET subcommand specifications to increment the spacing one character or one line at a time. The distance specified by \( n \) characters over and by \( n \) lines down is defined in the governing SETUNITS command. In this example, there are 10 characters per inch (CPI) and 6 lines per inch (LPI).

**Subcommand**

**LINESP** \( n \) \( \text{unit} \)

Determines the line density or “leading” of the text. Any unit of measurement can be used. This subcommand values affects:

- The following PRINTLINE NEXT subcommand
- The vertical (y) position of the first line on a logical page when the LINEONE subcommand is not specified and the default is assumed

The default is 6 LPI. If LINESP is allowed to default to 6 LPI, the LINEONE default is 1 L-unit less than 80% of 1/6 inch.

\( n \) The meaning is determined by the type of unit-of-measurement specified in the unit parameter.

- **LPI** The number of lines per inch
- **All others** The distance between lines

\( \text{unit} \) Specifies a unit of measurement. The choices are:

- **IN** Inch
- **LPI** Lines-per-inch
- **MM** Millimeter
- **CM** Centimeter
- **PELS** L-units per inch (240 L-units in an inch)
- **POINTS** Points per inch (72 points in an inch)

**TRCREF Command**

```
TRCREF [ \( n \) ]
FONT \( \text{nam} \)
[DIRECTION { ACROSS | DOWN | BACK | UP } ]
[ROTATION { 0 | 90 | 180 | 270 } ] ;
```

The TRCREF command specifies the relationship between a font and a table-reference character (TRC) in the data. When specified, the TRCREF command must immediately follow a PAGEFORMAT command.

```
PAGEFORMAT
TRCREF
SEGMENT
OVERLAY
```
Depending on the value specified for \( n \), the TRC is interpreted by PSF as being either S/370 1403 line-mode compatible or S/370 1403 line-mode noncompatible: Notice that, if compatibility TRCs are to be used, no fonts should be specified in any PRINTLINE or FIELD commands within the same PAGEFORMAT.

0–3 Indicate a compatible TRC for a S/370 1403 line-mode data stream

4–126 Indicate a noncompatible TRC for a S/370 1403 line-mode data stream

Also notice that any TRC number outside the range of 0-3 results in non-compatibility TRCs for the entire page definition. If compatibility TRCs are used, do not specify fonts on PRINTLINE or FIELD commands within the same PAGEFORMAT.

\[ \text{TRCREF} \ [ \ n \ ] \]

Specifies the TRC numbers that can appear in print data.

\( n \)
The allowable values are 0 to 126; each TRCREF command must contain a unique number within a page format.

If \( n \) is omitted, PPFA automatically adds one to the \( n \) value of the previous TRCREF command in the sequence and assigns that value.

The default for the first TRCREF command is 0.

Note: You may have multiple TRCs pointing to the same font.

**Subcommands**

**FONT name**

Specifies the font that is associated with the TRC number.

\( name \)

Specifies the local name of a font; the font must be one that has been named in a FONT command.

If you have used both the user-access name and the local name in the FONT command, use the local name here. If you have used only the user-access name, use it here.

**[ DIRECTION { ACROSS | DOWN | BACK | UP } ]**

Specifies the print direction of the line relative to the upper-left corner as you view the logical page. Not all printers can print in all print directions. For more information about your printer, refer to your printer documentation.

The DIRECTION on the TRCREF command must match the DIRECTION of the PRINTLINE command with which the TRC is to be used. If TRCREF DIRECTION subcommand is not specified, DIRECTION ACROSS is assumed. Note that this direction is additive to the direction specified in the PAGEFORMAT command.

**ACROSS** The page is printed with the characters added to the page from *left to right*, and the lines added from the top to the bottom.
TRCREF

DOWN  The page is printed with the characters added to the page from top to bottom, and the lines added from the right to the left.

BACK  The page is printed with the characters added to the page from right to left, and the lines added from the bottom to the top.

UP  The page is printed with the characters added to the page from bottom to top, and the lines added from the left to the right.

[ ROTATION { 0 | 90 | 180 | 270 } ]

Specifies the rotation of characters in degrees. The specified value is relative to the inline direction of the printline.

Valid rotations are 0°, 90°, 180°, or 270°; 0° is the default.

If the TRCREF ROTATION subcommand is not specified, the default is the rotation value specified on the FONT command.
Appendix A. System Dependencies for PPFA

PPFA is a cross system product that operates under:

- VSE (Virtual Storage Extended)
- MVS (Multiple Virtual Storage)
- VM (Virtual Machine)
- AIX (Advanced Interactive Executive)
- OS/400 (Operating System 400)

For the level of the operating system on which PPFA can run, refer to the Licensed Program Specification.

PPFA creates page definitions and form definitions used for printing by PSF/MVS, PSF/VM, PSF/AIX, and OS/400. PPFA creates a data base file member containing AFPDS that can be used to create the OS/400 objects PAGDFN and FORMDF, using the CRTPAGDFN and CRTFORMDFN commands. Page definitions and form definitions created on one system can be used for printing on another system. However, not all versions of PSF support all functions provided by PPFA. Use the Programming Guide or User's Guide for your PSF system to determine which functions are supported by your system.

While page definitions and form definitions created on one system can be used on any of the systems, the method of creating these resources is different.

Each system is presented to show how PPFA creates page definitions and form definitions. In the examples, the prefixes F1 and P1 are automatically added by PPFA to the user name designated for form definitions and page definitions.

VSE Environment

PPFA can operate in any partition of VSE. It operates in batch mode but is able to operate in a partition occupied by an interactive processor.

Storing PPFA Resources

Form definitions and page definitions are stored by name in a library. In VSE, sub-libraries are created for form-definition and page-definition storage within the system library.

The following job control statements (JCS) give an example of a PPFA execution under VSE. The ‘C’ in Column 72 indicates a continuation.
Rules for VSE

The rules for VSE commands in a PPFA execution follow:

- All characters in the EXEC statement parameters must be uppercase. Each keyword in a parameter must be unique; PPFA issues an error message if any keywords are duplicated.
- AKQPPFA is the program name.
- SIZE= is optional. Defaults to program size.
- PARM= is used to input PPFA parameters.
  - FORMLIB= (or PAGELIB=) libraryname.sublibraryname
    - All library names are alphanumeric (1 to 7 characters); the first character must be alphabetic.
    - All sublibrary names are alphanumeric (1 to 8 characters) including the first character.
  - size=nnK or nnnM
    - Defines the work area in which PPFA compiles the page definitions and form definitions. The default is 128k and the minimum 4K.
- The format for the FORMLIB or PAGELIB parameters is:
  - FORMLIB= (or PAGELIB=) libraryname.sublibraryname, where library names are 1 to 7 characters long and sublibrary names are 1 to 8 characters long.
  - All characters (library and sublibrary names) are alphanumeric, except that the first character must be alphabetic.
- Libraries must be defined prior to PPFA execution; Otherwise, an ABEND occurs. PPFA can perform a syntax check without libraries being defined, but it cannot define its own libraries;
- The SYSIPT file drives PPFA. It contains the commands used to build form definitions and page definitions. The records are fixed-length records of either 80 or 81 bytes, which can be blocked. The last 8 bytes of the records are treated as comments.
MVS Environment

The following example shows you how to create page definitions and form definitions in the MVS environment.

Form definitions and page definitions are stored by name in a library.

The following job control language (JCL) statements are an example of PPFA execution under MVS:

```
//JOBPPFA JOB TOKYO
//STEP EXEC PGM=AKQPPFA,PARM='SIZE=128K'
//STEPLIB DD DSN=ppfa.program,DISP=SHR
//SYSPRINT DD SYSOUT=A
//FORMLIB DD DSN=ppfa.formlib,DISP=SHR
//PAGELIB DD DSN=ppfa.pagelib,DISP=SHR
//SYSIN DD *
```

PPFA control statements

. . . .

/* The SYSIN file contains the commands used to build form definitions and page definitions. The records can be fixed length or variable length, and they can be blocked. The maximum length for fixed-length records is 100 bytes; the maximum length for variable-length records is 104 bytes. In the case of fixed 80-byte records, the last 8 bytes are treated as comments.

The record format for the page-definition and form-definition data sets must be variable blocked (VBM). The block size and record length must be 8209 and 8205.

Note: When concatenating multiple data sets in the SYSIN data definition, you must ensure that the data set with the largest block size is first in the concatenation order. Otherwise, the output may not be what you expect.

VM Environment

To create a page definition and form definition running PPFA under VM, use the following command syntax:

Note: The defaults require only filename (fn) and filetype (ft) for your PPFA source file.

```
PPFA fn ft fm [ { [PAGEDEF ( ft [ fm | A1 ] )
[FORMDEF ( ft [ fm | A1 ] )
[LISTING ( ft [ fm | A1 ] )
[SIZE nnnn(K|M) ] ]
```

- Each keyword parameter can be abbreviated as two letters.
- All parameters in the command can be omitted. However, any optional parameter following an open parenthesis must be specified.
- Operands must be enclosed in parentheses when more than one operand is specified for one parameter. Parentheses can be omitted when only one operand is specified for one parameter. Also, the final closing parenthesis can be omitted.
Any operand string longer than eight characters is truncated to the first eight characters.

Any parameter or operand can be separated from others by parentheses or blanks. The only exceptions are the K and the M operands of a size parameter. For example, in size 256K you cannot separate the 256 from the K.

The same parameter must not be specified more than once in a command. If duplicate parameters or operands appear, PPFA issues an error message and terminates the program.

For errors associated with a VM execution command, PPFA issues an error message with a return code 20, and does not generate any files (object or listing).

No optional parameters can follow the open parenthesis occurring after the input source file ID.

The size parameter varies according to the size of the command stream. Most command streams do not need a size value because the default specifies enough space for processing.

PPFA fn ft [ fm ]

PPFA is the command to run PPFA on VM. The filename (fn) is the name of your file that contains the PPFA control statements. The filename (fn) and filetype (ft) are required parameters. When you specify only the fn and ft, the filemode goes to your default disk.

The record format of the PPFA input source file is either V or F. The variable record length is a maximum of 100 bytes. In the case of a fixed 80-byte record, the last 8 bytes are treated as comments.

The PPFA command may include any of four optional parameters: PAGEDEF, FORMDEF, LISTING, and SIZE.

**PAGEDEF Parameter**

PAGEDEF (which can be abbreviated as PA) is the keyword used to specify the name of a page-definition resource. (The filetype is required; the filemode is optional. If you do not specify a filemode, A1 is assumed.) The page-definition filename is obtained from your input file, and P1 is prefixed to that name.

As an example, for the command

```
PPFA PCOM DATA A1 ( PAGEDEF ( PAGEOBJ B1 ) )
```

the input file, PCOM DATA A1, contains the following control statements:

```
PAGEDEF PAGE1;
PRINTLINE;
FORMDEF FORM1;
```

The result is a page-definition resource file with the filename P1PAGE1, the filetype PAGEOBJ, and the filemode B1.

If the page definition parameter is not used, a page-definition resource with the default name P1 (the page definition name from input file) PDEF38PP A1 is created.
The record format of the object file is VM and VA (5A records). 5A records contain the character X'5A' in the first byte of each record. The record size is up to 8205 bytes.

**FORMDEF Parameter**

FORMDEF (which can be abbreviated as FO) is the keyword used to specify the name of a form-definition resource. (The filetype is required; the filemode is optional.) The filename is obtained from your input file, and F1 is prefixed to that name. As an example, for the command

```
PPFA PCOM DATA A1 ( FORMDEF ( FORMOBJ B1 ) )
```

the input file, PCOM DATA A1, contains the following control statements:

```
PAGEDEF PAGE1;
PRINTLINE;
FORMDEF FORM1;
```

The result is a form-definition resource file with the filename F1FORM1, the filetype FORMOBJ, and the filemode B1.

If the form-definition parameter is not used, a form-definition resource with the default name F1 (form-definition name from input file) FDEF38PP A1 is created.

The record format of the object file is VM and VA (5A records). The record size is up to 8205 bytes.

**LISTING Parameter**

LISTING (which can be abbreviated as LI) is the keyword used to specify the name of an output listing file. You can specify the filetype and filemode of the resource; the filetype is required. If you do not specify a filemode, A1 is assumed. The filename is the same as the PPFA input filename.

As an example, for the command

```
PPFA PCOM DATA A1 ( LISTING ( LISTOUT B1 )
```

the result is an output listing file with the name PCOM LISTOUT B1.

If the LISTING parameter is not used, an output listing file with the default name (PPFA input filename) LISTING A1 is created.

The record format of an output listing file is VA. The record length is 121 bytes (120 bytes + 1 byte (channel control number)). CC numbers are 0 to 12 in the first column of the line data file.

**SIZE Parameter**

SIZE (which can be abbreviated as SI) is the keyword used to specify the working area of PPFA.

You can specify any work-area size up to four digits of K or M bytes, in the range of greater than or equal to 64K, and less than or equal to 16M.
RUN and OPTIONS file

This is an example of the VM files that print your data file with the form definition and page definition that you specify.

--- VM EXEC Example ---

`/*THE ENVIRONMENT IS NOW SET UP TO PRINT */
'CP SP PRT TO NET NOHOLD CLASS A FORM PRT035 COPY 1';
'CP TAG DEV PRT WASVM SYSTEM';
'PSF EXAMP1 PRTDATA A1 ( OPTIONS (EXAMP1) )';
'/*RESTORE THE ENVIRONMENT TO PRINT SOMETHING OTHER THAN THIS EXAMPLE*/`

--- VM OPTIONS Example ---

`FORMDEF ( F1EXAMP1 FDEF38PP ) SEND`
`PAGEDEF ( P1EXAMP1 PDEF38PP ) SYSDISK`
`OVERLAY ( * OVLY38PP ) SYSDISK`

`* COMMON OPTIONS`

`CC
NOTRC
BIN 1
CKPTPAGE 0
DATACK UNBLOCK
NODUMP
FILE SEND
FONT ( * FONT3820 ) SYSDISK
MESSAGES NO
NOOPT
PAGESEG ( * PSEG38PP ) SYSDISK
TRACE OFF`
AIX Environment

The `ppfa` command creates form definitions and page definitions on the AIX/6000 operating system. After they are created, you can transfer the form definitions and page definitions to other operating systems (such as MVS, VM, or VSE) to use as AFP resources.

**Note:** Many form definitions and page definitions are supplied as part of PSF for AIX. For a list of these AFP resources, refer to *IBM Print Services Facility for AIX: Print Administration* or *IBM Print Services Facility for AIX: Print Submission* or *Using IBM InfoPrint Control for Production Printing*.

**Syntax**

```
ppfa [ -f path.ext ] [ -p path.ext ] [ -s path.ext ] [ -x ] inputfile
```

**Flags and Values**

You can specify these flags and values with the `ppfa` command.

- `inputfile` The file containing the PPFA source statements to be “processed.”
- `f path.ext` Add path and extension information to the names of form definitions generated by PPFA. (The name itself will come from the FORMDEF command.)
- `p path.ext` Add path and extension information to the names of page definitions generated by PPFA. (The name itself will come from the PAGEDEF command.)
- `s path.ext` Add path and extension information to the listing file. The name of the listing file will be the same as the name of the inputfile.

Thus, for “FORMDEF name” when PPFA was invoked with `ppfa -f path.ext infile`, it generates form definition `/path/name.ext`.

Thus, for “PAGEDEF name” when PPFA was invoked with `ppfa -p/root/abc/def.xyz.nnn infile`, it generates page definition `/root/abc/def.xyz/name.nnn`. In another example, if you enter:

```
ppfa -p abc/def.xyz input.file
```

and it has a PAGEDEF statement in the source, then the page definition created will be `abc/def/P1NAME.xyz` or `./abc/def/P1NAME.xyz`.

However, if you enter:

```
ppfa -p abc/def.xyz input.file
```

PPFA generates the file `abc/def/P1name.xyz`, not `./abc/def/P1name.xyz`.

- `x` Causes `ppfa` to interpret information found in columns 1-72 of the `inputfile`. The information in the rest of the columns will be ignored. This is useful if you are downloading a Fixed-80 file from the host.
Examples

1. To create a form definition from an input file called `johnb` in the current library containing the PPFA source statements, enter:

   ```
   ppfa johnb
   ```

   The generated form definition is stored in the current library.

2. To create a form definition from an input file called `maryc` containing the PPFA source statements, and then storing the generated form definition in the `/usr/lpp/resources` library, enter:

   ```
   ppfa -f/usr/lpp/resources maryc
   ```

Implementation Specifics

This command is part of PPFA, and it is installed with the `psf.ppfa` option for PSF for AIX; it is installed with `ipr.ppfa.rte` for InfoPrint Manager for AIX.

Files

- `/usr/lpp/ppfa/bin/ppfa`
  PPFA program
- `/usr/lpp/ppfa/samples`
  Source code for the form definitions and page definitions supplied with PSF for AIX
- `/usr/lpp/pst/ppfa`
  Source code for the form definitions and page definitions supplied with InfoPrint Manager for AIX

OS/400 Environment

Support was added in OS/400, Version 2, Release 1, for line data, mixed data, page definitions, and form definitions for spooled files networked through NJE or PrintManager/400. The support was enhanced in OS/400, Version 3, Release 2 and Release 7, allowing S/370 users to more easily send data to AS/400, and to give OS/400 users full AFP function.

Note: In this document, the following terms are used interchangeably:

- *LINE and line data*
- *AFPDSLINE and mixed data*
- `PAGDFN` and page definition
- `FORMDF` and form definition

This section describes the following printer file attributes:

- `DEVTYPE`
- `CTLCHAR`
- `TBLREFCHR`
- `AFPCHARS`
- `PAGDFN`
- `FORMDF`

This section also provides information about line data application considerations, device type considerations, AS/400 printer file parameters, carriage control characters (ANSI and Machine), Table Reference Characters, IGC parameters,
Medium-Map-Name (INVMMAP) DDS keyword, restrictions when using PAGDFN and FORMDF, as well as information about the CVTPPFASRC command.

**DEVTYPE Values**

To place line data or mixed data onto the printer spool, specify either the *LINE or *AFPDSLINE values with the DEVTYPE parameter for the CRTPRTF, CHGPRTF, and OVRPRTF CL commands.

*LINE  Line data is placed onto the spool. For *LINE, specify any of the following:
  - CTLCHAR(*FCFC)
  - CTLCHAR(*MACHINE)
  - CTLCHAR(*NONE)

  To place line data and skipping or spacing controls directly onto the printer spool without converting it to another data stream, specify *LINE. The line data is not in a printer-ready format and, like AFPDS, will be converted to the appropriate printer format at print time.

*AFPDSLINE  Line data and AFPDS (mixed data) is placed onto the printer spool. For *AFPDSLINE, specify any of the following:
  - CTLCHAR(*FCFC)
  - CTLCHAR(*MACHINE)

  You can specify page definitions to format traditional application line data without making any application programming changes. If, however, you want to use any one of the following AFPDS structured fields (which can be intermixed with line data), you must specify, in hex, a X'5A' record in the output buffer. Refer to Advanced Function Printing: Programming Guide and Line Data Reference, (S544-3884), for more information about mixed documents and MO:DCA.

    - Invoke Data Map
    - Invoke Medium Map
    - Invoke Page Segment
    - Include Page Overlay
    - Presentation Text

  Refer to the Advanced Function Presentation Programming Guide and Line Data Reference (S544-3884) for more information about mixed data streams.

**CTLCHAR Values**

When machine code control characters exist in the data (rather than ANSI control characters), specify the CTLCHAR parameter with a value of *MACHINE on the CRTPRTF, CHGPRTF, or OVRPRTF CL commands.

See Table 9 on page 211 for information about machine code control characters.
TBLREFCHR Parameter

To indicate whether a table reference character (TRC) exists in the data, specify the TBLREFCHR parameter with the CRTPRTF, CHGPRTF, or OVRPRTF CL commands.

If forms control characters are used with the data, the TRC follows the forms control character, but precedes the data bytes. If forms control characters are not used, the TRC is the first byte of the data record. As with forms control characters, if table reference characters are used, every data record must contain a TRC byte.

Note: The TBLREFCHR parameter is ignored if specified for *USERASCII, *SCS, *IPDS, and *AFPDS device types.

See “Table Reference Characters (TRC)” on page 212 for more information.

AFPCHARS Parameter

To identify up to four 4-byte names of coded fonts, specify the AFPCHARS parameter with the CRTPRTF, CHGPRTF, or OVRPRTF CL commands. The 4-byte names are concatenated to ‘X0’ to identify up to four coded fonts that are needed when TBLREFCHR is used within the data.

PAGDFN Parameter

To use or identify a fully-qualified page definition, specify the PAGDFN parameter with the CRTPRTF, CHGPRTF, or OVRPRTF CL commands.

A page definition is an AFP resource object that allows line data and mixed data to be formatted, independent of the application. You can specify page definitions with *LINE or *AFPDSLINE data. After PSF/400 completes formatting, it converts the line data and page definition to IPDS.

Whenever you specify line data or mixed data through NJE or PrintManager/400, with output directed to an AFP printer but do not specify a page definition, an inline page definition is built from the print parameters and passed to PSF/400.

Depending upon the specifications given, some printer file parameters may be ignored when the spooled file is printed through PSF/400. For example, if you specify a page definition on the CRTPRTF, CHGPRTF, or OVRPRTF CL command, and also specify line data or mixed data, an inline page definition will not be built from the printer file parameters. In this case, if you send the data to an AFP printer, PSF/400 ignores the following print parameters:

- CDEFNT
- CHRID
- CPI
- FNTCHRSET
- FOLD
- FONT
- LPI
- MULTIUP
- PAGESIZE
- PAGRTT
- REDUCE
However, in this example, if you send the data to a non-AFP printer with a devtype of "LINE", the page definition parameter is ignored and the print parameters are used. The line data is converted to SCS or IPDS.

If you specify a page definition, but do not specify a form definition, an inline form definition is built from the appropriate printer file parameters, including:

- DRAWER
- DUPLEX
- FORMFEED
- PAGRTT
- PRTQLTY

**FORMDF Parameter**

To use or identify a fully-qualified form definition, specify the FORMDF parameter with the CRTPRTF, CHGPRTF or OVRPRTF CL commands.

A form definition is an AFP resource object that defines the characteristics of the form, including:

- overlays
- position of page data on the form
- rotation
- modification to pages

You can specify a form definition with the following data:

- *AFPDS
- *AFPDSLNE
- *LINE
- *IPDS
- *SCS

PSF/400 accepts *AFPDS, *AFPDSLNE, and *LINE data, and requires a form definition to print. (When you specify an AFP printer, the *IPDS and *SCS data streams are converted to AFPDS.)

Whenever you specify any output directed to an AFP printer but do not specify a form definition, an inline form definition is built from the print parameters and passed to PSF/400.

Depending upon the specifications given, some printer file parameters may be ignored when output is printed through PSF/400. For example, if you specify a form definition on the CRTPRTF, CHGPRTF or OVRPRTF CL command and also specify line data or mixed data, an inline form definition will not be built from the printer file parameters. In this example, if you send the data to an AFP printer PSF/400 ignores the following printer file parameters:

- BACKMGN
- DRAWER
- DUPLEX
- FORMFEED
- FRONTMGN
- MULTIUP (N_UP)
- PAGRTT
- PRTQLTY
• REDUCE (N_UP)

However, if you send the data to a non-AFP printer with a devtype of *IPDS or *SCS in this example, the form definition parameter is ignored and the printer file parameters are used.

When specifying a form definition on the printer file, the values you specify on the DRAWER and DUPLEX parameters will override the drawer and duplex values specified in the form definition. If you want to use the drawer and duplex values specified in the form definition, you must specify DRAWER(*FORMDF) and DUPLEX(*FORMDF) on the printer file.

To indicate an output bin for a form definition, specify the OUTBIN parameter on the printer file.

Application Considerations for Line Data

Line data and mixed data on AS/400 are used by S/370 AFP users who are migrating data to AS/400 and AS/400 users who are generating AFPDS, IPDS, or SCS data streams.

If you are a S/370 user, you should be familiar with the concepts of AFP and page definitions. When using line data, page definitions, and form definitions, your applications are generating line data with either an ANSI or machine code control character in column 1 of the spooled output. To migrate data to AS/400 prior to OS/400 Version 3, Release 2, and Version 3, Release 7, S/370 users had to use PrintManager/400 API calls to rewrite their applications. With line data support, applications do not need to be rewritten. To migrate data to AS/400, specify CTLCHAR(*FCFC) for ANSI code control, or CTLCHAR(*MACHINE) for machine code control. You should also specify DEVTYPE(*LINE) or DEVTYPE(*AFPDSLINE).

If you are an AS/400 user, and want to use line data, page definitions, and form definitions, you must determine whether your application generates an ANSI control character in column 1 of your spooled output.

If your application does generate an ANSI control character in column 1 of your spooled output to control skipping and spacing, specify CTLCHAR(*FCFC) on the printer file. Also, to convert to line data, specify the following in the AS/400 printer file:

• DEVTYPE(*LINE)
• PAGDFN (a page definition)
• FORMDF (a form definition) optional format

Note: Specifying a form definition in this format is optional; you could specify that an inline form definition be built from the print file parameters by indicating FORMDF(*NONE) on the AS/400 printer file.

You can change your application to place an ANSI control character in column 1 of your spooled output to control skipping and spacing, by using a language or application construct (such as a SKIP or SPACE option on a COBOL WRITE statement), or by making an RPG output specification.

When you specify a device type of *AFPDS, *IPDS, or *SCS, control information is used to generate the appropriate skipping or spacing commands in the specified
data stream. The control information for *AFPDS and *LINE that is passed by the compilers and application is converted to a machine code control character. Thus, applications that do not use ANSI control characters can generate line data with control characters onto the spool and use a page definition for post spool formatting, if you specify CTLCHAR(*NONE) and DEVTYPE(*LINE).

**Device Type Considerations**

When using line data, you can specify various combinations of DEVTYPE(*LINE), PAGDFN and FORMDF parameter support on the print file. For example:

- Specify DEVTYPE(*LINE), PAGDFN and FORMDF
  - When you print to an AFP printer, PSF/400 uses the PAGDFN and FORMDF parameters to transform the data to IPDS.
  - When you print to a non-AFP printer, the PAGDFN and FORMDF parameters are ignored. The parameters on the print file are used, and the line data is transformed to IPDS or SCS.

- Specify DEVTYPE(*LINE), no PAGDFN, with FORMDF
  - When you print to an AFP printer, an inline page definition is built from the print file parameters. PSF/400 uses the inline page definition and user-specified FORMDF parameter to transform the data to IPDS.
  - When you print to a non-AFP printer, the FORMDF parameter is ignored. The print file parameters are used, and the line data is transformed to IPDS or SCS.

- Specify DEVTYPE(*LINE), PAGDFN, no FORMDF
  - When you print to an AFP printer, an inline form definition is built from the print file parameters. PSF/400 uses the user-specified PAGDFN parameter and the inline form definition to transform the data to IPDS.
  - When you print to a non-AFP printer, the PAGDFN parameter is ignored. The print file parameters are used, and the line data is transformed to IPDS or SCS.

- Specify DEVTYPE(*LINE), no PAGDFN, no FORMDF
  - When you print to an AFP printer, an inline PAGDFN and FORMDF is built from the print file parameters. PSF/400 uses the inline page definition and inline form definition to transform the data to IPDS.
  - When you print to a non-AFP printer, the print file parameters are used, and the line data is transformed to IPDS or SCS.

The support for combinations of PAGDFN and FORMDF for DEVTYPE(*AFPDSLINE) are similar to DEVTYPE(*LINE) line data. However, you should be aware of the following exceptions:

- When you send data to a non-AFP printer, the data can not be transformed to IPDS or to SCS. The spooled file must be printed on an AFP printer.

- Although you are not required to specify PAGDFN or FORMDF with *AFPDSLINE data, certain AFPDS commands in the data stream (for example, Invoke Medium Map) may reference named structured fields in the...

---

5 You are not required to specify PAGDFN or FORMDF with *AFPDSLINE data because it can be built inline from a print file.
PAGDFN or FORMDF that may not match those in the inline PAGDFN and FORMDF.

The following parameters for line data can be changed with the CHGSPLFA command, after the data is spooled and before the spool file is printed:

- FORMDF
- PAGDFN
- AFPCHARS

**AS/400 Printer File Parameters**

The following table summarizes the print file keyword support provided when line data or mixed data is specified.

**Support of AS/400 printer file parameters**

When line data is redirected to a non-AFP printer, the attribute of the print file is used to print the file.

*Note:* Line data (*LINE) specified with a page definition or mixed data can not be redirected.

The following table summarizes the support for print file keywords when Line data or Mixed data is specified.

<table>
<thead>
<tr>
<th>Print keyword</th>
<th>Line data or Mixed data specified</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFPCHARS</td>
<td>Is supported when printing to an AFP printer. Ignored when line data is redirected to a non-AFP printer.</td>
</tr>
<tr>
<td>ALIGN</td>
<td>Is supported</td>
</tr>
<tr>
<td>CTLCHAR</td>
<td>Is supported</td>
</tr>
<tr>
<td>AUT</td>
<td>Is supported</td>
</tr>
<tr>
<td>BACKMGN</td>
<td>Ignored when a FORMDF is specified and printing to an AFP printer. Margin offset information is specified in the FORMDF. Used when printing to an AFP printer and no FORMDF is specified. Inline FORMDF is built from print parameters. Valid only with *AFPDS data stream.</td>
</tr>
<tr>
<td>BACKOVL</td>
<td>Is supported. Ignored when print file (IPDS) is redirected to a non-AFP printer.</td>
</tr>
<tr>
<td>CDEFNNT</td>
<td>Ignored when a PAGDFN is specified and printing to an AFP printer. Font is gotten from AFPCHARS parameter or is specified in PAGDFN. Used when printing to an AFP printer and no PAGDFN is specified. Inline PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>CHLVAL</td>
<td>Is ignored. The PAGDFN contains information for mapping channel numbers to line numbers.</td>
</tr>
<tr>
<td>Print keyword</td>
<td>Line data or Mixed data specified</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>CHRID</td>
<td>Ignored when a PAGDFN is specified and printing to an AFP printer. Code page is gotten from AFPCHARS parameter or is specified in PAGDFN. Used when printing to an AFP printer and no PAGDFN is specified. Inline PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>COPIES</td>
<td>Is supported</td>
</tr>
<tr>
<td>CPI</td>
<td>Ignored when a PAGDFN is specified and printing to an AFP printer. Font is gotten from AFPCHARS parameter or is specified in PAGDFN. Used when printing to an AFP printer and no PAGDFN is specified. Inline PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>CTLCHAR</td>
<td>Is supported</td>
</tr>
<tr>
<td>DEV</td>
<td>Is supported</td>
</tr>
<tr>
<td>DEVTYPET</td>
<td>*LINE and *AFPDSLINE only</td>
</tr>
<tr>
<td>DFRWRT</td>
<td>Is supported</td>
</tr>
<tr>
<td>DRAWER</td>
<td>Ignored when a FORMDF is specified and printing to an AFP printer. Drawer is gotten from FORMDF. Used when printing to an AFP printer and no FORMDF is specified. Inline FORMDF is built from print parameters (for SCS or IPDS data streams).</td>
</tr>
<tr>
<td>DUPLEX</td>
<td>Ignored when a FORMDF is specified and printing to an AFP printer. Duplex is gotten from FORMDF. Used when printing to an AFP printer and no FORMDF is specified. Inline FORMDF is built from print parameters (for SCS or IPDS data streams).</td>
</tr>
<tr>
<td>FIDELITY</td>
<td>Is supported when printing to an AFP printer. Ignored when line data is redirected to a non-AFP printer. Default is content fidelity.</td>
</tr>
<tr>
<td>FILE</td>
<td>Is supported</td>
</tr>
<tr>
<td>FILESEP</td>
<td>Is supported</td>
</tr>
<tr>
<td>FNTCHRSET</td>
<td>Ignored when a PAGDFN is specified and printing to an AFP printer. Font is gotten from AFPCHARS parameter or is specified in PAGDFN. Used when printing to an AFP printer and no PAGDFN is specified. Inline PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>FOLD</td>
<td>Is ignored if a PAGDFN or FORMDF is specified and printing to an AFP printer because when PAGDFN or FORMDF is specified, the page size and where data is positioned on a page is unknown.</td>
</tr>
<tr>
<td>FONT</td>
<td>Ignored when a PAGDFN is specified and printing to an AFP printer. Font is gotten from AFPCHARS parameter or is specified in PAGDFN. Used when printing to an AFP printer and no PAGDFN is specified. Inline PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>Print keyword</td>
<td>Line data or Mixed data specified</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>FORMDF</td>
<td>Is supported when printing to an AFP printer. Ignore when line data is redirected to a non-AFP printer.</td>
</tr>
<tr>
<td>FORMFEED</td>
<td>Ignored when a FORMDF is specified and printing to an AFP printer. FORMFEED is gotten from FORMDF. Used when printing to an AFP printer and no FORMDF is specified. Inline FORMDF is built from print parameters.</td>
</tr>
<tr>
<td>FORMTYPE</td>
<td>Is supported</td>
</tr>
<tr>
<td>FRONTMGN</td>
<td>Ignored when a FORMDF is specified and printing to an AFP printer. Margin offset information is specified in the FORMDF. Used when printing to an AFP printer and no FORMDF is specified. Inline FORMDF is built from print parameters. Valid only with *AFPDS data stream.</td>
</tr>
<tr>
<td>FRONTOVL</td>
<td>Is supported. Ignore when print file (IPDS) is redirected to a non-AFP printer.</td>
</tr>
<tr>
<td>HOLD</td>
<td>Is supported</td>
</tr>
<tr>
<td>LPI</td>
<td>Ignored when a PAGDFN is specified and printing to an AFP printer. Lines Per Inch is gotten from PAGDFN. Used when printing to an AFP printer and no PAGDFN is specified. Inline PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>LVLCHK</td>
<td>Ignored. Line data and Mixed data are not valid with a DDS print file</td>
</tr>
<tr>
<td>MAXRCDS</td>
<td>Is supported</td>
</tr>
<tr>
<td>MULTIUP</td>
<td>Ignored when a PAGDFN or FORMDF is specified and printing to an AFP printer. Multiup is specified in the FORMDF(N-UP) or can be achieved in the PAGDFN. Used when printing to an AFP printer and no FORMDF or PAGDFN is specified. Inline FORMDF and PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>OPTION</td>
<td>Is supported</td>
</tr>
<tr>
<td>OUTBIN</td>
<td>Is supported when printing to an AFP printer. Ignore when line data is redirected to a non-AFP printer</td>
</tr>
<tr>
<td>OUTPTY</td>
<td>Is supported</td>
</tr>
<tr>
<td>OUTQ</td>
<td>Is supported</td>
</tr>
<tr>
<td>OVRFLW</td>
<td>The overflow message will only be issued for Line data. It will not be issued for Mixed data.</td>
</tr>
<tr>
<td>PAGDFN</td>
<td>Is supported when printing to an AFP printer. Ignore when line data is redirected to a non-AFP printer.</td>
</tr>
<tr>
<td>PAGERANGE</td>
<td>Is supported</td>
</tr>
<tr>
<td>PAGESIZE</td>
<td>Ignored when a PAGDFN is specified and printing to an AFP printer. Page size is gotten from PAGDFN. Used when printing to an AFP printer and no PAGDFN is specified. Inline PAGDFN is built from print parameters.</td>
</tr>
<tr>
<td>Print keyword</td>
<td>Line data or Mixed data specified</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td>PAGRTT</td>
<td>Ignored when a FORMDF is specified and printing to an AFP printer. Rotation is specified in FORMDF. Used when printing to an AFP printer and no FORMDF is specified. Inline FORMDF is built from print parameters (for SCS or IPDS data streams). IF PAGRTT(*COR) is specified, then this parameter is ignored unless both PAGDFN and FORMDF are not specified.</td>
</tr>
<tr>
<td>PRTQLTY</td>
<td>Ignored when a FORMDF is specified and printing to an AFP printer. Print quality is gotten from FORMDF. Used when printing to an AFP printer and no FORMDF is specified. Inline FORMDF is built from print parameters.</td>
</tr>
<tr>
<td>PRTTXT</td>
<td>If the file is opened *LINE and no PAGDFN is specified, print text will be supported. If the file is opened as *AFPDSLLINE, print text will not be supported.</td>
</tr>
<tr>
<td>REDUCE</td>
<td>Ignored when a PAGDFN or FORMDF is specified and printing to an AFP printer. Multiup is specified in the FORMDF(N-UP) or can be achieved in the PAGDFN. Used when printing to an AFP printer and no FORMDF or PAGDFN is specified. Inline FORMDF and PAGDFN is built from print parameters. REDUCE(*TEXT) is supported when printing to a non-AFP printer.</td>
</tr>
<tr>
<td>REPLACE</td>
<td>Is supported</td>
</tr>
<tr>
<td>RPLUNPRT</td>
<td>Is supported</td>
</tr>
<tr>
<td>SAVE</td>
<td>Is supported</td>
</tr>
<tr>
<td>SCHEDULE</td>
<td>Is supported</td>
</tr>
<tr>
<td>SCRFILE</td>
<td>Ignored. Line data and mixed data are not valid with a DDS print file.</td>
</tr>
<tr>
<td>SCRMBR</td>
<td>Ignored. Line data and mixed data are not valid with a DDS print file.</td>
</tr>
<tr>
<td>SHARE</td>
<td>Is supported</td>
</tr>
<tr>
<td>SPOOL</td>
<td>*YES only. Direct IO (SPOOL(*NO)) is not supported.</td>
</tr>
<tr>
<td>TBLREFCHR</td>
<td>Is supported when printing to an AFP printer. Ignored when line data is redirected to a non-AFP printer (IPDS).</td>
</tr>
<tr>
<td>TEXT</td>
<td>Is supported</td>
</tr>
<tr>
<td>UOM</td>
<td>Is supported</td>
</tr>
<tr>
<td>USRDFNDTA</td>
<td>Is supported when printing to an AFP printer.</td>
</tr>
<tr>
<td>USRDFNOBJ</td>
<td>Is supported when printing to an AFP printer.</td>
</tr>
<tr>
<td>USRDFNOPT</td>
<td>Is supported when printing to an AFP printer.</td>
</tr>
<tr>
<td>USRDTA</td>
<td>Is supported</td>
</tr>
<tr>
<td>WAITFILE</td>
<td>Is supported</td>
</tr>
</tbody>
</table>
Carriage Control Characters

The carriage control character can be represented as either ANSI or machine code.

ANSI carriage control is a standard representation that is used with printers from many different manufacturers. Table 8 lists the ANSI codes and their functions. Machine code control characters were defined by IBM; they correspond to channel command words issued by the operating system. Table 9 on page 211 lists the IBM machine code values and functions.

**Note:** You may not use both ANSI and machine codes within a single data set.

There are differences in the conventions used by OS/400 for ANSI and machine code line spacing. The OS/400 convention for ANSI handles line spacing and then causes the line to be printed. The OS/400 convention for machine codes causes the line to be printed and then the spacing action is performed.

### ANSI Carriage Control Characters

**Table 8. ANSI Carriage Control Characters**

<table>
<thead>
<tr>
<th>Hexadecimal Value</th>
<th>Character</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'40'</td>
<td>(space)</td>
<td>Space 1 line, then print (single spacing)</td>
</tr>
<tr>
<td>X'F0'</td>
<td>0</td>
<td>Space 2 lines, then print (double spacing)</td>
</tr>
<tr>
<td>X'60'</td>
<td>-</td>
<td>Space 3 lines, then print (triple spacing)</td>
</tr>
<tr>
<td>X'4E'</td>
<td>+</td>
<td>Suppress spacing, then print (overstrike previous line)</td>
</tr>
<tr>
<td>X'F1'</td>
<td>1</td>
<td>Print the data at line position defined as Channel 1 (by convention, the first line on a new page)</td>
</tr>
<tr>
<td>X'F2'</td>
<td>2</td>
<td>Print the data at the line position defined as Channel 2</td>
</tr>
<tr>
<td>X'F3'</td>
<td>3</td>
<td>Print the data at the line position defined as Channel 3</td>
</tr>
<tr>
<td>X'F4'</td>
<td>4</td>
<td>Print the data at the line position defined as Channel 4</td>
</tr>
<tr>
<td>X'F5'</td>
<td>5</td>
<td>Print the data at the line position defined as Channel 5</td>
</tr>
<tr>
<td>X'F6'</td>
<td>6</td>
<td>Print the data at the line position defined as Channel 6</td>
</tr>
<tr>
<td>X'F7'</td>
<td>7</td>
<td>Print the data at the line position defined as Channel 7</td>
</tr>
<tr>
<td>X'F8'</td>
<td>8</td>
<td>Print the data at the line position defined as Channel 8</td>
</tr>
<tr>
<td>X'F9'</td>
<td>9</td>
<td>Print the data at the line position defined as Channel 9</td>
</tr>
<tr>
<td>X'C1'</td>
<td>A</td>
<td>Print the data at the line position defined as Channel 10</td>
</tr>
<tr>
<td>X'C2'</td>
<td>B</td>
<td>Print the data at the line position defined as Channel 11</td>
</tr>
<tr>
<td>X'C3'</td>
<td>C</td>
<td>Print the data at the line position defined as Channel 12</td>
</tr>
</tbody>
</table>

**Note:** When ANSI carriage controls are used, only the values that appear in this table are considered valid by PSF/400. PSF/400 treats any other ANSI carriage control value as invalid and prints any data on the line using single spacing.
### Machine Carriage Control Characters

<table>
<thead>
<tr>
<th>Control Character Value (in hexadecimal)</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>X'03'</td>
<td>No operation</td>
</tr>
<tr>
<td>X'09'</td>
<td>Print and space 1 line (single spacing)</td>
</tr>
<tr>
<td>X'11'</td>
<td>Print and space 2 lines (double spacing)</td>
</tr>
<tr>
<td>X'19'</td>
<td>Print and space 3 lines (triple spacing)</td>
</tr>
<tr>
<td>X'01'</td>
<td>Print without spacing (overstrike next line)</td>
</tr>
<tr>
<td>X'89'</td>
<td>Print the data, then skip to the line position defined as Channel 1 (by convention, the first line on a new page)</td>
</tr>
<tr>
<td>X'91'</td>
<td>Print the data, then skip to the line position defined as Channel 2</td>
</tr>
<tr>
<td>X'99'</td>
<td>Print the data, then skip to the line position defined as Channel 3</td>
</tr>
<tr>
<td>X'A1'</td>
<td>Print the data, then skip to the line position defined as Channel 4</td>
</tr>
<tr>
<td>X'A9'</td>
<td>Print the data, then skip to the line position defined as Channel 5</td>
</tr>
<tr>
<td>X'B1'</td>
<td>Print the data, then skip to the line position defined as Channel 6</td>
</tr>
<tr>
<td>X'B9'</td>
<td>Print the data, then skip to the line position defined as Channel 7</td>
</tr>
<tr>
<td>X'C1'</td>
<td>Print the data, then skip to the line position defined as Channel 8</td>
</tr>
<tr>
<td>X'C9'</td>
<td>Print the data, then skip to the line position defined as Channel 9</td>
</tr>
<tr>
<td>X'D1'</td>
<td>Print the data, then skip to the line position defined as Channel 10</td>
</tr>
<tr>
<td>X'D9'</td>
<td>Print the data, then skip to the line position defined as Channel 11</td>
</tr>
<tr>
<td>X'E1'</td>
<td>Print the data, then skip to the line position defined as Channel 12</td>
</tr>
<tr>
<td>X'0B'</td>
<td>Space 1 line without printing</td>
</tr>
<tr>
<td>X'13'</td>
<td>Space 2 lines without printing</td>
</tr>
<tr>
<td>X'1B'</td>
<td>Space 3 lines without printing</td>
</tr>
<tr>
<td>X'8B'</td>
<td>Skip to Channel 1 immediate (by convention, the first line on a new page)</td>
</tr>
<tr>
<td>X'93'</td>
<td>Skip to the Channel 2 position immediate</td>
</tr>
<tr>
<td>X'9B'</td>
<td>Skip to the Channel 3 position immediate</td>
</tr>
<tr>
<td>X'A3'</td>
<td>Skip to the Channel 4 position immediate</td>
</tr>
<tr>
<td>X'AB'</td>
<td>Skip to the Channel 5 position immediate</td>
</tr>
<tr>
<td>X'B3'</td>
<td>Skip to the Channel 6 position immediate</td>
</tr>
<tr>
<td>X'BB'</td>
<td>Skip to the Channel 7 position immediate</td>
</tr>
<tr>
<td>X'C3'</td>
<td>Skip to the Channel 8 position immediate</td>
</tr>
<tr>
<td>X'CB'</td>
<td>Skip to the Channel 9 position immediate</td>
</tr>
<tr>
<td>X'D3'</td>
<td>Skip to the Channel 10 position immediate</td>
</tr>
<tr>
<td>X'DB'</td>
<td>Skip to the Channel 11 position immediate</td>
</tr>
<tr>
<td>X'E3'</td>
<td>Skip to the Channel 12 position immediate</td>
</tr>
</tbody>
</table>
Table 9 (Page 2 of 2). Machine Code Control Characters

<table>
<thead>
<tr>
<th>Control Character Value (in hexadecimal)</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Note: PSF/400 ignores the following hexadecimal machine-code carriage control characters and does not print lines containing them: X'02', through X'07', X'0A', X'12', X'23', X'43', X'63', X'6B', X'73', X'7B', X'EB', X'F3', and X'FB'. PSF/400 treats any other carriage control value as invalid and prints any data on the line using single spacing.</td>
<td></td>
</tr>
</tbody>
</table>

Table Reference Characters (TRC)

Table Reference Characters (TRCs) allow an additional byte to appear at the beginning of a line to indicate which one of up to four different character arrangement tables (coded fonts specified by AFPCHARS parameter) will be used to print the line. This byte, the table reference character contains a value of X'F0', X'F1', X'F2', or X'F3', corresponding to the relative position of the desired coded font in the list of coded fonts specified by the AFPCHARS parameter. If carriage control bytes are used with the data, the table reference character follows the carriage control byte but precedes the data bytes. If a carriage control byte is not used, the table reference character is the first byte of the data record. As with carriage control, if table reference characters are used, every data record must contain a TRC byte.

Figure 72 summarizes the valid forms of line data.

A. Simple data line

D A T A

B. Data line with carriage control byte

CC D A T A

C. Data line with table reference character

TRC D A T A

D. Data line with carriage control byte and table reference character

CC TRC D A T A

Figure 72. Valid Line Data Records
IGC Parameters

The IGC parameters of an AS/400 printer file are described here.

**IGCDTA**
Indicates IGC data may be used in the file. The user for a line or mixed data file will need to indicate that there is SO/SI present in the data by setting **IGCDTA** to **YES**.

**IGCCPI**
For AFP printers, this parameter is ignored, as the pitch of the DBCS data is determined by the selected font.
For non-AFP printers, when line data is transformed to SCS, this parameter is used to specify the pitch of the DBCS data. DBCS SO/SI can not be transformed when going to an IPDS printer.
Mixed data can not be transformed when going to a SCS or IPDS printer.

**IGCSOSI**
This keyword indicates what action should be taken when SO/SI are found in the data. If the data is mixed, the SO/SI should be taken out and appropriate spaces inserted based on the value of this keyword.

*YES
   The SO/SI characters will be printed as blanks.

*NO
   The system does not print the shift control characters. These characters do not occupy a position on the printed output.

*RIGHT
   The system prints two blanks when printing the shift-in characters but does not print shift-out characters.

**IGCEXNCHR**
Ignored, as extension character processing only applies to SCS DBCS printer, not AFP attached printers.

**IGCCHRRRTT**
For AFP printers, this parameter is ignored. Character rotation can be specified in the PAGDFN.
For non-AFP printers, when line data is transformed to SCS, this parameter is used to rotate the DBCS data.

INVMMAP (Medium-Map-Name) DDS Keyword

**INVMMAP** is a record level keyword in DDS used to invoke a medium map. Invoke Medium Map (IMM) specifies the name of a medium map in a form definition. Use the **IMM** in the form definition to select or change print parameters such as input drawer, page rotation, overlays.

The medium map name is limited to 8 characters. You can specify the medium map name as a constant or a program-to-system field.

- medium-map-name
- field1

The **INVMMAP** keyword is valid only with **DEVTYPE(‘AFPDS)**. Also, a form definition must be specified on the print file. If **DEVTYPE** is changed to anything other than **‘AFPDS**, the **INVMMAP** keyword is ignored and a warning message will be issued at print time.
PSF/400 ends printing on the current sheet when a invoke medium map is encountered.

You may not specify `INVMMAP` at the same level as the following keywords:

- `SKIPB`
- `SPACEB`
- `SPACEA`
- `SKIPA`
- `DRAWER`
- `PAGRTT`

The medium map specified remains in effect for the rest of the file unless changed by another `INVMMAP` keyword.

The invoke medium map keyword is validated at print time. An error message will be issued if it is not valid.

Option indicators are valid for the `INVMMAP` keyword.

Figure 73 shows how to specify the `INVMMAP` keyword.

![Figure 73. Specifying the INVMMAP keyword](image)

If indicator 02 is on, RECORD1 uses a new medium map (MAP1).

RECORD2 allows the application program to specify the name of medium map by setting program variable MAP.

Restrictions When Using PAGDFN and FORMDF

Line data and mixed data device types are not supported for externally described printer files (DDS SUPPORT).

Device type `*LINE` or `*AFPDSLNE` and `SPOOL(*NO)` will not be allowed on the `CRTPTF` or `CHGPTF` commands. If this condition exists, Message CPD7341, indicating the value `SPOOL(*NO)` is not valid with device type, will be issued.

Today when native application output is printed, an inline `FORMDF` is generated by the OS/400 which utilizes the printer file parameters for drawer, duplex, outbin, etc. When users specify a form definition on their printer file, this will no longer be done. The form definition will be used for all media handling and the printer file parameters for this information will be ignored. The first media map in the `FORMDF` will be used unless the data itself calls out a different media map (such as with the new DDS keyword).
Native applications which write normal control list-type data (such as RPG and COBOL applications) without utilizing the CTLCHAR parameter of the printer file, will generate *MACHINE control character line data. There will be no support for generating ANSI control character line data except by the application actually writing that type of data using CTLCHAR(*FCFC).

**CVTPPFASRC Command (AFP PrintSuite-dependent)**

The CVTPPFASRC command creates a data base file member from the information contained in the library and file containing the PPFA source. The PPFA source file determines the name of the output data base file members that is used as input to the CRTFORMDF or CRTPAGDFN command.

If you want to create an object of type *FORMDF, use the CRTFORMDF command.

If you want to create an object of type *PAGDFN, use the CRTPAGDFN command.

**Note:** Form definitions and page definitions are limited to AFP printing with PSF/400.

**Syntax**

OS/400 is path-dependent; therefore, sequence and completeness of syntax is very important. The CVTPPFASRC command has two required parameters (FILE and MBR) and four subcommands (FORMDFFILE, PAGDFNFILE, OUTPUT, and OPTION).

```plaintext
CVTPPFASRC
   FILE ( *LIBL / input-PPFA-source-file-name
          | *CURLIB / input-PPFA-source-file-name
          | library-name / input-PPFA-source-file-name )
   MBR ( input-member -name )
   [ FORMDFFILE ( *NONE
                | *LIBL / form-definition-file
                | *CURLIB / form-definition-file
                | library-name / form-definition-file ) ]
   [ PAGDFNFILE ( *NONE
                | *LIBL / page-definition-file
                | *CURLIB / page-definition-file
                | library-name / page-definition-file ) ]
   [ OUTPUT ( *PRINT
             | *NONE ) ]
   [ OPTION ( { *SRC | *NOSRC }
             | { *SECLVL | *NOSECLVL } ) ]
```

*Figure 74. Specifying the CVTPPFASRC command (AFP PrintSuite-dependent)*

**Note:** *NONE may be specified for either FORMDFFILE or PAGDFNFILE, but not for both.
Subcommands and Parameters

The Convert PPFA Source (CVTPPFASRC) command creates a data base file member from the information contained in the PPFA source file. The data base file member can then be used either as input to the CRTFORMDF command to create an object of type *FORMDF or as input to the CRTPAGDFN command to create an object of type *PAGDFN.

The following parameters are required:

**FILE** Specifies the location of the PPFA source file and library. The possible library values are:

- ***LIBL** Specifies the PPFA source file is in the library list.
- ***CURLIB** Specifies the current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used for PPFA source.

**library name** Specifies the PPFA source file is in a specific library.

**File Name** File name of the PPFA source file

**MBR** Specifies the PPFA source file member.

**Member Name** Specifies the PPFA source member.

The following parameters are optional:

**FORMDFFILE** Specifies the target file for compiled form definitions.

- ***LIBL** The PPFA form definition output file is in the library list.
- ***CURLIB** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used for PPFA form definition output.
- ***FILE** The PPFA form definition output is the same as the source file.

**library name** The PPFA form definition output is to go to a specific library.

**File Name** The file name in which to place the form definition output of the PPFA.

**PAGDFNFILE** Specifies the target file for compiled form definitions.

- ***LIBL** The PPFA page definition output file is in the library list.
- ***CURLIB** The current library for the job is searched. If no library is specified as the current library for the job, the QGPL library is used for PPFA page definition output.
- ***FILE** The PPFA page definition output is the same as the source file.

**library name** The PPFA page definition output is to go to a specific library.
File Name The file name in which to place the page definition output of the PPFA.

OUTPUT Specifies the destination of the listing file and error messages that PPFA produces during compilation.

*PRINT The listing file is sent to the system queue for printing.

*NONE The listing file is not generated.

OPTION Specifies the type of output produced when the file is created.

*SRC A printout of the source statements, including a list of errors is created.

*NOSRC No printout of the source statements is created unless errors are detected. If errors are detected, they are listed along with the sequence number that caused the error.

*SECLVL The online help information appears in the PPFA printout.

*NOSECLVL The messages section of the PPFA printout does not contain the online help information for messages issued during PPFA processing.
Appendix B. More about Direction

In PPFA, directions specified with the PRINTLINE and TRCREF commands are relative to the direction specified in the PAGEFORMAT command. If no PAGEFORMAT command has been specified, the direction specified in the PAGEDEF command is used. If no direction has been specified in either of these commands, the default direction for the page format is ACROSS.

The PRINTLINE and TRCREF commands add their DIRECTION values to the DIRECTION value specified with the PAGEFORMAT command. Thus, you may select a PAGEFORMAT direction and code PRINTLINEs and TRCREFs relative to the PAGEFORMAT direction. For more information about the PRINTLINE and TRCREF commands, see Chapter 3, “Using Page Definition Commands” on page 35.

For instance, if a page is to be printed in the landscape page presentation on a printer that requires the DOWN or UP print direction to generate landscape output, the PAGEFORMAT command can specify DOWN as its DIRECTION. Once this direction is established, you can view the page as a landscape page and specify the PRINTLINE and the TRCREF commands with the ACROSS direction. Output specified in this way prints ACROSS relative to the landscape page, as shown in Figure 75.

![Figure 75. Printing Across a Landscape Page](image)

Note that if you specify the DOWN direction for the PRINTLINE or the TRCREF command in this case, the output looks like Figure 76 because the direction of the page format is also DOWN.

![Figure 76. Printing Down a Portrait Page](image)
Table 10 on page 220 shows the final result when all of the possible combinations of DIRECTION are specified. The final direction that PPFA computes from the PAGEFORMAT, PRINTLINE, and TRCREF commands determines the prefix assigned to the font names specified in the page definition. The final direction is particularly important when printing on the 3800 printer because its unbounded-box font architecture requires a separate font for each combination of print direction and character rotation. This information is encoded in the prefix of the font name (X1, X3,XA, and XF, for example).

<table>
<thead>
<tr>
<th>Page Format</th>
<th>PRINTLINE or TRCREF</th>
<th>Final Result</th>
<th>3800 Font Prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Across</td>
<td>Across</td>
<td>Across</td>
<td>X1 X5 X9 XD</td>
</tr>
<tr>
<td>Across</td>
<td>Down</td>
<td>Down</td>
<td>X2 X6 XA XE</td>
</tr>
<tr>
<td>Across</td>
<td>Back</td>
<td>Back</td>
<td>X3 X7 XB XF</td>
</tr>
<tr>
<td>Across</td>
<td>Up</td>
<td>Up</td>
<td>X4 X8 XC XG</td>
</tr>
<tr>
<td>Down</td>
<td>Across</td>
<td>Down</td>
<td>X2 X6 XA XE</td>
</tr>
<tr>
<td>Down</td>
<td>Down</td>
<td>Back</td>
<td>X3 X7 XB XF</td>
</tr>
<tr>
<td>Down</td>
<td>Back</td>
<td>Up</td>
<td>X4 X8 XC XG</td>
</tr>
<tr>
<td>Down</td>
<td>Up</td>
<td>Across</td>
<td>X1 X5 X9 XD</td>
</tr>
<tr>
<td>Back</td>
<td>Across</td>
<td>Back</td>
<td>X3 X7 XB XF</td>
</tr>
<tr>
<td>Back</td>
<td>Down</td>
<td>Up</td>
<td>X4 X8 XC XG</td>
</tr>
<tr>
<td>Back</td>
<td>Back</td>
<td>Across</td>
<td>X1 X5 X9 XD</td>
</tr>
<tr>
<td>Back</td>
<td>Up</td>
<td>Down</td>
<td>X2 X6 XA XE</td>
</tr>
<tr>
<td>Up</td>
<td>Across</td>
<td>Up</td>
<td>X4 X8 XC XG</td>
</tr>
<tr>
<td>Up</td>
<td>Down</td>
<td>Across</td>
<td>X1 X5 X9 XD</td>
</tr>
<tr>
<td>Up</td>
<td>Back</td>
<td>Down</td>
<td>X2 X6 XA XE</td>
</tr>
<tr>
<td>Up</td>
<td>Up</td>
<td>Back</td>
<td>X3 X7 XB XF</td>
</tr>
</tbody>
</table>

The entries in the **Final Result** column can be computed using a simple algorithm. If you assume that ACROSS is 0, DOWN is 1, BACK is 2, and UP is 3, you can add the direction specifications in the two commands, subtracting 4 when the result is 4 or greater, to compute the final direction.
Appendix C. Differences in Measurements and REPEATs with AFP Utilities

When repeating a DRAWRULE (OGL), PRINTLINE (PPFA), or “Line” (PMF), there are differences in the measurements of the repeated lines. For OGL, REPEAT indicates the number of repetitions in addition to the first. Therefore, REPEAT 1 yields 2 DRAWRULEs. For PPFA, REPEAT indicates the total number of PRINTLINEs. Therefore, REPEAT 2 yields 2 PRINTLINEs.

Another difference occurs when the linespacing (set by SETUNITS in OGL and PPFA, and by a screen item in PMF) results in the distance from one line to the next not being a whole number of pels. Each product handles the fractional pel differently. Because the printer cannot print parts of a pel, fractional pels cannot be represented at the printer. When linespacing calculations result in a fractional pel per linespace, the following occurs:

OGL Carries the fractions until they add up to a whole pel, then adds it in. This results in the final spot of a repeat being within a pel of where it is expected. Therefore, not all of the spaces between rules are even; they can vary by one pel.

PPFA Truncates the fractional pel prior to the repeat. Therefore, the spaces between the lines are even, but the total might be shorter than expected.

PMF Rounds the fractional pel prior to the repeat. Therefore, the spaces between the lines are even, but the total might be shorter or longer than expected. If the fractional pel is less than 0.5, it is handled the same as PPFA and the linespace will be shorter. If the fractional pel is greater than or equal to 0.5, the linespace will be longer.

Use linespacing in all products that result in a whole number of pels. To resolve existing problems, select the resource that you don't want to change, and code the remaining resource without using REPEAT because of the way the other products handle the fractional pels.

For example, if you want to print at 9 lines per inch, and repeat this for 20 lines, the following will occur. Starting at zero, and adding 9 lines per inch (converted to pels this is 240/9 = 26.6670), you will see the results illustrated in Table 11 on page 222.
Table 11. Differences in Measurements and REPEATs with AFP Utilities

<table>
<thead>
<tr>
<th>Repetition</th>
<th>Mathematics</th>
<th>OGL</th>
<th>PPFA</th>
<th>PMF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Position</td>
<td>FromLast</td>
<td>Position</td>
<td>FromLast</td>
</tr>
<tr>
<td>0.000</td>
<td>0.000</td>
<td>-.---</td>
<td>0.000</td>
<td>-.---</td>
</tr>
<tr>
<td>3</td>
<td>80.000</td>
<td>80.000</td>
<td>80.000</td>
<td>80.000</td>
</tr>
<tr>
<td>4</td>
<td>106.667</td>
<td>106.667</td>
<td>106.667</td>
<td>106.667</td>
</tr>
<tr>
<td>5</td>
<td>133.333</td>
<td>133.333</td>
<td>133.333</td>
<td>133.333</td>
</tr>
<tr>
<td>6</td>
<td>160.000</td>
<td>160.000</td>
<td>160.000</td>
<td>160.000</td>
</tr>
<tr>
<td>7</td>
<td>186.667</td>
<td>186.667</td>
<td>186.667</td>
<td>186.667</td>
</tr>
<tr>
<td>8</td>
<td>213.333</td>
<td>213.333</td>
<td>213.333</td>
<td>213.333</td>
</tr>
<tr>
<td>9</td>
<td>240.000</td>
<td>240.000</td>
<td>240.000</td>
<td>240.000</td>
</tr>
<tr>
<td>10</td>
<td>266.667</td>
<td>266.667</td>
<td>266.667</td>
<td>266.667</td>
</tr>
<tr>
<td>11</td>
<td>293.333</td>
<td>293.333</td>
<td>293.333</td>
<td>293.333</td>
</tr>
<tr>
<td>12</td>
<td>320.000</td>
<td>320.000</td>
<td>320.000</td>
<td>320.000</td>
</tr>
<tr>
<td>14</td>
<td>373.333</td>
<td>373.333</td>
<td>373.333</td>
<td>373.333</td>
</tr>
<tr>
<td>15</td>
<td>400.000</td>
<td>400.000</td>
<td>400.000</td>
<td>400.000</td>
</tr>
<tr>
<td>17</td>
<td>453.333</td>
<td>453.333</td>
<td>453.333</td>
<td>453.333</td>
</tr>
<tr>
<td>18</td>
<td>480.000</td>
<td>480.000</td>
<td>480.000</td>
<td>480.000</td>
</tr>
<tr>
<td>19</td>
<td>506.667</td>
<td>506.667</td>
<td>506.667</td>
<td>506.667</td>
</tr>
<tr>
<td>20</td>
<td>533.333</td>
<td>533.333</td>
<td>533.333</td>
<td>533.333</td>
</tr>
</tbody>
</table>

To resolve differences in how OGL, PPFA, and PMF handle repeated values, one of the following approaches may be taken:

- Don't use REPEAT
- Code units as PEL(s)

Note that in all of these products (OGL, PPFA, and PMF), a PEL is 1/240 of an inch.
Appendix D. More About Bar Code Parameters

This section contains supplemental information about Bar Code Object Content Architecture (BCOCA) specified by the BARCODE subcommand of the FIELD command, and includes the following topics:

- Bar code data
- MOD parameter

For more complete information, refer to Data Stream and Object Architectures: Bar Code Object Content Architecture Reference (S544-3766).

Bar Code Data

The data is specified as a series of single-byte code points from a specific code page. Some symbologies limit the valid code points to just the ten numerals (0 through 9), other symbologies allow a richer set of code points. The bar code symbol is produced from these code points; the code points are also used, along with a particular type style, when producing the HRI.

Table 12 lists, for each symbology, the valid code page from which characters are chosen and the type style used when printing HRI in terms of an IBM registered CPGID and FGID. More information about these values can be found in ABOUT TYPE:

IBM's Technical Reference for Core Interchange Digitized Type and in About Type: IBM's Code Pages for Digi tized Type.

<table>
<thead>
<tr>
<th>Type</th>
<th>Bar Code Symbology</th>
<th>EBCDIC-Based CPGID</th>
<th>FGID</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>MSI (modified Plessey code)</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>3</td>
<td>UPC/CGPC — Version A</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>5</td>
<td>UPC/CGPC — Version E</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>6</td>
<td>UPC — Two-digit Supplemental (Periodicals)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>7</td>
<td>UPC — Five-digit Supplemental (Paperbacks)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>8</td>
<td>EAN-8 (includes JAN-short)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>9</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>10</td>
<td>Industrial 2-of-5</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>11</td>
<td>Matrix 2-of-5</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>12</td>
<td>Interleaved 2-of-5, AIM USS-I 2/5</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>13</td>
<td>Codabar, 2-of-7, AIM USS-Codabar</td>
<td>500</td>
<td>Device specific</td>
</tr>
<tr>
<td>17</td>
<td>Code 128, AIM USS-128</td>
<td>1303</td>
<td>Device specific</td>
</tr>
<tr>
<td>22</td>
<td>EAN Two-digit Supplemental</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>23</td>
<td>EAN Five-digit Supplemental</td>
<td>893</td>
<td>3 (OCR-B)</td>
</tr>
<tr>
<td>24</td>
<td>POSTNET</td>
<td>500</td>
<td>None</td>
</tr>
<tr>
<td>26</td>
<td>RM4SCC</td>
<td>500</td>
<td>None</td>
</tr>
</tbody>
</table>
As shown in Table 12, the font used to print HRI depends on the symbology. Some symbologies use OCR-B; others use a device-specific font (usually OCR-A).

Table 13 lists the valid characters for each symbology and specifies how many characters are allowed for a bar code symbol.

<table>
<thead>
<tr>
<th>Code</th>
<th>Bar Code Type</th>
<th>Valid Characters</th>
<th>Valid Data Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Code 39 (3-of-9 Code), AIM USS-39</td>
<td>0123456789 ABCDEFGHIJKLMNOPQRSTUVWXYZ -.$/+% and the space character</td>
<td>Symbology: unlimited BCOCA range: 0 to 50 characters (see note)</td>
</tr>
<tr>
<td>2</td>
<td>MSI (modified Plessey code)</td>
<td>0123456789</td>
<td>3 to 15 characters for Modifier X’01’ 2 to 14 characters for Modifier X’02’ 1 to 13 characters for all other modifiers</td>
</tr>
<tr>
<td>3</td>
<td>UPC/CGPC - Version A</td>
<td>0123456789</td>
<td>11 characters</td>
</tr>
<tr>
<td>5</td>
<td>UPC/CGPC - Version E</td>
<td>0123456789</td>
<td>10 characters</td>
</tr>
<tr>
<td>6</td>
<td>UPC - Two-digit Supplemental (Periodicals)</td>
<td>0123456789</td>
<td>2 characters for Modifier X’00’ 13 characters for Modifier X’01’ 12 characters for Modifier X’02’</td>
</tr>
<tr>
<td>7</td>
<td>UPC - Five-digit Supplemental (Paperbacks)</td>
<td>0123456789</td>
<td>5 characters for Modifier X’00’ 16 characters for Modifier X’01’ 15 characters for Modifier X’02’</td>
</tr>
<tr>
<td>8</td>
<td>EAN-8 (includes JAN-short)</td>
<td>0123456789</td>
<td>7 characters</td>
</tr>
<tr>
<td>9</td>
<td>EAN-13 (includes JAN-standard)</td>
<td>0123456789</td>
<td>12 characters</td>
</tr>
<tr>
<td>10</td>
<td>Industrial 2-of-5</td>
<td>0123456789</td>
<td>Symbology: unlimited BCOCA range: 0 to 50 characters (see note)</td>
</tr>
</tbody>
</table>
### Table 13 (Page 2 of 2). Valid Characters and Data Lengths

<table>
<thead>
<tr>
<th>Code</th>
<th>Bar Code Type</th>
<th>Valid Characters</th>
<th>Valid Data Length</th>
</tr>
</thead>
</table>
| 11   | Matrix 2-of-5 | 0123456789       | Symbology: unlimited  
BCOCA range: 0 to 50 characters (see note) |
| 12   | Interleaved 2-of-5, AIM USS-I 2/5 | 0123456789 | Symbology: unlimited  
BCOCA range: 0 to 50 characters (see note) |
BCOCA range: 0 to 50 characters (see note)  
16 characters plus 4 start/stop characters (ABCD) |
| 17   | Code 128, AIM USS-128 | All characters defined in the Code 128 code page | Symbology: unlimited  
BCOCA range: 0 to 50 characters (see note) |
| 22   | EAN Two-digit Supplemental | 0123456789 | 2 characters for Modifier X’00’  
14 characters for Modifier X’01’ |
| 23   | EAN Five-digit Supplemental | 0123456789 | 5 characters for Modifier X’00’  
17 characters for Modifier X’01’ |
| 24   | POSTNET | 0123456789 | 5 characters for Modifier X’00’  
9 characters for Modifier X’01’  
11 characters for Modifier X’02’  
BCOCA range for Modifier X’03’: 0 to 50 characters (see note) |
| 26   | RM4SCC | 0123456789, ABCDEFGHIJKLMNOPQRSTUVWXYZ | Symbology: unlimited  
BCOCA range: 0 to 50 characters (see note) |

**Note:** All BCOCA receivers must support at least the BCOCA range. Some receivers support a larger data length.
Table 14. Characters and Code Points used in the BCOCA Symbologies; Excluding Code 128

<table>
<thead>
<tr>
<th>Character</th>
<th>EBCDIC Code Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$'F0'$</td>
</tr>
<tr>
<td>1</td>
<td>$'F1'$</td>
</tr>
<tr>
<td>2</td>
<td>$'F2'$</td>
</tr>
<tr>
<td>3</td>
<td>$'F3'$</td>
</tr>
<tr>
<td>4</td>
<td>$'F4'$</td>
</tr>
<tr>
<td>5</td>
<td>$'F5'$</td>
</tr>
<tr>
<td>6</td>
<td>$'F6'$</td>
</tr>
<tr>
<td>7</td>
<td>$'F7'$</td>
</tr>
<tr>
<td>8</td>
<td>$'F8'$</td>
</tr>
<tr>
<td>9</td>
<td>$'F9'$</td>
</tr>
<tr>
<td>A</td>
<td>$'C1'$</td>
</tr>
<tr>
<td>B</td>
<td>$'C2'$</td>
</tr>
<tr>
<td>C</td>
<td>$'C3'$</td>
</tr>
<tr>
<td>D</td>
<td>$'C4'$</td>
</tr>
<tr>
<td>E</td>
<td>$'C5'$</td>
</tr>
<tr>
<td>F</td>
<td>$'C6'$</td>
</tr>
<tr>
<td>G</td>
<td>$'C7'$</td>
</tr>
<tr>
<td>H</td>
<td>$'C8'$</td>
</tr>
<tr>
<td>I</td>
<td>$'C9'$</td>
</tr>
<tr>
<td>J</td>
<td>$'D1'$</td>
</tr>
<tr>
<td>K</td>
<td>$'D2'$</td>
</tr>
<tr>
<td>L</td>
<td>$'D3'$</td>
</tr>
<tr>
<td>M</td>
<td>$'D4'$</td>
</tr>
<tr>
<td>N</td>
<td>$'D5'$</td>
</tr>
<tr>
<td>O</td>
<td>$'D6'$</td>
</tr>
<tr>
<td>P</td>
<td>$'D7'$</td>
</tr>
<tr>
<td>Q</td>
<td>$'D8'$</td>
</tr>
<tr>
<td>R</td>
<td>$'D9'$</td>
</tr>
<tr>
<td>S</td>
<td>$'E2'$</td>
</tr>
<tr>
<td>T</td>
<td>$'E3'$</td>
</tr>
<tr>
<td>U</td>
<td>$'E4'$</td>
</tr>
<tr>
<td>V</td>
<td>$'E5'$</td>
</tr>
<tr>
<td>W</td>
<td>$'E6'$</td>
</tr>
<tr>
<td>X</td>
<td>$'E7'$</td>
</tr>
<tr>
<td>Y</td>
<td>$'E8'$</td>
</tr>
<tr>
<td>Z</td>
<td>$'E9'$</td>
</tr>
<tr>
<td>-</td>
<td>$'60'$</td>
</tr>
<tr>
<td>$</td>
<td>$'5B'$</td>
</tr>
<tr>
<td>/</td>
<td>$'61'$</td>
</tr>
<tr>
<td>+</td>
<td>$'4E'$</td>
</tr>
<tr>
<td>%</td>
<td>$'6C'$</td>
</tr>
<tr>
<td>:</td>
<td>$'7A'$</td>
</tr>
<tr>
<td>Space</td>
<td>$'40'$</td>
</tr>
</tbody>
</table>

The Code 128 code page (CPGID = 1303) is defined as shown in Figure 77 on page 227.
<table>
<thead>
<tr>
<th>Hex Digits</th>
<th>0-</th>
<th>1-</th>
<th>2-</th>
<th>3-</th>
<th>4-</th>
<th>5-</th>
<th>6-</th>
<th>7-</th>
<th>8-</th>
<th>A-</th>
<th>B-</th>
<th>C-</th>
<th>D-</th>
<th>E-</th>
<th>F-</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0</td>
<td>NUL</td>
<td>DLE</td>
<td>SP00000</td>
<td>SP00000</td>
<td>SP00000</td>
<td>^</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td>SOH</td>
<td>DC1</td>
<td>aSP00000</td>
<td>aSP00000</td>
<td>aSP00000</td>
<td>j</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-2</td>
<td>STX</td>
<td>DC2</td>
<td>bSP00000</td>
<td>bSP00000</td>
<td>bSP00000</td>
<td>s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-3</td>
<td>ETX</td>
<td>DC3</td>
<td>cSP00000</td>
<td>cSP00000</td>
<td>cSP00000</td>
<td>t</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-4</td>
<td></td>
<td></td>
<td>dSP00000</td>
<td>dSP00000</td>
<td>dSP00000</td>
<td>u</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-5</td>
<td>HT</td>
<td>LF</td>
<td>eSP00000</td>
<td>eSP00000</td>
<td>eSP00000</td>
<td>v</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-6</td>
<td>BS</td>
<td>ETB</td>
<td>fSP00000</td>
<td>fSP00000</td>
<td>fSP00000</td>
<td>o</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-7</td>
<td>ESC</td>
<td>EOT</td>
<td>gSP00000</td>
<td>gSP00000</td>
<td>gSP00000</td>
<td>p</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-8</td>
<td>CAN</td>
<td></td>
<td>hSP00000</td>
<td>hSP00000</td>
<td>hSP00000</td>
<td>q</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-9</td>
<td>EM</td>
<td></td>
<td>iSP00000</td>
<td>iSP00000</td>
<td>iSP00000</td>
<td>r</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-A</td>
<td>VT</td>
<td></td>
<td>!SP00000</td>
<td>!SP00000</td>
<td>!SP00000</td>
<td>z</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-B</td>
<td></td>
<td></td>
<td>$SP00000</td>
<td>$SP00000</td>
<td>$SP00000</td>
<td>^</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-C</td>
<td>FF</td>
<td>DC4</td>
<td>&lt;SP00000</td>
<td>*SP00000</td>
<td>%SP00000</td>
<td>@</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-D</td>
<td>CR</td>
<td>GS</td>
<td>&gt;SP00000</td>
<td>&gt;SP00000</td>
<td>&gt;SP00000</td>
<td>&amp;</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-E</td>
<td>SO</td>
<td>RS</td>
<td>?SP00000</td>
<td>?SP00000</td>
<td>?SP00000</td>
<td>=</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-F</td>
<td>SI</td>
<td>US</td>
<td>FN1</td>
<td>FN1</td>
<td>FN1</td>
<td>DEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 77.** Code 128 Code Page (CPGID = 1303)

**Note:** All START, STOP, SHIFT, and CODE characters are generated by the printer to produce the shortest bar code possible from the given data; these characters are not specified in the Bar Code Symbol Data. All code points not listed in the table are undefined. The code points that do not have graphic characters shapes, such as X’00’ (NUL) and X’8F’ (FN1), are control codes defined within the Code 128 symbology; in the HRI, control codes print in a device-dependent manner.

**MOD Parameter**

The modifier field gives additional processing information about the bar code symbol to be generated. For example, it indicates whether a check-digit is to be generated for the bar code symbol.

Table 15 on page 228 shows the modifier values for each bar code type.
The modifier values, by bar code type, are as follows:

**Code 39 (3-of-9 Code), AIM USS-39**

- **X'01'** Present the bar code without a generated check digit.
- **X'02'** Generate a check digit and present it with the bar code.

**MSI (modified Plessey code)**

- **X'01'** Present the bar code without check digits generated by the printer.
- **X'02'** Present the bar code with a generated IBM modulo-10 check digit. This check digit will be the second check digit; the first check digit is the last character of the data as defined in the associated FIELD START and LENGTH subcommands.
- **X'03'** Present the bar code with two check digits. Both check digits are generated using the IBM modulo-10 algorithm.
- **X'04'** Present the bar code with two check digits. The first check digit is generated using the NCR modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals the remainder; exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10.
- **X'05'** Present the bar code with two check digits. The first check digit is generated using the IBM modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals the remainder;
exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10.

X'06' Present the bar code with two check digits. The first check digit is generated using the NCR modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; a first check digit value of 10 is assigned the value zero.

X'07' Present the bar code with two check digits. The first check digit is generated using the IBM modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10.

X'08' Present the bar code with two check digits. The first check digit is generated using the NCR modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10.

X'09' Present the bar code with two check digits. The first check digit is generated using the IBM modulo-11 algorithm; the second using the IBM modulo-10 algorithm. The first check digit equals 11 minus the remainder; exception condition EC-0E00 exists if the first check-digit calculation results in a value of 10.

UPC/CGPC—Version A
X'00' Present the standard UPC-A bar code with a generated check digit. The data to be encoded consists of eleven digits. The first digit is the number-system digit; the next ten digits are the article number.

UPC/CGPC—Version E
X'00' Present a UPC-E bar code symbol. Of the 10 input digits, six digits are encoded. The check digit is generated using all 10 input data digits. The check digit is not encoded; it is only used to assign odd or even parity to the six encoded digits.

UPC—Two-Digit Supplemental
X'00' Present a UPC two-digit supplemental bar code symbol. This option assumes that the base UPC Version A or E symbol is presented as a separate bar code object. The bar and space patterns used for the two supplemental digits are left-odd or left-even parity, with the parity determined by the digit combination.

X'01' The two-digit UPC supplemental bar code symbol is preceded by a UPC Version A, Number System 0, bar code symbol. The bar code object contains both the UPC Version A symbol and the two-digit supplemental symbol. The input data consists of the number system digit, the ten-digit article number, and the two supplement digits, in that order. A check digit is generated for the UPC Version A symbol. The two-digit supplemental bar code symbol is presented after the UPC Version A symbol using left-hand odd and even parity as determined by the two supplemental digits.

X'02' The two-digit UPC supplemental bar code symbol is preceded by a UPC Version E symbol. The bar code object contains both the UPC Version E symbol and the two-digit supplemental symbol. The input
data consists of the ten-digit article number and the two supplemental digits. The bar code object processor generates the six-digit UPC Version E symbol and a check digit. The check digit is used to determine the parity pattern of the six-digit Version E symbol. The two-digit supplemental bar code symbol is presented after the Version E symbol using left-hand odd and even parity as determined by the two digits.

**UPC—Five-Digit Supplemental**

X'00' Present the UPC five-digit supplemental bar code symbol. This option assumes that the base UPC Version A or E symbol is presented as a separate bar code object. A check digit is generated from the five supplemental digits and is used to assign the left-odd and left-even parity of the five-digit supplemental bar code. The supplemental check digit is not encoded or interpreted.

X'01' The five-digit UPC supplemental bar code symbol is preceded by a UPC Version A, Number System 0, bar code symbol. The bar code object contains both the UPC Version A symbol and the five-digit supplemental symbol. The input data consists of the number system digit, the ten-digit article number, and the five supplement digits, in that order. A check digit is generated for the UPC Version A symbol. A second check digit is generated from the five supplement digits. It is used to assign the left-hand odd and even parity of the five-digit supplemental bar code symbol. The supplement check digit is not encoded or interpreted.

X'02' The five-digit UPC supplemental bar code symbol is preceded by a UPC Version E symbol. The bar code object contains both the UPC Version E symbol and the five-digit supplemental symbol. The input data consists of the ten-digit article number and the five-digit supplemental data. The bar code object processor generates the six-digit UPC Version E symbol and check digit. The check digit is used to determine the parity pattern of the Version E symbol. The five-digit supplemental bar code symbol is presented after the Version E symbol. A second check digit is calculated for the five-digit supplemental data and is used to assign the left-hand odd and even parity. The supplement check digit is not encoded or interpreted.

**EAN-8 (includes JAN-short)**

X'00' Present an EAN-8 bar code symbol. The input data consists of seven digits: two flag digits and five article number digits. All seven digits are encoded along with a generated check digit.

**EAN-13 (includes JAN-standard)**

X'00' Present an EAN-13 bar code symbol. The input data consists of twelve digits: two flag digits and ten article number digits, in that order. The first flag digit is not encoded. The second flag digit, the article number digits, and generated check digit are encoded. The first flag digit is presented in HRI form at the bottom of the left quiet zone. The first flag digit governs the A and B number-set pattern of the bar and space coding of the six digits to the left of the symbol center pattern.
Industrial 2-of-5
X'01' Present the bar code without a generated check digit.
X'02' Generate a check digit and present it with the bar code.

Matrix 2-of-5
X'01' Present the bar code symbol without a generated check digit.
X'02' Generate a check digit and present it with the bar code.

Interleaved 2-of-5, AIM USS-I 2/5
X'01' Present the bar code symbol without a check digit.
X'02' Generate a check digit and present it with the bar code.

Codabar, 2-of-7, AIM USS-Codabar
X'01' Present the bar code without a generated check digit. The input data consists of a start character, digits to be encoded, and a stop character, in that order.
X'02' Generate a check digit and present it with the bar code. The input data consists of a start character, digits to be encoded, and a stop character, in that order.

Code 128, AIM USS-128
X'02' Generate a check digit and present it with the bar code.

EAN Two-Digit Supplemental
X'00' Present the EAN two-digit supplemental bar code symbol. This option assumes that the base EAN-13 symbol is presented as a separate bar code object. The value of the two digit supplemental data determines their bar and space patterns chosen from number sets A and B.
X'01' The two-digit supplemental bar code symbol is preceded by a normal EAN-13 bar code symbol. The bar code object contains both the EAN-13 symbol and the two-digit supplemental symbol. The two-digit supplemental bar code is presented after the EAN-13 symbol using left hand odd and even parity as determined by the two supplemental digits chosen from number sets A and B.

Note: Used for both books and paperbacks.

EAN Five-Digit Supplemental
X'00' Present the EAN five-digit supplemental bar code. This option assumes that the base EAN-13 symbol is presented as a separate bar code object. A check digit is calculated from the five supplemental digits. The check digit is also used to assign the bar and space patterns from number sets A and B for the five supplemental digits. The check digit is not encoded or interpreted.
X'01' The five-digit supplemental bar code symbol is preceded by a normal EAN-13 bar code symbol. The bar code object contains both the EAN-13 symbol and the five-digit supplemental symbol. A check digit is generated from the five-digit supplemental data. The check digit is used to assign the bar and space patterns from number sets A and B. The check digit is not encoded or interpreted.
**Note:** Used for books and paperbacks.

**POSTNET**

For all POSTNET modifiers that follow, the BSA HRI flag field and the BSD module width, element height, height multiplier, and wide-to-narrow ratio fields are not applicable to the POSTNET bar code symbology. These fields are ignored because the POSTNET symbology defines specific values for these parameters.

X'00' Present a POSTNET ZIP Code bar code symbol. The ZIP Code to be encoded is defined as a five-digit, numeric (0–9), data variable to the BSA data structure. The POSTNET ZIP Code bar code consists of a leading frame bar, the encoded ZIP Code data, a correction digit, and a trailing frame bar.

X'01' Present a POSTNET ZIP+4 bar code symbol. The ZIP+4 code to be encoded is defined as a nine-digit, numeric (0–9), data variable to the BSA data structure. The POSTNET ZIP+4 bar code consists of a leading frame bar, the encoded ZIP+4 data, a correction digit, and a trailing frame bar.

X'02' Present a POSTNET Advanced Bar Code (ABC) bar code symbol. The ABC code to be encoded is defined as an eleven-digit, numeric (0–9), data variable to the BSA data structure. The POSTNET ABC bar code consists of a leading frame bar, the encoded ABC data, a correction digit, and a trailing frame bar.

X'03' Present a POSTNET variable-length bar code symbol. The data to be encoded is defined as an n-digit, numeric (0–9), data variable to the BSA data structure. The bar code symbol is generated without length checking; the symbol is not guaranteed to be scannable or interpretable. The POSTNET variable-length bar code consists of a leading frame bar, the encoded data, a correction digit, and a trailing frame bar.

**RM4SCC**

A 4 state customer code defined by the Royal Mail Postal service of England for use in bar coding postal code information.

X'00' Present a RM4SCC bar code symbol with a generated start bit, checksum character, and stop bit. The start and stop bits identify the beginning and end of the bar code symbol and also the orientation of the symbol.

Some bar code types and modifiers call for the calculation and presentation of check digits. Check digits are a method of verifying data integrity during the bar coding reading process. Except for UPC Version E, the check digit is always presented in the bar code bar and space patterns, but is not always presented in the HRI. The following table shows the check digit calculation methods for each bar code type and the presence or absence of the check digit in the HRI.
<table>
<thead>
<tr>
<th>Bar Code Type</th>
<th>Modifier</th>
<th>In HRI?</th>
<th>Check Digit Calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 – Code 39 (3-of-9 Code), AIM USS-39</td>
<td>X'02'</td>
<td>Yes</td>
<td>Modulo 43 of the sum of the data characters’ numerical values as described in a Code 39 specification. The start and stop codes are not included in the calculation.</td>
</tr>
<tr>
<td>2 – MSI (modified Plessey code)</td>
<td>X'02' –</td>
<td>No</td>
<td>IBM Modulus 10 check digit: 1. Multiply each digit of the original number by a weighting factor of 1 or 2 as follows: multiply the units digit by 2, the tens digit by 1, the hundreds digit by 2, the thousands digit by 1, and so forth. 2. Sum the digits of the products from step 1. This is not the same as summing the values of the products. 3. The check digit is described by the following equation where “sum” is the resulting value of step 2: (10 - (sum modulo 10)) modulo 10</td>
</tr>
<tr>
<td></td>
<td>X'09'</td>
<td></td>
<td>IBM Modulus 11 check digit: 1. Multiply each digit of the original number by a repeating weighting factor pattern of 2, 3, 4, 5, 6, 7 as follows: multiply the units digit by 2, the tens digit by 3, the hundreds digit by 4, the thousands digit by 5, and so forth. 2. Sum the products from step 1. 3. The check digit depends on the bar code modifier. The check digit as the remainder is described by the following equation where “sum” is the resulting value of step 2: (sum modulo 11) The check digit as 11 minus the remainder is described by the following equation: (11 - (sum modulo 11)) modulo 11</td>
</tr>
<tr>
<td>Bar Code Type</td>
<td>Modifier</td>
<td>In HRI?</td>
<td>Check Digit Calculation</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>--------------</td>
<td>---------</td>
<td>-------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| NCR Modulus 11 check digit:       |              |         | 1. Multiply each digit of the original number by a repeating weight factor pattern of 2, 3, 4, 5, 6, 7, 8, 9 as follows: multiply the units digit by 2, the tens digit by 3, the hundreds digit by 4, the thousands digit by 5, and so forth.  
2. Sum the products from step 1.  
3. The check digit depends on the bar code modifier. The check digit as the remainder is described by the following equation where “sum” is the resulting value of step 2: (sum mod 11)  
The check digit as 11 minus the remainder is described by the following equation: (11 - (sum mod 11)) mod 11 |
| 3 – UPC/CGPC Version A            | X’00’        | Yes     | UPC/EAN check digit calculation:                           |
|                                   |              |         | 1. Multiply each digit of the original number by a weighting factor of 1 or 3 as follows: multiply the units digit by 3, the tens digit by 1, the hundreds digit by 3, the thousands digit by 1, and so forth.  
2. Sum the products from step 1.  
3. The check digit is described by the following equation where “sum” is the resulting value of step 2: (10 - (sum mod 10)) mod 10 |
| 5 – UPC/CGPC Version E            | X’00’        | Yes     | See UPC/CGPC Version A                                    |
| 8 – EAN 8 (includes JAN-short)     | X’00’        | Yes     | See UPC/CGPC Version A                                    |
| 9 – EAN 13 (includes JAN-standard) | X’00’        | Yes     | See UPC/CGPC Version A                                    |
| 10 – Industrial 2-of-5             | X’02’        | Yes     | See UPC/CGPC Version A                                    |
| 11 – Matrix 2-of-5                 | X’02’        | Yes     | See UPC/CGPC Version A                                    |
| 12 – Interleaved 2-of-5            | X’02’        | Yes     | See UPC/CGPC Version A                                    |
| 13 – Codabar, 2-of-7, AIM USS-Codabar | X’02’    | No      | Codabar check digit calculation:                           |
|                                   |              |         | 1. Sum of the data characters’ numerical values as described in a Codabar specification. All data characters are used, including the start and stop characters.  
2. The check digit is described by the following equation where “sum” is the resulting value of step 1: (16 - (sum mod 16)) mod 16 |
<table>
<thead>
<tr>
<th>Bar Code Type</th>
<th>Modifier</th>
<th>In HRI?</th>
<th>Check Digit Calculation</th>
</tr>
</thead>
<tbody>
<tr>
<td>17 – Code 128, AIM USS-128</td>
<td>X'02'</td>
<td>No</td>
<td>Code 128 check digit calculation:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1. Going left to right starting at the start character, sum the value of</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the start character and the weighted values of data and special characters. The weights</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>are 1 for the first data or special character, 2 for the second, 3 for the third,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>and so forth. The stop character is not included in the calculation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2. The check digit is modulo 103 of the resulting value of step 1.</td>
</tr>
<tr>
<td>24 – POSTNET</td>
<td>X'00' – X'03'</td>
<td>NA</td>
<td>The POSTNET check digit is (10 - (sum modulo 10)) modulo 10, where sum is the sum of</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>the ZIP code data.</td>
</tr>
<tr>
<td>26 – RM4SCC</td>
<td>X'00'</td>
<td>NA</td>
<td>The RM4SCC checksum digit is calculated using an algorithm that weights each of the 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>bars within a character in relation to its position within the character.</td>
</tr>
</tbody>
</table>
Appendix E. PPFA Messages and Codes

At the end of processing for each command, the maximum error level encountered during processing is printed on the system printer, providing the error was not caused by the system printer itself. The meaning of the return codes is shown in Table 16.

<table>
<thead>
<tr>
<th>Return Code</th>
<th>Severity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Return Code 0</td>
<td>I=Information; the command is processed.</td>
<td>PPFA did not encounter any problems. No warning, error, severe-error, or termination-error message was issued.</td>
</tr>
<tr>
<td>Return Code 4</td>
<td>W=Warning; the command is processed.</td>
<td>PPFA encountered at least one non-terminating error, solved by an assumption. At least one warning message was issued. No error, severe-error, or terminating-error message was issued. The requested function was probably correctly performed. The program executed to completion.</td>
</tr>
<tr>
<td>Return Code 8</td>
<td>E=Error; the command is partially processed.</td>
<td>PPFA encountered at least one error, but no severe or terminating error. A requested function may be partially incomplete.</td>
</tr>
<tr>
<td>Return Code 12</td>
<td>S=Severe error; the command is not processed.</td>
<td>PPFA encountered a severe error. The program executed to completion, but some of the functions requested were not performed.</td>
</tr>
<tr>
<td>Return Code 16</td>
<td>T=Termination error; the job is terminated.</td>
<td>PPFA encountered a terminating error. The program terminated prematurely.</td>
</tr>
</tbody>
</table>

PPFA Messages and Their Meanings

The general format of the error message is as follows:

All messages consist of a standard seven-character prefix, followed by the message text:

AKQnnnS THIS IS THE MESSAGE TEXT . . .

AKQ is the three-character identifier of Page Printer Formatting Aid for AIX (PPFA).

nnn is the message number.

S is the message-severity indicator. The indicators are defined in Table 16.

Note: You cannot use the psfmsg command to view PPFA messages.

In addition, PPFA errors are written to a listing file. AIX messages are written to standard error. Sometimes, AIX-specific errors mean that PPFA errors are not written to a listing file.

Note: PPFA issues a maximum of 269 user errors generated within a source file, and one additional message is used for the message queue to indicate an out-of-storage condition.
AKQ001E END OF COMMENT (*/) IS NOT SPECIFIED.

Explanation: The end mark of a comment ( */) is not specified.

System Action: The page definition or form definition is not generated. The syntax check may be ended.

Operator Response: Specify the end mark of a comment.

AKQ002E DBCS STRING DOES NOT END WITH SHIFT-IN.

Explanation: DBCS strings in comments must terminate with shift-in.

System Action: The form definition or page definition is not generated. The syntax check continues, assuming shift-in.

Operator Response: Specify a valid DBCS string enclosed by SO and SI.

AKQ003E LITERAL DOES NOT END WITH APOSTROPE.

Explanation: A literal must end with an apostrophe.

System Action: The page definition is not generated. The syntax check continues, assuming an apostrophe.

Operator Response: Specify a valid literal enclosed by apostrophes. Note that an apostrophe in a literal is specified by consecutive double apostrophes.

AKQ004E DBCS LITERAL DOES NOT END WITH SHIFT-IN AND APOSTROPE.

Explanation: A DBCS literal must end with shift-in and apostrophe.

System Action: The page definition is not generated. The syntax check continues, assuming the end of the DBCS literal at the end of a record.

Operator Response: Specify a valid literal ended by shift-in and apostrophe.

AKQ101E COMMAND SEQUENCE IS INVALID.

Explanation: The command sequence is invalid.

System Action: A page definition or form definition is not generated. The syntax check continues from a valid command.

Operator Response: Specify commands in a valid sequence.

AKQ102E INVALID COMMAND (erroneous entry) IS SPECIFIED.

Explanation: An invalid command is specified in the input data.

System Action: A page definition or form definition is not generated. The syntax check continues from a valid command.

Operator Response: Specify a valid command.

AKQ103E INVALID SUBCOMMAND (value) IS SPECIFIED.

Explanation: An invalid subcommand was specified in the input data. This message is often issued when a semicolon (;) is missing.

System Action: A page definition or form definition is not generated. The syntax check continues from the next keyword.

Operator Response: Specify a valid subcommand.

AKQ104E (command or parameter name) NAME IS NOT SPECIFIED.

Explanation: The required name is not specified.

System Action: A page definition or form definition is not generated. The syntax check continues, assuming blanks or default as the name.

Operator Response: Specify the required name.

AKQ105E REQUIRED PARAMETER IN (subcommand name) IS NOT SPECIFIED.

Explanation: The subcommand indicated in the message requires a correct PPFA format.

System Action: A page definition or form definition is not generated. The syntax check continues, assuming the default values.

Operator Response: Refer to the command reference section of this publication for help in specifying a valid subcommand parameter.

AKQ106E (command or parameter name) NAME IS SPECIFIED WITH INVALID SYNTAX.

Explanation: The required name is specified with invalid syntax. See Table 6 on page 121 for the correct length of names.

System Action: A page definition or form definition is not generated. The syntax check continues.

Operator Response: Specify a valid name.
AKQ107E  PARAMETER IN (subcommand name) IS INVALID.

Explanation:  The parameter in the subcommand is invalid (invalid format or out of range).

System Action:  A page definition or form definition is not generated. The syntax check continues, assuming the default values as the parameter.

Operator Response:  Specify a valid parameter value.

AKQ108E  (subcommand name) SUBCOMMAND IS DUPLICATED IN ONE COMMAND.

Explanation:  The subcommand indicated in the message was specified more than once in the same command. Only one such subcommand is permitted within this command.

System Action:  A page definition or form definition is not generated. The syntax check continues, ignoring the duplicate subcommand.

Operator Response:  Delete one subcommand.

AKQ109E  (subcommand name) SUBCOMMAND CONFLICTS WITH (subcommand name) SUBCOMMAND.

Explanation:  One subcommand conflicts with another (FONT, PRINTLINE, FIELD).

System Action:  A page definition or form definition is not generated. The syntax check continues, ignoring the latter subcommand.

User Response:  Delete one subcommand.

AKQ110E  THE VALUE OF THE (command name) SUBCOMMAND IS TOO LARGE OR TOO SMALL.

Explanation:  The parameter in the subcommand is out of range.

IN 136.5
MM 3467.1
CM 346.7
POINTS 9828.0
PELS (L-units) 32760

These values are specified in:

FORMDEF N_UP OVERLAY relative_xpos relative_ypos
PAGEDEF PRINTLINE OVERLAY SEGMENT relative_xpos relative_ypos

Note:  The values specified for the CPI and LPI are set in the SETUNITS subcommand.

System Action:  No form definition or page definition is generated. PPFA continues syntax checking.

Operator Response:  Specify a valid parameter value.

AKQ111E  SUBCOMMAND SEQUENCE IS INVALID: (subcommand name) OCCURS AFTER (subcommand name)

Explanation:  A WHEN subcommand occurs after an OTHERWISE subcommand in a CONDITION command.

System Action:  A page definition or form definition is not generated. The syntax check continues, ignoring the subcommand.

Operator Response:  Reorder or rewrite the conditions.

AKQ112E  CONDITION COMMAND DOES NOT ALLOW '*' IN ITS START SUBCOMMAND.

Explanation:  A relative position ('*', '* + n', or '* - n') was specified in a START subcommand of a CONDITION command.

System Action:  A page definition or form definition is not generated. The syntax check continues from the valid subcommand.

Operator Response:  Specify an absolute starting position.

AKQ113E  MORE THAN ONE 'WHEN' SUBCOMMAND SPECIFIED THE CHANGE PARAMETER.

Explanation:  More than one WHEN subcommand specified CHANGE for its field comparison.

System Action:  A page definition or form definition is not generated. The syntax check continues from the valid subcommand.

Operator Response:  Remove the extra subcommands specifying the CHANGE parameter.

AKQ114E  NUMBER OF PARAMETERS EXCEED LIMIT FOR (subcommand name) SUBCOMMAND OR KEYWORD.

Explanation:  The named subcommand/keyword in the messages limits the number of parameters that may be coded with a single subcommand or keyword. The number of parameters that can be coded with the named subcommand or keyword is defined in the command reference sections of this publication; see Chapter 8, Form Definition Command Reference and Chapter 9, Page Definition Command Reference.  
System Action:  The form definition is not generated. The syntax check continues from the valid subcommand.

Operator Response:  Remove the extra parameters.
AKQ201E  (subcommand name) SUBCOMMAND IS NOT SPECIFIED.

Explanation:  The required subcommand is not specified.

System Action:  A page definition or form definition is not generated.  The syntax check continues, assuming the default.

Operator Response:  Specify the required subcommand.

AKQ202E  SPECIFIED (command name) NAME IS NOT DEFINED.

Explanation:  A resource name (OVERLAY, SUPPRESSION, FONT) is not defined.

System Action:  A page definition or form definition is not generated.  The syntax check continues.

Operator Response:  Correct the name.

AKQ203W  (command name) NAME IS DUPLICATED.

Explanation:  The required name must be unique for OVERLAY, COPYGROUP, FONT, PAGEFORMAT or SUPPRESSION.

System Action:  A page definition or form definition is generated.

Operator Response:  Specify a unique name.

AKQ204E  (object) NAME IS DUPLICATED.

Explanation:  The name must be unique (OVERLAY, COPYGROUP, FONT, PAGEFORMAT, SEGMENT).

System Action:  A page definition or form definition is not generated.  The syntax check continues.

Operator Response:  Specify a unique name.

AKQ205E  PAGEFORMAT (pageformat name) WAS NOT FOUND IN THIS PAGE DEFINITION.

Explanation:  A WHEN or OTHERWISE subcommand of CONDITION specifies a PAGEFORMAT name not found in the page definition being processed.

System Action:  A page definition or form definition is not generated.  The syntax check continues.

Operator Response:  Specify a pageformat name that is in the page definition.

AKQ206E  CONDITION (condition name) HAS ALREADY BEEN DEFINED.

Explanation:  A CONDITION command specifies LENGTH, WHEN, or OTHERWISE, and the condition with this condition name has already been defined by an earlier CONDITION command.

System Action:  A page definition is not generated.  The syntax check continues.

Operator Response:  Define the condition only the first time it occurs.

AKQ210E  THE RELATIVE POSITION VALUE EXCEEDS THE ALLOWED RANGE

System Action:  The value specified for the relative x position or relative y position on the N_UP subcommand (for an OVERLAY) or PRINTLINE command (for an OVERLAY or SEGMENT) exceeds the range of +32760 to -32760 L-units.  Values must be equal to, or less than the following:

<table>
<thead>
<tr>
<th>Units</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td>136.5</td>
</tr>
<tr>
<td>MM</td>
<td>3467.1</td>
</tr>
<tr>
<td>CM</td>
<td>346.7</td>
</tr>
<tr>
<td>POINTS</td>
<td>9828.0</td>
</tr>
<tr>
<td>PELS (L-units)</td>
<td>32760 (+ or -)</td>
</tr>
<tr>
<td>CPI</td>
<td>*</td>
</tr>
<tr>
<td>LPI</td>
<td>*</td>
</tr>
</tbody>
</table>

The value specified for CPI or LPI in the SETUNITS command will determine whether the value will exceed 32760 L-units.

System Action:  The page definition or form definition is not generated.  The syntax check continues.

Operator Response:  Correct the relative x and y position values within the allowed range.

AKQ211E  FRONT/BACK SIDE IS NOT SPECIFIED FOR DUPLEX.

Explanation:  The SUBGROUP specified with BACK does not exist after the SUBGROUP specified with FRONT, or the SUBGROUP specified with FRONT does not exist before the SUBGROUP specified with BACK.

System Action:  A form definition is not generated.  The syntax check continues.

Operator Response:  Specify subgroups for both sides.
AKQ212W PAPER SIDE IS SPECIFIED FOR SIMPLEX.

Explanation: A subgroup specified with BOTH, FRONT, or BACK is invalid with single-sided printing.

System Action: A form definition is generated, ignoring the subcommand specifying the paper side.

User Response: Either delete the subcommand that specified the paper side or specify DUPLEX.

AKQ213E LOGICAL PAGE POSITION EXCEEDS THE LIMIT.

Explanation: The logical page position specified by the OFFSET subcommand in the FORMDEF or COPYGROUP command exceeds the limits.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ214E MORE THAN 127 SUPPRESSIONS ARE SPECIFIED IN ONE FORMDEF.

Explanation: More than 127 suppressions are specified in one FORMDEF.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ215E MORE THAN 127 OVERLAYS ARE SPECIFIED IN ONE COPYGROUP.

Explanation: More than 127 OVERLAYs are specified in one copy group. PPFA can issue this message for an N_UP subcommand that specifies more than 127 overlays.

System Action: No form definition is generated. The syntax check continues.

User Response: Correct the error.

AKQ216E MORE THAN ONE RASTER OVERLAY IS SPECIFIED IN ONE COPYGROUP.

Explanation: More than one raster OVERLAY is specified in one copy group.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ217W LOGICAL PAGE POSITION FOR BACK SIDE OF PAGE SPECIFIED IN SIMPLEX PROCESSING.

Explanation: The logical-page position specified by the OFFSET subcommand in a FORMDEF or COPYGROUP command for the back side of a page was specified, but simplex was specified in a COPYGROUP command.

System Action: A form definition is generated, with the back side logical page position included, as if duplex had been specified. The syntax check continues.

Operator Response: Correct the error by specifying duplex in the COPYGROUP command or remove the second set of coordinates in the OFFSET subcommand.

AKQ218E MORE THAN 255 COPIES ARE SPECIFIED IN ONE COPYGROUP.

Explanation: More than 255 copies are specified in a COPYGROUP.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ219E MORE THAN 127 SUBGROUPS ARE SPECIFIED IN ONE COPYGROUP.

Explanation: More than 127 subgroups are specified in a COPYGROUP.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ220E MORE THAN 8 OVERLAYS ARE SPECIFIED IN ONE SUBGROUP.

Explanation: More than eight overlays are specified in one SUBGROUP.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ221E MORE THAN 8 SUPPRESSIONS ARE SPECIFIED IN ONE SUBGROUP.

Explanation: More than eight suppressions are specified in one SUBGROUP.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the error.
AKQ222W DIFFERENT NUMBERS OF COPIES ARE SPECIFIED FOR EACH SIDE OF DUPLEX.

Explanation: The number of copies for BACK side is not equal to those for FRONT side.

System Action: A form definition is generated assuming the number of copies specified for front side.

Operator Response: Check the number of copies.

AKQ223E LOGICAL PAGE POSITION FOR (page side) SIDE OF PAGE EXCEEDS THE LIMIT.

Explanation: The logical-page position specified by the OFFSET subcommand in a FORMDEF or COPYGROUP command exceeds the limit for the current side of the page.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct the positioning OFFSET parameter.

AKQ224E MORE THAN 127 OVERLAYS ARE SPECIFIED IN A PAGEFORMAT.

Explanation: The maximum number of OVERLAY commands is 127. PPFA can issue this message for the OVERLAY subcommand of the PRINTLINE command.

System Action: A page definition is not generated. The syntax check continues.

User Response: Specify a valid number of OVERLAY commands.

AKQ225E CONSTANT SUBCOMMAND PARAMETER (parameter) SPECIFIED IN SIMPLEX PROCESSING

Explanation: The BACK or BOTH parameter has been specified for the CONSTANT subcommand within simplex processing.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Correct this CONSTANT subcommand or indicate DUPLEX.

AKQ226E DIRECTION SUBCOMMAND ONLY ALLOWED WITH PRESENT SUBCOMMAND.

Explanation: The DIRECTION subcommand has been specified, but the PRESENT subcommand has not.

System Action: A form definition is not generated. The syntax check continues.

Operator Response: Either add the PRESENT subcommand or remove the DIRECTION subcommand.

AKQ227E THE ORIGIN OF THE RESOURCE (name) NAMED IN THE PRINTLINE COMMAND IS OFF THE LOGICAL PAGE.

Explanation: The relative position of the PRINTLINE overlay or segment named is off the logical page. The origin of the overlay or segment specified for the resource named in the N_UP subcommand is off the medium.

System Action: The page definition that has the overlay or segment in question is not generated. PPFA continues the syntax check, ignoring the problem.

Operator Response: Correct the x-position and y-position for the OVERLAY or SEGMENT subcommand.

AKQ228E THE ORIGIN OF THE OVERLAY (overlay name) NAMED IN THE (command) COMMAND IS OFF THE MEDIUM

Explanation: The resource position values will position the resource such that at least part of the resource will be off the medium (physical page).

System Action: The form definition that has the overlay in question is not generated. PPFA continues the syntax check, ignoring the problem.

User Response: Correct the relative x-position and relative y-position values for the OVERLAY named in the N_UP subcommand.

AKQ229W SUBGROUPS FOR FRONT AND BACK OF SAME SHEET USED DIFFERENT BINS.

Explanation: In your subgroup command you specified FRONT and BACK parameters. However, your COPYGROUP has different bins specified.

System Action: A form definition is generated that specifies the bin used for the front side.

Operator Response: Check the number of copies and correct the bin setting.

AKQ231E PRINTLINE IS NOT SPECIFIED.

Explanation: There is no PRINTLINE command in the page format.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Specify the PRINTLINE command.
AKQ232E REQUIRED SUBCOMMAND TEXT OR LENGTH IS NOT SPECIFIED.

Explanation: A FIELD subcommand must have a TEXT or LENGTH subcommand.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Specify either a TEXT subcommand or a LENGTH subcommand.

AKQ233E THE LOGICAL PAGE SIZE IS TOO LARGE OR TOO SMALL.

Explanation: The specified page size is too large or too small. The page size must be from 1 to 30000 pels. The HEIGHT and WIDTH subcommands must have values between 1 and 30000 PELS, inclusive, or the same measurements expressed in other units.

System Action: A page definition is not generated. The syntax check continues, assuming the defaults.

Operator Response: Correct the error.

AKQ234E POSITION OF LINEONE EXCEEDS THE LOGICAL PAGE BOUNDARY.

Explanation: The TOP or MARGIN position specified by the LINEONE subcommand exceeds the logical page boundary. This error message is issued only if TOP or MARGIN is specified.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Specify a valid position value.

AKQ235E MORE THAN 127 SEGMENTS ARE SPECIFIED IN ONE PAGEFORMAT.

Explanation: More than 127 segments are specified in a single PAGEFORMAT command. PPFA can issue this message for the SEGMENT subcommand of the PRINTLINE command.

System Action: No page definition is generated. The syntax check continues.

User Response: Correct the error.

AKQ238E MORE THAN 127 FONTS ARE SPECIFIED IN ONE PAGEFORMAT.

Explanation: More than 127 fonts are specified in one PAGEFORMAT or the specified TRC number exceeds 126. PPFA counts each use of a font in more than one direction or rotation as a separate font.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ239E PRINT POSITION EXCEEDS THE LOGICAL PAGE BOUNDARY.

Explanation: The print position specified by POSITION subcommand exceeds the logical page boundary.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ240E NUMBER OF PRINTLINES, FIELDS, AND CONDITIONS EXCEEDS 65,535 IN ONE PAGEFORMAT.

Explanation: The total number of PRINTLINES, FIELDS, and CONDITIONS exceeds 65,535 in one page format.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Reduce the number of PRINTLINES, FIELDS, or CONDITIONS in the page format.

AKQ241E TOTAL LENGTH OF TEXT DATA EXCEEDS 65,534 BYTES.

Explanation: The total length of text may be up to 65,534 bytes.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ242E THE VALUE OF THE STARTING POSITION OF A RECORD IS TOO LARGE OR TOO SMALL.

Explanation: The START position of a record exceeds the maximum (65,535) or minimum (1) value.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Correct the error.

AKQ243E DBCS LENGTH IS NOT A MULTIPLE OF 2.

Explanation: The number of bytes of DBCS must be a multiple of two. This means that the value of the LENGTH parameter must be a multiple of two.

System Action: A page definition is not generated. The syntax check continues.

Operator Response: Specify a valid length or a valid DBCS.
AKQ244E  INVALID CODE IS SPECIFIED IN THE TEXT.

Explanation:  SBCS text must be within code range X'00' to X'FE'.

Valid double-byte character set (DBCS) codes are between X'41' and X'FE' for each byte. PPFA checks this range. Code X'4040' (blank) is the only exception. For example, the following are valid DBCS codes: X'4040', X'4141', X'41FE', X'FE41', X'FEFE'.

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Specify a valid code.

AKQ245E  HEXADECIMAL TEXT IS INVALID.

Explanation:  Hexadecimal text is specified in an invalid format. Hexadecimal text must have an even length parameter and be in hexadecimal notation ('0' to 'F').

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Specify valid hexadecimal text.

AKQ246E  NULL LITERAL IS SPECIFIED.

Explanation:  The literal has no string.

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Specify a valid literal.

AKQ247E  KANJI NUMBER TEXT IS INVALID.

Explanation:  A kanji number is specified in invalid format. Kanji number text must be a string of kanji numbers delimited by commas. Each kanji number must be a decimal number equal to a valid DBCS code, minus X'4000'.

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Specify valid kanji number(s) in a valid format.

AKQ248E  TEXT ATTRIBUTE CONFLICTS WITH FONT.

Explanation:  SBCS font is specified for DBCS text (type G, K), or DBCS font is specified for SBCS text (type C).

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Correct the error.

AKQ249E  TEXT ATTRIBUTE CONFLICTS WITH TEXT TYPE.

Explanation:  The literal type conflicts with text type. SBCS literal is specified as type G or X, and DBCS literal is specified as type C, X, or K.

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Correct the error.

AKQ250E  TRC NUMBER IS DUPLICATED.

Explanation:  The specified TRC number is duplicated in one page format.

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Correct the error.

AKQ251W  SPECIFIED LENGTH IS SHORTER THAN THE TEXT AND WAS TRUNCATED.

Explanation:  The LENGTH parameter of the TEXT subcommand is shorter than the length of the specified literal, which is truncated to a specified length.

System Action:  The operation continues, truncating the literal.

Operator Response:  Check the truncation.

AKQ252E  TEXT IS NOT THE LENGTH SPECIFIED BY THE LENGTH SUBCOMMAND.

Explanation:  The length of the comparison text in a WHEN or OTHERWISE subcommand of a CONDITION command is not equal to the length specified by the LENGTH subcommand of that CONDITION command.

System Action:  A page definition is not generated. The syntax check continues.

Operator Response:  Change the comparison text or the LENGTH parameter so that they match.

AKQ253E  TEXT IN THE 'WHEN' SUBCOMMAND IS TOO LONG.

Explanation:  Constant text in a WHEN subcommand of a CONDITION command is too long to fit into an 8150-byte CCP structured field.

System Action:  A page definition or form definition is not generated. The syntax check continues.

Operator Response:  Shorten the field to 8000 bytes or fewer, and shorten the comparison text accordingly.
AKQ254E  (text type) LITERAL WAS EXPECTED
   BUT (text type) WAS FOUND.
Explanation:  An SBCS literal occurs where a DBCS
one was expected, or vice versa.
System Action:  A page definition or form definition is
not generated. The syntax check continues.
Operator Response:  In a FIELD command, do not
use a DBCS literal without specifying a DBCS font. In a
CONDITION command, do not mix SBCS and DBCS
literals in the comparison text of a single WHEN
subcommand.

AKQ255E  INVOKE SPECIFIES A SIDE FOR WHICH
NO PLACE SUBCOMMANDS PUT DATA.
Explanation:  The N_UP PLACE subcommand
contains an error that makes it incompatible with the
value specified in the INVOKE subcommand. Either
INVOKE BACK was specified, but PLACE n BACK was
not specified, or INVOKE FRONT was specified, but
PLACE n FRONT was not specified.
System Action:  No form definition is generated.
Processing continues.
User Response:  Specify the same value (FRONT or
BACK) for both the INVOKE and PLACE
subcommands.

AKQ256E  INCORRECT NUMBER OF PLACE
SUBCOMMANDS.
Explanation:  The required number of PLACE
subcommands must be specified.
System Action:  No form definition is generated.
Processing continues.
User Response:  When using N_UP PLACE
subcommands with single-sided printing, the number of
PLACE subcommands must equal the value specified on N_UP. When using duplex printing, the number of
PLACE subcommands must equal two times the value
specified on N_UP.

AKQ257W  CONSTANT (parameter) FOUND WITH
PLACE SUBCOMMAND.
Explanation:  The CONSTANT (parameter)
subcommand can not be specified when N_UP PLACE
subcommands are specified.
System Action:  A form definition is generated without
constant forms control. The syntax check continues.
User Response:  Delete the CONSTANT (parameter)
from the FORMDEF or COPYGROUP command.

AKQ258W  MORE THAN 122 OPERATION
POSITIONS SPECIFIED FOR A FINISH
OPERATION.
Explanation:  More than 122 operation finishing
positions are specified.
System Action:  A form definition will be generated
with 122 finishing positions. All others will be ignored.
User Response:  Move extraneous operator position
values.

AKQ259W  OPCOUNT AND OPPOS VALUES
SPECIFIED. OPCOUNT IGNORED.
Explanation:  Both OPCOUNT and OPPOS are
specified.
System Action:  A form definition is not generated.
Operator Response:  If OPCOUNT is specified,
OPPOS is ignored. When using OPPOS for controlling
the position of each operation on the operation axis,
OPCOUNT is ignored.

AKQ301I  PAGE PRINTER FORMATTING AID
ENDED, MAX RETURN CODE=(max
return code).
Explanation:  This message accompanies the output
listings of all form definitions and page definitions with
the maximum return code for that particular object. Only
when the return code is less than 8 is the object
generated.
System Action:  None.
Operator Response:  None.

AKQ302I  NO ERRORS FOUND IN (resource name)
DEFINITION.
Explanation:  There are no control statements in the
input data.
System Action:  This definition is generated, and
stored or replaced.
Operator Response:  None.

AKQ303S  NO CONTROL STATEMENT(S) ARE
SPECIFIED IN INPUT DATA.
Explanation:  There are no control statements in the
input data.
System Action:  The operation terminates.
Operator Response:  Specify a valid PPFA command.
AKQ304S DEFINITION STATEMENT IS NOT SPECIFIED.
Explanation: There is no FORMDEF or PAGEDEF command in the system input command stream.
System Action: The operation terminates.
Operator Response: Specify valid definition commands.

AKQ305S THIS DEFINITION IS NOT STORED BECAUSE MEMBER ALREADY EXISTS.
Explanation: This form definition or page definition is not saved because a file with the same name already exists in the directory (REPLACE option is NO).
System Action: A page definition or form definition is not generated. The syntax check continues to next definition.
Operator Response: Check the specified form definition or page definition name, and specify REPLACE subcommand YES. Specify another form definition or page definition name.

AKQ311I FORMDEF (form definition name) IS GENERATED AND STORED. MAX RETURN CODE=(max return code).
Explanation: The form definition is generated and stored.
System Action: A form definition is generated.
Operator Response: None.

AKQ312I FORMDEF (command name) IS GENERATED AND REPLACED. MAX RETURN CODE=(max return code).
Explanation: The form definition is generated and is replaced. The maximum return code is listed.
System Action: A form definition is generated.
Operator Response: None.

AKQ313E FORMDEF (form definition name) IS NOT GENERATED. MAX RETURN CODE=(max return code).
Explanation: The form definition is not generated because of an error. The error is indicated by another message.
System Action: A page definition is not generated.
Operator Response: Correct the error.

AKQ321I PAGEDEF (page definition name) IS GENERATED AND FILED. MAX RETURN CODE=(max return code).
Explanation: The page definition is generated and stored.
System Action: A page definition is generated.
Operator Response: None.

AKQ322I PAGEDEF (page definition name) IS GENERATED AND REPLACED. MAX RETURN CODE=(max return code).
Explanation: The page definition is generated and is replaced.
System Action: A page definition is generated.
Operator Response: None.

AKQ323E PAGEDEF (page-definition name) IS NOT GENERATED. MAX RETURN CODE=(max return code).
Explanation: The page definition is not generated because of an error. The error is indicated by another message.
System Action: A page definition is not generated.
Operator Response: Correct the error.

AKQ350T AN UNRECOVERABLE PROGRAM ERROR OCCURRED.
Explanation: There was an error in PPFA logic.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.

AKQ360E FONT COMMAND DOES NOT CONTAIN SUFFICIENT INFORMATION.
Explanation: The FONT command referred to does not contain enough information to generate a valid MCF. This is caused by having a CS parameter without a CP parameter, or vice versa.
System Action: A page definition is not generated.
Operator Response: Correct the referenced FONT command.

AKQ361E FONT COMMAND SPECIFIES CONFLICTING PARAMETERS.
Explanation: A FONT is specified in more than one way, only one of the following is allowed:

- Coded Font
- Character Set, Code Page pair (CS and CP parameters)
- GRID
System Action: A page definition is not generated.
Operator Response: Correct the referenced FONT command.

AKQ362E FONT RATIO SPECIFIED WITHOUT FONT HEIGHT.
Explanation: To scale a font, both the HEIGHT and RATIO must be specified. If a RATIO subcommand is found without a HEIGHT subcommand, the scaling information can not be calculated by PPFA.
System Action: A page definition is not generated.
Operator Response: Correct the referenced FONT command.

AKQ363W HEIGHT SPECIFIED, WIDTH IN GRID IGNORED.
Explanation: You have specified both a HEIGHT and GRID in the FONT command.
System Action: A page definition is not generated.
Operator Response: Correct the referenced FONT command.

AKQ364E INVALID DIRECTION WITH RELATIVE PRINTLINE
Explanation: You specified an incorrect direction with the relative printline in your page definition source. The field direction must match the direction of the printline. The printline direction must be ACROSS.
System Action: A page definition is not generated.
Operator Response: Correct the referenced DIRECTION subcommand.

AKQ365W COLOR AND EXTENDED COLOR SPECIFIED
Explanation: Both COLOR and one of the extended color keywords (RGB, CMYK, HIGHLIGHT, CIELAB) was specified.
System Action: Both requests are placed into the output resource. Output depends on printer function.
Operator Response: If output does not print as expected, remove one of the specifications.

AKQ370E BARCODE NAME WAS NOT PREVIOUSLY DEFINED.
Explanation: You attempted to reference a barcode name that had not been previously defined.
System Action: A page definition is not generated.
Operator Response: Correct the referenced BARCODE subcommand of the FIELD command.

AKQ371E BARCODE NAME WAS PREVIOUSLY DEFINED.
Explanation: You attempted to define a barcode name that had been previously defined.
System Action: A page definition is not generated.
Operator Response: Correct the referenced BARCODE subcommand of the FIELD command.

AKQ372W BARCODE MODIFICATION UNDEFINED FOR TYPE GIVEN.
Explanation: You specified a modification for a barcode that is not defined for the type specified.
See Appendix D, “More About Bar Code Parameters” on page 223 for more information.
System Action: A page definition is generated as specified. This is done so that, as new bar code types and modifications are introduced, you can create page definitions for them. However, you will receive this warning, because the specification could also be an error.
Operator Response: Correct the referenced BARCODE subcommand of the FIELD command, if appropriate.

AKQ373W BARCODE TYPE IS UNDEFINED.
Explanation: You specified a bar code type that is not defined.
System Action: A page definition is generated as specified. This is done so that, as new bar code types and modifications are introduced, you can create page definitions for them. However, you will receive this warning, because this specification could also be an error.
Operator Response: Correct the referenced BARCODE subcommand of the FIELD command, if appropriate.

AKQ374W INVALID DATA LENGTH FOR SELECTED BARCODE TYPE AND MODIFICATION.
Explanation: You specified a data length for a defined barcode type and modification that is invalid for that combination of type and modification.
See Appendix D, “More About Bar Code Parameters” on page 223 for more information.
System Action: A page definition is generated as specified. This is done so that, as new bar code types and modifications are introduced, you can create page definitions for them. However, you will receive this warning, because this specification could also be an error.
Operator Response: Correct the referenced BARCODE subcommand of the FIELD command, if appropriate.

AKQ401E EXEC PARAMETER IS INVALID.
Explanation: The program parameter specification is invalid.
System Action: A page definition or form definition is not generated. The syntax check continues.
Operator Response: Specify a valid program parameter.

AKQ402T ERROR OCCURRED DURING ATTEMPT TO OBTAIN STORAGE
Explanation: conditions generate this message:
1. Exceeds the defined size to hold the compiled data for the page definition and form definition (usually default of 128K).
2. Insufficient available disk space on the file system to write the output of the compiler.
3. Exceeds the limit of 269 user errors generated within a PPFA source file.
System Action: The operation terminated.
Operator Response:
1. Increase the parameter size for the page definition and form definition to correct the defined size.
2. Increase the size of the file system or specify a directory on another file system that has more disk space.
3. Fix the errors reported to this point and re-run PPFA.

AKQ403T ERROR OCCURRED DURING ATTEMPT TO FREE STORAGE.
Explanation: A system error occurred while PPFA attempted to free disk space at the end of an execution.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.

AKQ404T SYSIPT OPEN FAILURE.
Explanation: SYSIPT cannot be opened.
System Action: The operation terminates.
Operator Response: Assign a valid input data file.

AKQ405T INSUFFICIENT STORAGE TO EXECUTE PPFA.
Explanation: The region size is too small to execute PPFA.
System Action: The operation terminates.
Operator Response: Increase the region size available to the job.

AKQ410T (Librarian error message).
Explanation: The message describes a librarian error.
System Action: The operation terminates.
Operator Response: Contact a system programmer.

AKQ411T FORMDEF LIBRARY OPEN FAILURE.
Explanation: The FORMDEF library cannot be opened.
System Action: The operation terminates.
Operator Response: Assign a valid FORMDEF library.

AKQ412T FORMDEF LIBRARY I/O ERROR.
Explanation: An I/O error occurred during an attempted access of a form definition directory.
System Action: The operation terminates.
Operator Response: Check the permissions of the directory. If you do not have access, contact the owner of the directory. If this does not resolve the problem, contact a system programmer.

AKQ413T FORMDEF DIRECTORY CANNOT BE UPDATED.
Explanation: The FORMDEF member cannot be registered on the directory.
System Action: The operation terminates.
Operator Response: Contact a system programmer.

AKQ414T FORMDEF LIBRARY CLOSE FAILURE.
Explanation: A form definition directory cannot be closed.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.
AKQ415T  PAGEDEF LIBRARY OPEN FAILURE.
Explanation: The PAGEDEF library cannot be opened.
System Action: The operation terminates.
Operator Response: Assign a valid PAGEDEF library.

AKQ416T  PAGEDEF LIBRARY I/O ERROR.
Explanation: I/O error occurs during an attempted access of a page definition directory.
System Action: The operation terminates.
Operator Response: Check the permissions of the directory. If you do not have access, contact the owner of the directory. If this does not resolve the problem, contact a system programmer.

AKQ417T  PAGEDEF DIRECTORY CANNOT BE UPDATED.
Explanation: A page definition file cannot be registered on the directory.
System Action: The operation terminates.
Operator Response: Contact a system programmer.

AKQ418T  PAGEDEF LIBRARY CLOSE FAILURE.
Explanation: A page definition directory cannot be closed.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.

AKQ420T  SYSTEM ERROR. ABEND CODE=(ABEND code).
Explanation: System forces PPFA to terminate abnormally.
System Action: The operation terminates.
Operator Response: Contact a system programmer. Refer to the documentation for your operating system.

AKQ421T  FORMDEF LIBRARY IS FULL.
Explanation: The file system into which PPFA attempted to save the form definition is full.
System Action: The operation terminates.
Operator Response: Increase the size of the file system or specify a directory on a file system that has more disk space.

AKQ422T  PAGEDEF LIBRARY IS FULL.
Explanation: The file system into which PPFA attempted to save the page definition is full.
System Action: The operation terminates.
Operator Response: Increase the size of the file system or specify a directory on a file system that has more disk space.

AKQ501T  SYSIN OPEN FAILURE.
Explanation: The PPFA input source file cannot be opened.
System Action: The operation terminates.
Operator Response: Specify a valid input source file.

AKQ502T  SPANNED RECORD OF SYSIN IS NOT SUPPORTED.
Explanation: The spanned record of the PPFA input source file is not supported.
System Action: The operation terminates.
Operator Response: Specify a valid input record format.

AKQ503T  UNDEFINED LENGTH RECORD OF SYSIN IS NOT SUPPORTED.
Explanation: An undefined length record of PPFA input source file is not supported.
System Action: The operation terminates.
Operator Response: Specify a valid input record format.

AKQ504T  LOGICAL RECORD LENGTH OF SYSIN EXCEEDS LIMIT.
Explanation: The logical record length of the PPFA input source file exceeds limit.
System Action: The operation terminates.
Operator Response: Specify a valid input record length.

AKQ510T  FORMDEF/PAGEDEF LIBRARY OPEN FAILURE.
Explanation: The FORMDEF or PAGEDEF directory cannot be opened.
System Action: The operation terminates.
Operator Response: Specify a valid FORMDEF or PAGEDEF or check to make sure that the directory is correct.
AKQ511T  I/O ERROR OCCURRED DURING (FORMDEF/PAGEDEF) DIRECTORY SEARCH. RETURN CODE = (return code) REASON CODE = (reason code)

Explanation:  I/O error occurred while performing FIND function.
System Action:  The operation terminates.
Operator Response:  Contact a system programmer.

AKQ512T  LOGICAL RECORD LENGTH OF FORMDEF/PAGEDEF EXCEEDS LIMIT.

Explanation:  The logical record length exceeds maximum or minimum value.
System Action:  The operation terminates.
Operator Response:  Specify a filename that has a valid record length.

AKQ513T  BLOCK SIZE OF FORMDEF/PAGEDEF EXCEEDS LIMIT.

Explanation:  The block size exceeds maximum or minimum value.
System Action:  The operation terminates.
Operator Response:  Assign a filename that has a valid block size.

AKQ514T  UNDEFINED LENGTH RECORD IS NOT SUPPORTED IN FORMDEF/PAGEDEF LIBRARY.

Explanation:  An undefined length record is not supported in FORMDEF/PAGEDEF directory.
System Action:  The operation terminates.
Operator Response:  Assign a valid record format.

AKQ515T  FIXED LENGTH RECORD IS NOT SUPPORTED IN FORMDEF/PAGEDEF LIBRARY.

Explanation:  The fixed length record is not supported in the FORMDEF or PAGEDEF library.
System Action:  The operation terminates.
Operator Response:  Assign a valid record format.

AKQ516T  NO CONTROL CHARACTER RECORD IS SUPPORTED IN FORMDEF/PAGEDEF LIBRARY.

Explanation:  No control character record is supported in FORMDEF/PAGEDEF directory.
System Action:  The operation terminates.
Operator Response:  Assign a valid record format.

AKQ517T  NO SPACE IN FORMDEF/PAGEDEF DIRECTORY.

Explanation:  No space was available in the FORMDEF directory or the PAGEDEF directory to add or replace the resource.
System Action:  The operation terminates.
Operator Response:  Increase the directory space or specify a directory on another file system that has more disk space.

AKQ518T  I/O ERROR OCCURRED WHILE UPDATING FORMDEF/PAGEDEF DIRECTORY. RETURN CODE=(return code). REASON CODE=(reason code).

Explanation:  A permanent I/O error was detected, or the specified data control block is not opened, or insufficient disk space exists to perform the write function.
System Action:  The operation terminates.
Operator Response:  Contact a system programmer.

AKQ519T  I/O ERROR OCCURRED DURING WRITE.

Explanation:  The error message is displayed.
System Action:  The operation terminates.
Operator Response:  Contact a system programmer.

AKQ520T  SPANNED RECORD IS NOT SUPPORTED IN FORMDEF/PAGEDEF LIBRARY.

Explanation:  The spanned record is not supported in the FORMDEF or PAGEDEF library.
System Action:  The operation terminates.
Operator Response:  Remove the SPAN attribute and assign a valid dataset.

AKQ522T  BLOCK SIZE IS NOT SPECIFIED FOR FORMDEF/PAGEDEF DATA SET.

Explanation:  A block size is not specified for FORMDEF/PAGEDEF data set.
System Action:  The operation terminates.
Operator Response:  Specify a BLKSIZE in the DD statement.
AKQ540T  SYSTEM ABEND (code) OCCURRED IN PPFA/MVS PROCESS.
Explanation: A system ABEND (code) occurred in PPFA/MVS process. Termination processing was performed by the ESTAE macro instruction.
System Action: The operation terminates.
Operator Response: Contact a system programmer. Refer to AIX Operating System Messages

AKQ541T  USER ABEND (code) OCCURRED IN PPFA/MVS PROCESS.
Explanation: A user ABEND (code) occurred in PPFA/MVS process. Termination processing was performed by the ESTAE macro instruction.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.

AKQ600T  INPUT FILENAME NOT SPECIFIED.
Explanation: You did not specify an input filename.
System Action: The operation terminates.
Operator Response: Enter the input filename.

AKQ601T  INPUT FILETYPE NOT SPECIFIED.
Explanation: You did not specify an input filetype.
System Action: The operation terminates.
Operator Response: Enter the input filetype.

AKQ602T  COMMAND SYNTAX IS NOT VALID.
Explanation: The command syntax you entered was not accepted.
System Action: The operation terminates.
Operator Response: Enter a valid command.

AKQ603T  FILEMODE FOR (FORMDEF/PAGEDEF/LISTING) IS INVALID.
Explanation: You entered an invalid filemode for FORMDEF, PAGEDEF, or LISTING.
System Action: The operation terminates.
Operator Response: Enter a valid file extension.

AKQ604T  INVALID PARAMETER IS SPECIFIED IN (FORMDEF/PAGEDEF/LISTING/SIZE) OPTION.
Explanation: You entered an invalid parameter for FORMDEF, PAGEDEF, LISTING, or SIZE.
System Action: The operation terminates.
Operator Response: Enter a valid option parameter.

AKQ605T  (FORMDEF/PAGEDEF/LISTING/SIZE) KEYWORD IS DUPLICATED.
Explanation: You entered a duplicate keyword for FORMDEF, PAGEDEF, LISTING, or SIZE.
System Action: The operation terminates.
Operator Response: Enter a unique keyword.

AKQ606T  FILETYPE FOR (FORMDEF/PAGEDEF/LISTING) NOT SPECIFIED.
Explanation: The filetype for FORMDEF, PAGEDEF, or LISTING was not entered.
System Action: The operation terminates.
Operator Response: Enter an appropriate filetype.

AKQ607T  INVALID KEYWORD SPECIFIED.
Explanation: The keyword you entered was not accepted.
System Action: The operation terminates.
Operator Response: Enter a valid keyword.

AKQ608T  INVALID SIZE PARAMETER SPECIFIED.
Explanation: The size parameter specified is not valid.
System Action: The operation terminates.
Operator Response: Enter a valid size parameter.

AKQ610T  SIZE PARAMETER VALUE EXCEEDS THE ALLOWABLE MAXIMUM.
Explanation: The size entered exceeds the maximum allowable.
System Action: The operation terminates.
Operator Response: Enter a valid size value.

AKQ611T  SIZE PARAMETER VALUE IS TOO SMALL.
Explanation: The size entered is too small for executing in PPFA/VM.
System Action: The operation terminates.
Operator Response: Enter a valid size value.
AKQ612T INVALID FILE IDENTIFIER ‘*’ SPECIFIED FOR INPUT FILE.
Explanation: ‘*’ is specified for input filename or filetype.
System Action: The operation terminates.
Operator Response: Enter a valid filename or filetype.

AKQ613T SIZE PARAMETER VALUE IS MISSING.
Explanation: You did not specify a size parameter
System Action: The operation terminates.
Operator Response: Specify a valid size parameter.

AKQ620T INPUT FILE WAS NOT FOUND.
Explanation: The input filename entered was not found.
System Action: The operation terminates.
Operator Response: Correct the input filename.

AKQ621T NO READ/WRITE (file mode) DISK ACCESSED FOR (INPUT/LISTING/FORMDFF/PAGEDEF/OUTPUT).
Explanation: The disk on which the file is saved cannot be read from or written to because it either was not accessed or was accessed using an invalid access mode.
System Action: The operation terminates.
Operator Response: Access the file system using a valid access mode.

AKQ622T INPUT FILE EXCEEDS THE ALLOWABLE LOGICAL RECORD LENGTH MAXIMUM.
Explanation: The logical record length of the input file exceeds the limit.
System Action: The operation terminates.
Operator Response: Correct the logical record length of the file.

AKQ624T I/O ERROR OCCURRED IN (AKQINIO/AKQLBIO/AKQPRIO) MODULE. RC = (return code from FWRITE/FGETS macro instruction).
Explanation: An I/O error occurred during either FGETS or FWRITE processing of module AKQINIO, AKQLBIO, or AKQPRIO.
System Action: The operation terminates.
Operator Response: None.

AKQ625T DISK (file mode) IS FULL.
Explanation: Not enough space is available on the specified file system to write the file.
System Action: The operation terminates.
Operator Response: Erase some files from the specified file disk and re-execute.

AKQ639T ABEND EXIT ROUTINE FAILED TO EXECUTE. RC = (return code from ABNEXIT macro)
Explanation: ABEND exit routine cannot be established.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.

AKQ640T SYSTEM ABEND (code) OCCURRED IN PPFA/VM PROCESS.
Explanation: A system ABEND occurred during processing. The ABEND exit routine ended processing.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.

AKQ641T USER ABEND (code) OCCURRED IN PPFA/VM PROCESS.
Explanation: A user-initiated ABEND occurred during processing. The ABEND exit routine ended the processing.
System Action: The operation terminates.
Operator Response: Use local problem-reporting procedures to report this message.

AKQ700I SIZE PARAMETER IS NO LONGER NECESSARY IN PPFA/370.
Explanation: The storage required to contain the messages and control blocks is not automatically set at 32K and 128K respectively. If the control block storage is used up, an additional 128K will be gotten and chained to the previous. All storage necessary to perform the compile will be obtained during processing.
System Action: The compile process continues.
Operator Response: None.
Glossary

Source Identifiers

This publication includes terms and definitions from the IBM Dictionary of Computing, SC20-1699.

Definitions reprinted from the American National Dictionary for Information Processing Systems are identified by the symbol (A) following the definition.

Definitions reprinted from a published section of the International Organization for Standardization’s Vocabulary—Information Processing or from a published section of the ISO Vocabulary—Office Machines are identified by the symbol (1) following the definition. Because many ISO definitions are also reproduced in the American National Dictionary for Information Processing Systems, ISO definitions may also be identified by the symbol (A).

Definitions reprinted from working documents, draft proposals, or draft international standards of ISO Technical Committee 97, Subcommittee 1 (Vocabulary) are identified by the symbol (T) following the definition, indicating that final agreement has not yet been reached among its participating members.

Definitions that are specific to IBM products are so labeled, for example, “In SNA,” or “In the 3820.”

References

The following cross-references are used in this glossary:

Contrast with. This refers to a term that has an opposed or substantively different meaning.

See. This refers the reader to multiple-word terms that have the same last word.

See also. This refers the reader to related terms that have a related, but not synonymous, meaning.

Synonym for. This indicates that the term has the same meaning as a preferred term, which is defined in its proper place in the glossary.

Synonymous with. This is a backward reference from a defined term to all other terms that have the same meaning.

Terms

A

advanced function printing (AFP)

?. The ability of program products to place text and image data at any addressable point on the page.

AFP. Advanced function printing.

all-points addressability

?. The capability to address, reference, and position data elements at any addressable position in a presentation space or on a physical medium. An example of all points addressability is the positioning of text, graphics, and images at any addressable point on the physical medium. See also picture element.

all-points-addressable mode

?. Synonym for page mode.

alphanumeric string

?. A sequence of characters consisting solely of the letters a through z and the numerals 0 through 9.

American National Standards Institute (ANSI)

?. An organization consisting of producers, consumers, and general interest groups. ANSI establishes the procedures by which accredited organizations create and maintain voluntary industry standards in the United States. It is the United States constituent body of the International Organization for Standardization (ISO).

ANSI. See American National Standards Institute.

APA. All points addressable.

application

?. (1) The use to which an information system is put.
(2) A collection of software components used to perform specific types of work on a computer.

application program

?. A program written for or by a user that applies to the user’s work.

ascender
? The parts of certain lowercase letters, such as b, d, or f, which at zero-degree character rotation rise above the top edge of other lowercase letters such as a, c, and e. Contrast with descender.

attribute

? A property or characteristic of one or more constructs. For example, character attribute, color attribute, current drawing attributes, default drawing attributes, line attributes, marker attributes, and pattern attributes.

B

bar

? In bar codes, the darker element of a printed bar code symbol.

bar code

? An array of parallel rectangular bars and spaces that together represent data elements or characters of a particular type. The bars and spaces are arranged in a predetermined pattern following unambiguous rules defined by the bar code type.

bar code command set

? In the IPDS architecture, a collection of commands used to present bar code symbols in a page, page segment, or overlay.

bar code density

? The number of characters per inch (cpi) in a bar code symbology. In most cases, the range is three to ten cpi.

bar code object area

? The rectangular area on a logical page into which a bar code presentation space is mapped.

Bar Code Object Content Architecture (BCOCA)

? An architected collection of constructs used to interchange and present bar code data.

bar code symbol

? A combination of characters including start and stop characters, quiet zones, data characters, and check characters required by a particular bar code type, that form a complete, scannable entity.

bar height

? In bar codes, the bar dimension perpendicular to the bar width. Synonymous with bar length and height.

bar code symbology

? A bar code language. Bar code symbologies are defined and controlled by various industry groups and standards organizations. Bar code symbologies are described in public domain bar code specification documents. Synonymous with symbology. Examples of bar code symbology include: Canadian Grocery Product Code (CGPC), European Article Numbering (EAN), Japanese Article Numbering (JAN), and Universal Product Code (UPC).

bar length

? In bar codes, the bar dimension perpendicular to the bar width. Synonymous with bar length and height.

bar width

? In bar codes, the thickness of a bar measured from the edge closest to the symbol start character to the trailing edge of the same bar.

baseline

? A conceptual line with respect to which successive characters are aligned.

baseline direction

? The direction in which successive lines of text appear on a logical page.

BCOCA. See Bar Code Object Content Architecture.

bin

? The standard-size paper source on cut-sheet page printers that have more than one paper source. Each printer is set up with either A4 or letter-size paper as the standard size. Contrast with cassette.

BITS

? A data type for architecture syntax, indicating one or more bytes to be interpreted as bit string information.

body. (1) On a printed page, the area between the top and bottom margins that can contain data. (2) In a book, the portion between the front matter and the back matter.

boldface. (1) A heavy-faced type. (2) Printing in heavy-faced type.
carriage control character. If present, the first character of an output record (line) that is to be printed or spaced; it determines how many lines should be skipped before the line.

cassette

code page

character ascender. See ascender.

character code. An element of a code page or a cell in a code table to which a character can be assigned. The element is associated with a binary value. The assignment of a character to an element of a code page determines the binary value that will be used to represent each occurrence of the character in a character string.

character descender. See descender.

character identifier

character set


CGPC. See Canadian Grocery Product Code.

CHAR

Code Page Global Identifier (CPGID)

Code39

Code128

coded font

character set attribute. An attribute used to specify a coded font.

code point

character attribute

character baseline

character code

character identifier

character set

Glossary 255
match the graphic character set of the code page for
the coded font resource pair to be valid. (3) In the
IPDS architecture, a raster font resource containing
code points that are directly paired to font metrics and
the raster representation of character shapes, for a
specific graphic character set. (4) In the IPDS
architecture, a font resource containing descriptive
information, a code page, font metrics, and a
digital-technology representation of character shapes for
a specific graphic character set.

Coded Graphic Character Set Global Identifier
(CGCSGID)

A four-byte binary or a ten-digit decimal identifier
consisting of the concatenation of a GCSGID and a
CPGID. The CGCSGID identifies the code point
assignments in the code page for a specific graphic
character set, from among all the graphic characters
that are assigned in the code page.

color attribute

An attribute that affects the color values provided in
a graphics primitive, a text control sequence, or an
IPDS command. Examples of color attributes are
foreground color and background color.

command

A request for performance of an operation or
execution of a program. In Page Printer Formatting Aid,
commands are control statements for major formatting
functions. For example, FORMDEF and COPYGROUP
are commands. Commands are further specified by
subcommands and parameters.

command stream. The sequence of Page Printer
Formatting Aid commands that is submitted with the job
control statements in a Page Printer Formatting Aid
execution. The commands and subcommands are the
task statements that define the object or objects to
be generated.

compatibility mode

Use of Table Reference Characters (TRCs) that are
acceptable to line printers and page printers and that
access page definitions with little or no change to the
user's data or to the job command stream. Contrast with
page mode.

composted-text data file. A file containing text data
and text control information that dictates the format,
placement, and appearance of the data to be printed.

conditional processing. A page definition function
that allows input data records to partially control their
own formatting.

Default

An architected set of data such as a structured field
or a triplet.

color character

A character that denotes the start, modification,
or end of a control function. A control character can be
recorded for use in a subsequent action, and it can
have a graphic representation. See also character.
(2) A control function the coded representation of which
consists of a single code point.

copy group. A subset of a form definition containing a
set of controls for the physical pages of a printout. Such
functions as the selection of either of two paper sources
on the page printer, the use of duplex printing, or the
positioning of the reference point for all printing on the
sheet are available in the copy group.

cm. Centimeters.

CMS. Conversational Monitor System.

cpi. Characters per inch.

cut-sheet media

Unconnected sheets. Contrast with continuous-form
media.

D

data map. An internal object whose structured fields
control the formatting of data on a logical page of a
printout. Created by a PAGEDEF command or a
PAGEFORMAT command.

data stream

A continuous stream of data that has a defined
format. An example of a defined format is a structured
field.

DBCS. Double-byte character set.

default. Pertaining to an attribute, value, or option that
is assumed when none is explicitly specified and one is
needed to continue processing.

density

The number of characters per inch (cpi) in a bar
code symbology. In most cases, the range is three to
ten cpi.

descender

In a font, the distance from the baseline to the
bottom of the character box. This value may differ for
different characters in a given font. Contrast with ascender.

direction

? The print position of data in a logical page, line, or field. In Page Printer Formatting Aid, the ultimate reference point for all direction controls on a page is the hardware origin. Secondary and tertiary reference points are possible as well, allowing more than one print direction on a page.

document

? (1) A machine-readable collection of one or more objects that represents a composition, a work, or a collection of data. (2) A publication or other written material.

double-byte character set (DBCS)

? A character set, such as a set of Japanese ideographs, requiring two bytes to identify each character.

duplex printing

? Printing on both sides of a sheet.

E

EAN. See European Article Numbering.

EAN2SUP. A parameter that specifies a bar code type of European Article Numbering, Two-digit Supplemental.

EAN5SUB. A parameter that specifies a bar code type of European Article Numbering, Five-digit Supplemental.

EAN8. A parameter that specifies a bar code type of European Article Numbering 8 (includes Japanese Article Numbering-short).

EAN13. A parameter that specifies a bar code type of European Article Numbering 13 (includes Japanese Article Numbering-standard).

EBCDIC. See Extended Binary-Coded Decimal Interchange Code.

electronic overlay

? In IBM Print Services Facility, a collection of constant data that are electronically composed in the host processor and can be merged with variable data on a sheet during printing. Contrast with page segment. See also overlay, preprinted form.

European Article Numbering (EAN)

? The bar code symbology used to code grocery items in Europe.

Extended Binary-Coded Decimal Interchange Code (EBCDIC)

? A coded character set that consists of eight-bit coded characters.

external library resource (member). Objects that can be used by other program products while running print jobs; for example, coded fonts, code pages, font character sets, form definitions, page definitions, and page segments. Synonym for resource object.

eexternal object. Synonym for resource object.

F

FCB. Forms control buffer.

field. (1) In a record, a specified area used for a particular class of data; for example, a group of character positions used to enter or display wage rates on a screen. (2) In Page Printer Formatting Aid, any area of a record singled out for particular formatting treatment.

field processing. Mapping individual fields to a page of output with special formatting controls.

file. A named set of records stored or processed as a unit. (T)

fixed medium information. Information that can be applied to a sheet by a printer or printer-attached device that is independent of data provided through the data stream. Fixed medium information does not mix with the data provided by the data stream and is presented on a sheet either before or after the text, image, graphics, or bar code data provided within the data stream. Fixed medium information can be used to create “pre-printed forms,” or other types of printing, such as colored logos or letterheads, that cannot be created conveniently within the data stream.

FOCA. See Font Object Content Architecture.

font

? A family or assortment of characters of a given size and style; for example, 9-point Bodoni Modern. (A)

font character set

? A FOCA resource containing descriptive information, font metrics, and the digital representation of character shapes for a specified graphic character set.

Font Object Content Architecture (FOCA)
An architected collection of constructs used to describe fonts and to interchange those font descriptions.

form

A physical piece of paper or other medium on which output data is printed. For cut-sheet printers, a form is one sheet of paper or other medium. For continuous-forms printers, the form is the area of paper (or other medium) defined to the printer as a single physical page, which for fan-fold paper is normally the area between perforations. See also medium, sheet, and page.

format. The arrangement or layout of data on a physical medium or in a presentation space.

formatted data

In FD:OCA, data whose implied syntax and semantics are represented by architected controls that accompany the data.

formatted data object (FDO)

An object that contains formatted data. See also object.

Formatted Data Object Content Architecture (FD:OCA)

An architected collection of constructs used to interchange formatted data.

formatter. A process used to prepare a document for presentation.

Formdef. See Form Definition.

form definition

In IBM Print Services Facility, a resource object that defines the characteristics of the form, which include: overlays to be used, text suppression, position of page data on the form, and modifications and number of copies of a page.

forms control buffer (FCB). A line printer control. In the 3800 Printing Subsystem, a buffer for controlling the vertical format of printed output.

forms flash

(1) In the 3800 Printing Subsystem, the function of the printer that allows user-prepared images to be printed with variable page data. An operator must insert the desired image holder when forms overlay printing is desired. (2) The photographic negative of a predefined design to be exposed to the photoconductor by a flash of light. The forms overlay can be merged with variable data during printing. See also electronic overlay.

G

GCGID. See Graphic Character Global Identifier.

GCSGID. See Graphic Character Set Global Identifier.

GID. See global identifier.

Global Identifier (GID)

Any of the following:
- Code Page Global ID (CPGID)
- Graphic Character Global Identifier (GCGID)
- Font Typeface Global Identifier (FGID)
- Graphic Character Set Global Identifier (GCSGID)
- Coded Graphic Character Set Global Identifier (CGCSGID)
- In MO:DCA, an encoded graphic character string that provides a reference name for a document element.
- Global Resource Identifier (GRID)
- Object Identifier (OID)
- Coded Character Set Identifier (CCSID).

Global resource identifier (GRID)

An eight-byte identifier that identifies a coded font resource. A GRID contains the following fields in the order shown:

1. GCSGID of a minimum set of graphic characters required for presentation. It can be a character set that is associated with the code page, or with the font character set, or with both.
2. CPGID of the associated code page
3. FGID of the associated font character set
4. Font width in 1440ths of an inch.

GOCA. See Graphics Object Content Architecture.

graphic character

A member of a set of symbols that represent data. Graphic characters can be letters, digits, punctuation marks, or other symbols. Synonymous with glyph. See also character.

Graphic Character Global Identifier (GCGID)

An alphanumeric character string used to identify a specific graphic character. A GCGID can be from four-bytes to eight-bytes long.

graphic character identifier
Glossary

host

? (1) In the IPDS architecture, a computer that drives a printer. (2) In IOCA, the host is the controlling environment.

HRI. See human-readable interpretation.

human-readable interpretation (HRI)

? The printed translation of bar code characters into equivalent Latin alphabetic characters, Arabic numeral decimal digits, and common special characters normally used for printed human communication.

I

image

? An electronic representation of a picture produced by means of sensing light, sound, electron radiation, or other emanations coming from the picture or reflected by the picture. An image can also be generated directly by software without reference to an existing picture.

image content

? Image data and its associated image data parameters.

Image Object Content Architecture (IOCA)

? An architected collection of constructs used to interchange and present images.

in. Inches.

IND2OF5. A parameter that specifies a bar code type of Industrial 2-of-5.

inline. In printing, the direction of successive characters in a line of text. Synonymous with inline direction.

inline direction. Synonym for inline.

Intelligent Printer Data Stream (IPDS)

? An architected host-to-printer data stream that contains both data and controls defining how the data is to be presented.

International Organization for Standardization (ISO)

? An organization of national standards bodies from various countries established to promote development of standards to facilitate international exchange of goods and services, and develop cooperation in intellectual, scientific, technological, and economic activity.
Invoke Data Map. A control record placed in the user's data to begin a new page format.

Invoke Medium Map. A control record placed in the user's data to begin a new copy group.

IOCA. See Image Object Content Architecture.

IPDS. See Intelligent Printer Data Stream.

ISO. See International Organization for Standardization.

italics. A typeface with characters that slant upward to the right. In FOCA, italics is the common name for the defined inclined typeface posture attribute or parameter.


J

JAN. See Japanese Article Numbering.

Japanese Article Numbering (JAN)

?.. The bar code symbology used to code grocery items in Japan.

jog. Offset stacking of individual sheets or sets of sheets in the output hopper of a page printer or copy mark in continuous

K

kanji. A graphic character set consisting of symbols used in Japanese ideographic alphabets. Each character is represented by 2 bytes.

keyword. A two-part self-defining parameter consisting of a one-byte identifier and a one-byte value.

L

landscape presentation

?.. The position of a printed sheet that has its long edges at the top and bottom and its short edges at the sides. Contrast with portrait presentation.

language

?.. A set of symbols, conventions, and rules that is used for conveying information.

leading. A printer's term for the distance between lines of type measured in points. It refers to the lead slug placed between lines of type in traditional typesetting.

library. System storage for generated form definitions and page definitions.

library resource (member). A named collection of records or statements in a library.

library resource name. A name by which an object may be called from a library by IBM Print Services Facility as part of a print job. Includes the two-character prefix for the type of object, such as P1 for page definitions, F1 for form definitions, or O1 for overlays (also known as resource name).

line data files. Files formatted for printing on line printers.

line printer

?.. A device that prints a line of characters as a unit. (I) (A) Synonymous with line-at-a-time printer. Contrast with page printer.

lines per inch (lpi). (1) On a printer, a measurement of the number of lines per vertical inch of paper. (2) A unit of measure for specifying the baseline increment.

local name

?.. A name for a suppression, an overlay, or a font that is used only within the Page Printer Formatting Aid command stream. Contrast with user-access name.

location. A site within a data stream. A location is specified in terms of an offset in the number of structured fields from the beginning of a data stream, or in the number of bytes from another location within the data stream.

logical page

?.. (1) The area on a surface of a form that is formatted for printing. (2) A collection of data that can be printed on one side of a sheet of paper. See also form and page.

logical page origin. (1) The user-defined point that acts as a reference for all positioning of printed material on the page. (2) The point nearest the hardware origin where printing can occur.

Logical unit (L-unit)

?.. A unit of linear measurement expressed with a unit base and units per unit-base value. For example, in Page Printer Formatting Aid, 1 logical unit = 1/240 inch (unit base = 10 inches, units per unit base = 2400).
ipi.  Lines per inch.

lowercase

? Pertaining to small letters as distinguished from capital letters. Examples of small letters are a, b, and g. Contrast with uppercase.

L-unit

? A unit of linear measurement expressed with a unit base and units per unit-base value. For example, in MO:DCA and IPDS architectures, the following L-units are used:

- 1 L-unit = 1/1440 inch ; (unit base = 10 inches, ; units per unit base = 14400)
- 1 L-unit = 1/240 inch ; (unit base = 10 inches, ; units per unit base = 2400)

Synonymous with logical unit.

M

MAT2OF5. A parameter that specifies a bar code type of Matrix 2-of-5.

media origin. The first hardware addressable point on the physical page. The point from which the logical page origin is positioned by the medium map.

medium

? The physical material (for example, paper) on which data is printed. See also form.

medium map. An internal object whose structured fields control the physical sheets of a printout, including the choice of duplex printing, the beginning print position, and the paper source to use. Controlled by a COPYGROUP command in a Page Printer Formatting Aid command stream.

medium overlay. Synonym for overlay.

mixed data files. Files consisting of composed and uncomposed portions.

mm. Millimeters.

MOD. A parameter that specifies additional processing information about the bar code symbol to be generated. Refer to Data Stream and Object Architecture: Bar Code Object Content Architecture Reference (S544-3766) for more information.

Mixed Object Document Content Architecture (MO:DCA)

? An architected, device-independent data stream for interchanging documents.


module. In a bar code symbology, the nominal width of the smallest element of a bar or space. Actual bar code symbology bars and spaces can be a single module wide or some multiple of the module width. The multiple need not be an integer.

MODWIDTH. A parameter that specifies the width of the smallest defined bar code element, using mils (thousandths of an inch).

MSI. A parameter that specifies a bar code type of modified Plessey code.

multiple up. The printing of more than one page on a single side of a sheet of paper.

N

name. A table heading for architecture syntax. The entries under this heading are short names that give a general indication of the contents of the construct.

noncompatibility mode. The use of table reference character (TRC) numbers not compatible with a line printer.

normal duplex printing

? Duplex printing for sheets that are to be bound on the long edge of the paper, regardless of whether the printing is portrait or landscape. Contrast with tumble duplex printing.

N_UP. The printing of more than one logical page on a single side of a medium.

O

object

? A collection of data referred to by a single name. Form definitions and page definitions stored in a library are resources.

offset. A table heading for architecture syntax. The entries under this heading indicate the numeric displacement into a construct. The offset is measured in bytes and starts with byte zero. Individual bits can be expressed as displacements within bytes.

orientation
The angular distance a presentation space or object area is rotated in a specified coordinate system, expressed in degrees and minutes. For example, the orientation of printing on a physical medium, relative to the $X_m$ axis of the $X_m, Y_m$ coordinate system.

**Origin.** A picture element (pel)

**Outline Font.** A shape technology in which the graphic character shapes are represented in digital form by a series of mathematical expressions that define the outer edges of the strokes. The resultant graphic character shapes can be either solid or hollow.

**Overlay.** A collection of predefined data such as lines, shading, text, boxes, bar codes, or logos, that can be merged with variable data on a page during printing. See electronic overlay.

**Overlay Generation Language (OGL).** A programming language used to produce electronic overlays.

**Page.**

- (1) A collection of data that can be printed on one side of a sheet of paper or a form. (2) The boundary for determining the limits of printing. See also logical page and physical page.

**Page Definition.** A resource containing a set of Page Printer Formatting Aid formatting controls for printing pages of data. Includes controls for number of lines per printed sheet, font selection, print direction, and mapping of individual fields in the data to positions on the printed sheets.

**Page Ejection.** The point at which the printer finishes printing on one sheet and moves to the beginning of the next sheet.

**Page Format.** A subset of a page definition, containing all the same controls for formatting printed output as a page definition. Includes controls for number of lines per printed sheet, font selection, print direction, and mapping of individual fields in the data to positions on the printed sheets.

**Page Mode.** The mode of operation in which an AFP printer can accept a page of data from a host processor to be printed on an all-points-addressable output medium. Printed data can consist of pages composed of text, images, overlays, and page segments. Contrast with compatibility mode.

**Page Printer.** A device that prints a page at a time. Contrast with line printer.

**Page Printer Formatting Aid for AIX (Page Printer Formatting Aid).** An IBM licensed program that allows you to create and store form definitions and page definitions, which are resource objects for print-job management. By writing a command stream specifying form definitions, page definitions, or both, for executing Page Printer Formatting Aid, you can store the objects specified in the library. These objects can then be used to format printed output.

**Page Segment.**

- (1) An object that can contain text and images and be included at any addressable point on a page or electronic overlay. It assumes the environment of an object it is included in. (2) A library resource that contains the definition of a page segment. Contrast with electronic overlay.

**Parameter.**

- (1) A variable that is given a constant value for a specified application and that may denote the application. (1) (A) (2) In Page Printer Formatting Aid, the values specified for a subcommand.

**Partition.**

- (1) Dividing the medium presentation space into a specified number of equal-sized areas in a manner determined by the current physical media. (2) In FD:OCA, a conceptual subdivision of a string of data fields. A partition can be further divided into subpartitions.

**Pel.** Picture element.

**PELS.** In Page Printer Formatting Aid, a unit of measure under the SETUNITS command. See also logical unit.

**Physical Page.**

- A single surface (front or back) of a sheet. See also form and page.

**Picture Element.**

- (1) In computer graphics, the smallest element of a display space that can be independently assigned color and intensity. (T) (2) The smallest area that can be individually toned by the printer.

**PMF.** Print Management Facility
point. In printing, a unit of about 1/72 of an inch used in measuring typographical material, for example: 10-point Helvetica. There are 12 points to a pica.

portrait presentation

?Portrait presentation is the position of a printed sheet that has its short edges at the top and bottom and its long edges at the sides. Contrast with landscape presentation.

position. The location specified for a line or field on the output page.

POSTNET. A parameter that specifies a bar code type of POSTal Numberic Encoding Technique (United States Postal Service), and defines specific values for the BSD module width, element height, height multiplier, and wide-to-narrow ratio fields.

PPFA. Page Printer Formatting Aid.

preprinted form

?A sheet of paper containing a preprinted design of constant data. Variable data can be merged with the constant data on such a form. See also electronic overlay, forms flash.

print line. A single line of text. In the formatting of line data, it refers to the output generated by one data record. Governed by the PRINTLINE command.

Print Management Facility (PMF). A program that can create fonts, segments, page definitions, and form definitions.

Print Services Facility (PSF). A program that produces printer commands from the data sent to it.

printer–attached device. Either a preprocessor or postprocessor attached to the printer.

PSF. Print Services Facility.

R

range. A table heading for architecture syntax. The entries under this heading give numeric ranges applicable to a construct. The ranges can be expressed in binary, decimal, or hexadecimal. The range can consist of a single value.

raster. (1) In computer graphics, a predetermined pattern of lines that provides uniform coverage of a display space. (T) (2) In AFP printers, an on-or-off pattern of electrostatic images produced by the laser print head.

RASTER NORASTER subcommand. A subcommand that specifies whether an overlay is to be kept in the printer (3800 only) as raster data.

RATIO. A parameter that specifies the ratio of the wide-element dimension to the narrow-element dimension whenever two different size elements exist.

ratio. The relationship in quantity, amount, or size between two or more things.

record. (1) In programming languages, an aggregate that consists of data objects, possibly with different attributes, that usually have identifiers attached to them. In some programming languages, records are called structures. (l) (2) A set of data treated as a unit. (T) (3) A set of one or more related data items grouped for processing.

RM4SCC. A parameter that specifies a 4-state customer code defined by the Royal Mail Postal Service of England for bar coding postal code information. See Royal Mail 4 State Customer Code.

resource

?A collection of printing instructions, and sometimes data to be printed, that consists entirely of structured fields. A resource object is stored as a member of a library and can be called for by IBM Print Services Facility when needed. The different resource objects are: page segments, overlays, form definitions, and page definitions.

RNORMAL. Rotated normal. A Page Printer Formatting Aid parameter that specifies the type of duplex printing. It means the tops of both sides of a duplex-printed sheet are toward the same physical edge of the sheet, for side binding of the document. Used with landscape-presentation pages.

rotation. The orientation of the characters of a font with respect to the baseline.

Royal Mail 4 State Customer Code (RM4SCC)

?A two-dimensional bar code symbology developed by the United Kingdom's Royal Mail postal service for use in automated mail-sorting processes.

RTUMBLE. Rotated tumble. A Page Printer Formatting Aid parameter that specifies a type of duplex printing. It means the top of one side of a duplex-printed sheet and the bottom of the other are toward one physical edge of the sheet, for top binding of the document. Used with landscape-presentation pages.

rule

?A solid line of any line width.
S

SBCS. Single-byte character set.

scanner. In bar codes, an electronic device that converts optical information into electrical signals. Sometimes called a reader or decoder.

segment. (1) A collection of composed text and images, prepared before formatting and included in a document when it is printed. See page segment. (2) The resource that contains the structured-field definition of a page segment.

sheet

?. A single piece of paper. For cut-sheet printers, a synonym for form.

shift-in and shift-out characters (SOSI). Characters used to delimit literals in Page Printer Formatting Aid command streams: X'0E' and X'0F'.

simplex printing

?. A method used to print data on one side of a sheet; the other side is left blank. Contrast with duplex printing.

single-byte character set. A character set whose codes require a single byte of data. The character set used for English is an example.

skip-to-channel control. A line printer control appearing in line data. Allows space to be left between print lines. Compatible with page printers when the data is formatted by page definitions.

space

?. In bar codes, the lighter element of a printed bar code symbol, usually formed by the background between bars.

space width. In bar codes, the thickness of a bar code symbol space measured from the edge closest to the symbol start character to the trailing edge of the same space.

SSASTERISK. A parameter that specifies whether an asterisk is to be generated as the HRI for CODE39 bar code start and stop characters.

start-stop character or pattern. In bar codes, a special bar code character that provides the scanner with start and stop reading instructions as well as a scanning direction indicator. The start character is normally at the left end and the stop character at the right end of a horizontally-oriented bar code symbol.

structured field

?. A self-identifying string of bytes and its data or parameters.

subcommand. (1) In Page Printer Formatting Aid, the next level of control below commands. (2) A request for an operation that is within the scope of work requested by a previously issued command.

subgroup. A subset of a form definition that is used to reprint the same page of data more than once. Subgroups provide for variations in the same page of data within one print job. Modifications that distinguish one subgroup from another are number of copies, type of duplex printing, inclusion of overlays, inclusion of suppressions, and (only for the 3800 printer) forms flash. A set of modifications within a copy group that applies to a certain number of copies of a form. A copy group can contain more than one subgroup.

subpage. A part of a logical page on which line data may be placed. In the page definition, multiple subpages can be placed on a physical page based on changes in the print data.

suppression. The electronic equivalent of a spot carbon, preventing selected data from being printed on certain copies.

symbology

?. A bar code language. Bar code symbologies are defined and controlled by various industry groups and standards organizations. Bar code symbologies are described in public domain bar code specification documents. Synonymous with bar code symbology. See also Canadian Grocery Product Code (CGPC), European Article Numbering (EAN), Japanese Article Numbering (JAN), and Universal Product Code (UPC).

syntax

?. The rules governing the structure of a construct.

T

table reference character (TRC). Usually, the second byte on a line in the user's data. This byte contains a value (0 - 126) that is used to select a font to be used to print that line.

tate. The Japanese word for top-to-bottom, as applied to the formatting of writing and printing. The traditional arrangement of Japanese kanji characters on the page. Pronounced ta-tay.

text

?. A graphic representation of information on an output medium. Text can consist of alphanumeric characters
and symbols arranged in paragraphs, tables, columns, and other shapes.

**TRC.** Table reference character.

**tumble duplex printing**

?. Duplex printing for sheets that are to be bound on the top, as is often done for legal documents. The top of one side of each sheet is at the same edge as the bottom of the other side. Contrast with normal duplex printing.

**triplet**

?. A three-part self-defining variable-length parameter consisting of a length byte, an identifier byte, and one or more parameter-value bytes.

**type.** A table heading for architecture syntax. The entries under this heading indicate the types of data present in a construct. Examples include: BITS, CHARCODE, SBIN, UBIN, UNDF.

**TYPE.** A parameter that specifies the kind of bar code symbol to be generated. For example, CODE39, MSI, UPCA, UPCE, and so on.

**type font.** See font.

**UPC.** See Universal Product Code.

**UPCA.** A parameter that specifies a bar code type of Universal Product Code (United States) and the Canadian Grocery Product Code, Version A.

**UPCE.** A parameter that specifies a bar code type of Universal Product Code (United States) and the Canadian Grocery Product Code, Version E.

**UPC2SUPP.** A parameter that specifies a bar code type of Universal Product Code (United States) two-digit Supplemental (periodicals).

**UPC5SUPP.** A parameter that specifies a bar code type of Universal Product Code (United States) five-digit Supplemental (paperbacks).

**uppercase**

?. Pertaining to capital letters. Examples of capital letters are A, B, and C. Contrast with lowercase.

**user-access name**

?. The library resource name of a font or an overlay, less its two-character prefix. Contrast with local name.

**USS.** See Uniform Symbol Specification.

**W**

**width.** In Page Printer Formatting Aid, refers to the horizontal dimension of a logical page, is specified in the page definition, and is controlled by the WIDTH subcommand.

**X**

**x-coordinate**

?. The horizontal or inline position that defines a page origin or the starting point of a line or field.

**Y**

**y-coordinate.** The vertical or baseline position that defines a page origin or the starting point of a line or field.
Index

A
ACROSS parameter
   in COPYGROUP command 128
   in FIELD command 163
   in FORMDEF command 137
   in PAGEDEF command 175
   in PAGEFORMAT command 178
   in PRINTLINE command 184
   in TRCREF command 189
ADJUST subcommand
   in COPYGROUP command 127
   in FORMDEF command 135
AJPCHARS parameter
   OS/400 202
AFTER parameter
   in CONDITION command 155
AIX
   PPFA system dependencies 199
alphabet characters 120
alphanumeric characters
   maximum number allowed 121
application considerations for line data
   OS/400 204
arrangement of N_UP partitions 98
AS/400 printer file parameters
   OS/400 206

B
BACK parameter
   in COPYGROUP command 129
   in FIELD command 163
   in FORMDEF command 138, 142
   in PAGEDEF command 175
   in PAGEFORMAT command 178
   in PRINTLINE command 184
   in TRCREF command 190
BACK subcommand
   duplexing pages 23
   in SUBGROUP command 148
   rules 24
bar code data 223
   BCOCA characters 223
   BCOCA code points 223
   characters 223
   code 128 code page
      CPGID 223
   code pages 223
   type styles 223
bar codes
   Bar Code Object Content Architecture (BCOCA) 223
BARCODE subcommand
   AIX 166
   BCOCA 166
   in FIELD command 166
   MVS 166
   supplemental information 223
   VM 166
   VSE 166
baseline direction
   definition 8
basic N_UP printing
   compared to enhanced N_UP 97
   examples
      normal duplex 104
      tumble duplex 106
      using INVOKE and OVERLAY 102
   list of printers 97
   subcommands and parameters enabled 101
BEFORE parameter
   in CONDITION command 155
BIN subcommand
   in COPYGROUP command 127
   in FORMDEF command 136
   in SUBGROUP command 148
blank characters 120
blank lines in command streams 121
blank truncation, conditional processing 86
BOTH subcommand
   duplexing pages 23
   in SUBGROUP command 148
bounded-box fonts
   description 9

C
carriage control characters
   ANSI 210
   machine code 210
   OS/400 210
   relationship with START subcommand 83
   specified for printline 180
CHANGE parameter
   in CONDITION command 154
CHANNEL subcommand
   in PRINTLINE command 180
character sets
   definition 120
   types 120
characters
   alphabetic 120
   blank 120
   number allowed 121
characters (*continued*)
  numeric 120
  shift-out/shift-in codes 120
  special 120
Codabar 223, 225, 228, 231
Code 128 223, 225, 226, 228, 231
Code 39 223, 224, 228
color
  specified for field 164
  specified for print line 184
COLOR subcommand
  in FIELD command 164
  in PRINTLINE command 184
command delimiters
  definition 121
command sequence
  for form definitions 125
  for page definitions 151
command stream, examples of
  defining literals 122
  defining logical page size 37
  for form definitions 17
  for page definitions 35
  programmer comments 122
  units of measurement 123
commands
  abbreviating 119
  CONDITION 152
  COPYGROUP 126
  definition 11
  ENDSUBPAGE 158
  FIELD 158
  FONT 170
  FORMDEF 135
  notation xvii, 124
OVERLAY
  in form definitions 144
  in page definitions 172
PAGEND 173
PAGEFORMAT 175
PRINTLINE 178
SEGMENT 186
sequence
  for form definitions 125
  for page definitions 151
SETUNITS
  in form definitions 145
  in page definitions 187
SUBGROUP 146
SUPPRESSION 150
TRCREF 188
comments in command streams
  delimiters 122
  location 122
comparisontype parameter
  in CONDITION command 154
CONDITION command
  blank truncation, consideration 86
  interaction with CHANNEL subcommand 81
  interaction with REPEAT subcommand 81
  name 152
  selecting copy groups and page formats 84
  short form 152
  syntax 152
  using with enhanced N_UP 113
  variable-length records, consideration 86
conditional processing
  blank truncation 86
  considerations 79
  copy group, selection 74, 84
  description 73
  duplex output example 87
  examples 86
  offset stacking (jog) example 86
  page format, selection 75, 84
  paper (bin) selection example 89
  PRINTLINE command 75
  record reprocessing 78
  record reprocessing example 88
  repeated printlines examples 93
  restrictions 79
  rules 79
  setting the environment 74
  subpage, description 77
  using multiple conditions, examples 89
  variable-length records 86
  versus normal line data processing 73
  WHEN CHANGE always false 83
constant forms
  definition 22
  example 22
constant overlays
  placement 107
CONSTANT parameter
  in COPYGROUP command 133
  in FORMDEF command 142
CONSTANT subcommand
  enhanced N_UP printing example 109
  example 22
  in COPYGROUP command 128
  in FORMDEF command 136
continuous forms
  example 28
  narrow 29
  rules 29
  wide 29
control record
  Invoke Data Map (IDM) structured field
    in PAGEFORMAT command 176
conventions
  highlighting xvi
  notational xvii, 124
copied in SUBGROUP command 147

COPIES subcommand
example 22
in SUBGROUP command 147
using with enhanced N_UP 113
copy group
defined with COPYGROUP command 126
definition 17
purpose 17
selection, conditional processing 74, 84
starting or restarting 84

COPYGROUP command
specifying the N_UP subcommand
basic N_UP printing 101
enhanced N_UP printing 107
syntax 126
copygroup parameter
in CONDITION command 155

CRTPRTF CL command
*AFPDSLINE 201
*LINE 201
DEVTYPE values 201

CTLCHAR values
OS/400 201

CURRENT parameter
in FIELD command 162, 163
CUT parameter
in COPYGROUP command 132
in FORMDEF command 141

CVTPPFASRC Command
considerations 215

CVTPPFASRC command (OS/400)
required PPFA parameters 216
subcommands 216

D
data file types
line 6
mixed 6
MO:DCA-P 6
unformatted ASCII 7
data lengths 223

DBCS subcommand
in FONT command 171
delimiters
definition 121
device type considerations
OS/400 205
DEVTYPE values
*AFPDSLINE 201
*LINE 201
OS/400 201
differences in measurements and REPEATs 221
direction
additive 189
baseline, definition 8
change print direction of logical page 39
inline, definition 8
of fonts 55
relationship to duplex 28
specifying
for fields 54
for fonts using TRCs 189
for lines 54
in a page definition 39

DIRECTION subcommand
changing logical page print direction 39
example 31, 39, 54
in COPYGROUP command 128
in FIELD command 163
in FORMDEF command 136
in PAGEDEF command 174
in PAGEFORMAT command 177
in PRINTLINE command 184
in TRCREF command 189
using with enhanced N_UP 113
when to use 29
double-byte code characters (type G text) 161
down parameter
in COPYGROUP command 128
in FIELD command 163
in FORMDEF command 137
in PAGEDEF command 175
in PAGEFORMAT command 178
in PRINTLINE command 184
in TRCREF command 190
duplex printing
conditional processing example 87
definition 12
element of basic N_UP printing 104
in landscape presentation 26
in portrait presentation 26
possible combinations 28
specifying in form definition 23
specifying side modifications in subgroup 148
using BACK subcommand 23
using BOTH subcommand 23
using DUPLEX subcommand 128, 137
using FRONT subcommand 23

duplex subcommand
example 23, 26
in COPYGROUP command 128
in FORMDEF command 137
duplication parameter
in FIELD command 160
EAN 223, 224, 228, 230, 231

EBCDIC data
  blank characters 120
  shift-out/shift-in codes 120

ENDSUBPAGE command
  syntax 158
  enhanced N_UP printing
    compared to basic N_UP 97
    examples
      asymmetric pages 111
        using CONSTANT and OVERLAY 109
        using PLACE 108
      list of printers 97
      subcommands and parameters enabled 106
  error messages 237

extended color subcommands
  CIELAB 164, 185
  CMYK 164, 185
  HIGHLIGHT 164, 185
  in FIELD command 164
  in PRINTLINE command 185
  RGB 164, 185

E

FIELD command
  example 50, 55
  ordering 171
  rotating fonts 55
  syntax 170

FIELD subcommand
  example 56
    in FIELD command 161
    in PRINTLINE command 180
    in TRCREF command 189
    rotating data 56

fonts
  bounded-box 9
  in tate presentation 57
  naming in a page definition 50
  rotation of 55
  specified
    for field 161
    for printline 180
    for PRINTLINE command 52
    with FONT command 170
  unbounded-box 9

form definition
  command nesting 19
  command sequence 125
  contents of 4
  copy groups in 17
  defined with FORMDEF command 135
  defining overlays 21
  duplex printing
    using NORMAL 26
    using RTUMBLE 26
    using subgroups 24
    using TUMBLE 27
  example command streams 17
  examples
    asymmetric pages 112
    normal duplex 105
    tumble duplex 106
    using CONSTANT and OVERLAY 110
    using INVOKE and OVERLAY 103
    using PLACE 108
  logical pages 19
  page definition 3
  print jobs requiring 6
  specifying the N_UP subcommand 101
  steps for creating 3
  storage location 3
  formatting multiple applications pages on a single sheet 115
  formatting print lines 178

FORMDEF command
  specifying the N_UP subcommand
    basic N_UP printing 101
    enhanced N_UP printing 107
  syntax 135

F

field
  direction of 54
  outside record boundary of 155
  specified with FIELD command 158

FIELD command
  bar code, supplemental information 223
  example 55
  nesting in PRINTLINE commands 55
  specifying location 48
  syntax 158

FIELD parameter
  in PRINTLINE command 179

field processing
  combining data 68
  combining with overlay 61
  positioning fields 48
  rules 46
  selection of fields 48
  use of fixed text with 65

first line of data
  positioning 37
  specified in page definition 174
  specified in page format 177

fixed text
  example 65
    in page definition 65
    specified in FIELD command 160

FLASH subcommand
  in SUBGROUP command 147

FONT command
  example 50, 55
  ordering 171
  rotating fonts 55
  syntax 170

fonts
  bounded-box 9
  in tate presentation 57
  naming in a page definition 50
  rotation of 55
  specified
    for field 161
    for printline 180
    for PRINTLINE command 52
    with FONT command 170
  unbounded-box 9

form definition
  command nesting 19
  command sequence 125
  contents of 4
  copy groups in 17
  defined with FORMDEF command 135
  defining overlays 21
  duplex printing
    using NORMAL 26
    using RTUMBLE 26
    using subgroups 24
    using TUMBLE 27
  example command streams 17
  examples
    asymmetric pages 112
    normal duplex 105
    tumble duplex 106
    using CONSTANT and OVERLAY 110
    using INVOKE and OVERLAY 103
    using PLACE 108
  logical pages 19
  page definition 3
  print jobs requiring 6
  specifying the N_UP subcommand 101
  steps for creating 3
  storage location 3
  formatting multiple applications pages on a single sheet 115
  formatting print lines 178

FORMDEF command
  specifying the N_UP subcommand
    basic N_UP printing 101
    enhanced N_UP printing 107
  syntax 135
FORMDF parameter
OS/400 203
FORMDF restrictions
OS/400 214
forms flash
in SUBGROUP command 147
FRONT parameter
in COPYGROUP command 129
in FORMDEF command 138, 142
FRONT subcommand
duplexing pages 23
in SUBGROUP command 148
rules 24

H
HEIGHT subcommand
example 37
in FONT command 172
in PAGEdEF command 174
in PAGEFORMAT command 177
hexadecimal codes (type X text) 161
highlight color
naming in a page definition 48
HIGHLIGHT command
e xample 48
highlighting conventions xvi
horizontal parameter
in FIELD command 162

IBM 3800 Printing Subsystem
coexistence with the IBM 3835 and 3900 Page Printers 31
migration 31
IBM 3835 Page Printer
coeexistence with the IBM 3800 Printing Subsystem 31
specifying page presentation for 29
IBM 3900 Page Printer
coeexistence with the IBM 3800 Printing Subsystem 31
IBM 3900 Printing Subsystem
restrictions on printing area of sheet 20
IDM structured field
and the PAGEFORMAT command 176
IGC parameters
OS/400 213
Industrial 2-of-5 223, 224, 228, 231
inline direction
definition 8
specified for print line 184
specified in form definition
in COPYGROUP command 128
in FORMDEF command 136

INDEX
271

FORMDF parameter
OS/400 203
FORMDF restrictions
OS/400 214
forms flash
in SUBGROUP command 147
FRONT parameter
in COPYGROUP command 129
in FORMDEF command 138, 142
FRONT subcommand
duplexing pages 23
in SUBGROUP command 148
rules 24

H
HEIGHT subcommand
example 37
in FONT command 172
in PAGEdEF command 174
in PAGEFORMAT command 177
hexadecimal codes (type X text) 161
highlight color
naming in a page definition 48
HIGHLIGHT command
e xample 48
highlighting conventions xvi
horizontal parameter
in FIELD command 162

IBM 3800 Printing Subsystem
coexistence with the IBM 3835 and 3900 Page Printers 31
migration 31
IBM 3835 Page Printer
coeexistence with the IBM 3800 Printing Subsystem 31
specifying page presentation for 29
IBM 3900 Page Printer
coeexistence with the IBM 3800 Printing Subsystem 31
IBM 3900 Printing Subsystem
restrictions on printing area of sheet 20
IDM structured field
and the PAGEFORMAT command 176
IGC parameters
OS/400 213
Industrial 2-of-5 223, 224, 228, 231
inline direction
definition 8
specified for print line 184
specified in form definition
in COPYGROUP command 128
in FORMDEF command 136

INDEX
271
line spacing
  specified with SETUNITS command 188
LINEONE subcommand
  example 37
  in PAGEDEF command 174
  in PAGEFORMAT command 177
  positioning first line of data 37
LINESP subcommand
  in SETUNITS command 188
  positioning the first line of data 39
literals
  definition 122
  syntax 122
  used in TEXT subcommand 122
  used in WHEN subcommand 122
local name
  definition 21
logical page
  definition 7
  positioning 19
  size 36
  specifying height
    in page definitions 174
    in page format 177
  specifying the origin 19

M
MANUAL parameter
  in COPYGROUP command 127
  in FORMDEF command 136
MARGIN parameter
  in PRINTLINE command 181
Matrix 2-of-5 223, 225, 228, 231
measurement
  differences in repeated lines 221
  units specified with SETUNITS command
    in form definitions 145
    in page definitions 187
  units, described 123
MEDIA_INFO parameter
  in COPYGROUP command 132
  in FORMDEF command 141
medium overlay
  definition 114
  using with N_UP 114
messages and codes 237
mixed data
  description 6
MO:DCA-P data
  description 6
MOD parameter
  bar code type 227
  MOD value 227
modifications
  definition 10

modifications (continued)
  in SUBGROUP command 147
  MSI 223, 224, 228
multiple conditions, conditional processing
  examples 89
multiple-up printing
  compared to N_UP printing 7, 115
  conditional processing 77
  definition 57
  example 57
MVS
  multiple data sets, concatenating 195
  PPFA execution 195
  PPFA system dependencies 195

N
N_UP partitions
  arrangement 98
  definition 10, 97
N_UP printing
  basic description 97
  basic N_UP printing 97
  compared to multiple-up printing 7, 115
  enhanced N_UP printing 97
  examples
    asymmetric pages 111
    normal duplex 104
    tumble duplex 106
    using CONSTANT and OVERLAY 109
    using INVOKE and OVERLAY 102
    using PLACE 108
  list of printers 97
N_UP subcommand
  in COPYGROUP command 101, 107, 132
  in FORMDEF command 101, 107, 141
names
  character length allowed 121
commands
  BIN 148
  CONDITION 152
  COPYGROUP 127
  FONT 171
  FORMDEF 135
  OVERLAY 132, 141, 145, 149, 172
  PAGEDEF 173
  PAGEFORMAT 176
  SEGMENT 187
  SUPPRESSION 149, 150
library-resource 22
local 21
overlay 21
resource 21
subcommands
  FONT 180, 189
  SUPPRESSION 164
names (continued)
  user-access 21
narrow forms
  definition 29
NEWFORM parameter
  in CONDITION command 157
  using with enhanced N_UP 113
NEWSIDE parameter
  in CONDITION command 157
  using with enhanced N_UP 113
NEXT parameter
  in COPYGROUP command 129
  in FIELD command 163
  in FORMDEF command 138
  in PRINTLINE command 182
NORASTER subcommand
  in OVERLAY command 145
normal duplex
  definition 12
  example 104
normal line data processing
  versus conditional processing 73
NORMAL parameter
  in COPYGROUP command 128
  in FORMDEF command 137
notational conventions xvii, 124
numeric characters 120
numeric values
  definition 123
O
OFFSET parameter
  in COPYGROUP command 133
  in FORMDEF command 142
offset stacking
  example, conditional processing 86
  specified in copy group 129, 138
OFFSET subcommand
  example 19, 20
  in COPYGROUP command 130
  in FORMDEF command 138
  landscape presentation 20
  positioning a logical page 19
Optional PPFA parameters 216
origin
  logical page, definition 19
  specifying with OFFSET subcommand 19
OS/400
  AFPCHARS parameter 202
  application considerations for line data 204
  AS/400 printer file parameters 206
  carriage control characters 210
  CTLCHAR values 201
  CVTTPFASRC command 215
  device type considerations 205
OS/400 (continued)
  DEVTYPE values 201
  DEVTYPE("AFPDSLINE") 205
  DEVTYPE("LINE") 205
  FORMDF parameter 203
  IGC parameters 213
  INVMMAP (medium-map-name) DDS keyword 213
  LPI 46
  overflow 46
  PAGDFN parameter 202
  PPFA system dependencies 200, 215
  printing line data on a PSF printer 46
  required PPFA parameters 216
  restrictions for PAGDFN and FORMDF 214
  subcommands 216
  table reference characters (TRC) 212
  TBLREFCHR parameter 202
OTHERWISE subcommand
  in CONDITION command 157
  interaction with REPEAT subcommand 81
OVERLAY command
  example 21
  syntax
    in form definitions 144
    in page definitions 172
OVERLAY parameter
  in COPYGROUP command 134, 143
OVERLAY subcommand
  basic N_UP printing example 102
  enhanced N_UP printing example 109
  in COPYGROUP command 132
  in FORMDEF command 141
  in PRINTLINE command 183
  in SUBGROUP command 149
  using with enhanced N_UP 113
overlays
  combining with field processing 61
  examples of invoking 114
  form definition example 21
  identified with OVERLAY command 144
  in OVERLAY subcommand 183
  in page definitions 183
  names 21
  specified with PAGEDEF command 172
P
PAGDFN parameter
OS/400 202
PAGDFN restrictions
OS/400 214
page definition
  command nesting 36
  command sequence 151
  contents of 5
  defining font rotation 56
page definition (continued)
defining individual lines 43
equivalent command streams 35
field processing 47, 49
formatting lines 54
incorporating fixed text into 65
multiple-up printing 58
naming fonts 50
page formats in 35
page sequence, alteration of 59
positioning the first line of data 37
print jobs requiring 6
size of logical pages 36
specified with PAGEDEF command 173
steps for creating 3
page format 160
defined with PAGEFORMAT command 175
number of font names available 171
number of SEGMENT commands available in 186
purpose 35
restarting 155, 156
selecting with conditional processing 75, 84
starting or restarting 85
page overlay
definition 114
page presentation
equivalent command definition 28
specified in form definition 131, 139
page printers, use of line data with 41
page segment
identifying with SEGMENT command 186
page sequence
altering 59
PAGEDEF command
syntax 173
PAGEFORMAT command
and the IDM structured field 176
syntax 175
pageformat parameter
in CONDITION command 156
paper source
selection, conditional processing example 89
parameters
definition 11
derogetty order 119
PARTITION parameter
in COPYGROUP command 134
in FORMDEF command 143
PERFORATE parameter
in COPYGROUP command 132
in FORMDEF command 141
physical page
definition 7
PLACE subcommand (continued)
equivalent command 107
example 108
PLACE subcommand
in COPYGROUP command 133
in FORMDEF command 142
PORTRAIT parameter
in COPYGROUP command 131
in FORMDEF command 140
portrait presentation
definition 9
specifying on continuous-forms printers 28
with duplex printing 26
position
first line of data 37
specified for field 162
specified for printline 180
POSITION subcommand
in FIELD command 162
in PRINTLINE command 180
processing fields 48
POSTNET 223, 225, 228, 232
ppfa command stream
rules for creating 119
PRESENT subcommand
equivalent command example 28, 31
in COPYGROUP command 131
in FORMDEF command 139
using with enhanced N_UP 113
when to use 29
presentation
definition 9
equivalent command example 28
landscape 20
specifying for continuous-forms printers 28
print quality 18
specified in copy group 131, 140
specified in form definition 132, 140
printer file parameters
OS/400 206
printers
used in N_UP printing 97
printing
equivalent command basic N_UP 101
equivalent command basic N_UP example
normal duplex 104
tumble duplex 106
using INVOKE and OVERLAY 102
controlling direction 54
duplex 12
duplex example 23
enhanced N_UP 106
enhanced N_UP example
asymmetric pages 111
using CONSTANT and OVERLAY 109
using PLACE 108
line data 41
lines in two directions 54
multiple up 57
printing (continued)
on both sides 23
  See also duplex printing
two versions of same data 63
  using form definitions 3
  using page definitions 3
printing area
  for 3900 20
printline
definition 8
PRINTLINE command
  conditional processing 75
  defining individual lines 43
  example 43, 54
  in field processing 46
  printing direction of 54
  specifying fonts for 52
  syntax 178
PROCESSING subcommand
  in COPYGROUP command 132
  in FORMDEF command 141
PSF processing 155
publications
  related xvii
Q
QUALITY subcommand
  in COPYGROUP command 131
  in FORMDEF command 140
R
RASTER subcommand
  in OVERLAY command 145
RATIO subcommand
  in FONT command 172
record reprocessing
  conditional processing 78
  considerations 80
  example 88
  restriction 80
REPEAT subcommand
  in PRINTLINE command 179
REPLACE subcommand
  in FORMDEF command 140
  in PAGEDEF command 175
restarting page format 155, 156
restrictions
  conditional processing 79
FORMDF, OS/400 214
PAGDFN, OS/400 214
record reprocessing 80
return codes 237
RM4SCC 223, 225, 228, 232
RNORMAL parameter
  in COPYGROUP command 128
  in FORMDEF command 137
rotation
  definition of 9
  of fonts 55
  specified for font
    in FONT command 172
    in TRCREF command 190
  rotate 57
ROTATION parameter
  in COPYGROUP command 134
  in FORMDEF command 143
ROTATION subcommand
  in FONT command 172
  in TRCREF command 190
RTUMBLE parameter
  in COPYGROUP command 128
  in FORMDEF command 137
rules
  command nesting in form definitions 19
  command nesting in page definitions 36
  conditional processing 79
  continuous forms 29
  field processing 46
  for BACK subcommand 24
  for creating a command stream 119
  for FRONT subcommand 24
  for tokens 120
S
SAME parameter
  in PRINTLINE command 181, 183
SBCS subcommand
  in FONT command 171
SEGMENT command
  syntax 186
SEGMENT subcommand
  in PRINTLINE command 183
selecting a copy group
  conditional processing 74, 84
selecting a page format
  conditional processing 75, 84
SETUNITS command
  positioning the first line of data 38
  syntax
    in form definitions 145
    in page definitions 187
SHEET parameter
  in COPYGROUP command 129
  in FORMDEF command 138
single-byte code characters (type C text) 160
SPACE_THEN_PRINT
  in CONDITION command 153
START subcommand
  in CONDITION command 153
  in FIELD command 160
  relationship with CC and TRC fields 83
structured fields
  in line data 12
subcommands
  abbreviating 119
  ADJUST 127, 135
  BACK 148
  BARCODE 166
  BIN 127, 136, 148
  BOTH 148
  CHANNEL 180
  CIELAB 164, 185
  CMYK 164, 185
  COLOR 164, 184
  CONSTANT 128, 136
  COPIES 147
  DBCS 171
  definition 11
  DIRECTION 128, 136, 163, 174, 177, 184, 189
  DUPLEX 128, 137
  entry order 119
  FLASH 147
  FONT 161, 180, 189
  for extended color 164, 185
  FRONT 148
  HEIGHT 172, 174, 177
  HIGHLIGHT 164, 185
  INVOKE 129, 137
  JOG 129, 138
  LENGTH 153, 160
  LINEONE 174, 177
  LINESP 188
  N_UP 132, 141
  NORASTER 145
  OFFSET 130, 138
  OTHERWISE 157
  OVERLAY 132, 141, 149, 183
  PLACE 133, 142
  POSITION 162, 180
  PRESENT 131, 139
  PROCESSING 132, 141
  QUALITY 131, 140
  RASTER 145
  RATIO 172
  REPEAT 179
  REPLACE 140, 175
  RGB 164, 185
  ROTATION 172, 190
  SBCS 171
  SEGMENT 183
  SPACE_THEN_PRINT 153
  START 153, 160
  SUPPRESSION 149, 164

subcommands (continued)
  TEXT 160
  VFYSETUP 144
  WHEN 154
  WIDTH 173, 176
SUBGROUP command
  syntax 146
subgroups
  defined with SUBGROUP command 146
  maximum number within a copy group 147
  use in duplex printing 23
subpage
  conditional processing 77
  definition 7
  ending a subpage 158
SUBPAGE parameter
  in CONDITION command 155
suppression
  definition 63
  example 63
  specified
    for field 164
      in subgroup 149
    with SUPPRESSION command 150
SUPPRESSION command
  syntax 150
SUPPRESSION subcommand
  in FIELD command 164
  in SUBGROUP command 149
  using with enhanced N_UP 113
syntax
  literals 122
SYSIN data definition, MVS 195
system name
  definition 22

T

table reference characters (TRC)
  ANSI 212
  machine code 212
  OS/400 212
  relating to fonts with TRCREF command 188
  relationship with START subcommand 83
tate 57
TBLREFCHR parameter
  OS/400 202
text parameter
  in CONDITION command 154
TEXT subcommand
  in FIELD command 160
texttype parameter
  in FIELD command 160
tokens
  creating 120
tokens (continued)
   definition 120
TOP parameter
   in PRINTLINE command 182
TRCREF command
   syntax 188
tumble duplex
   definition 12
   example 106
TUMBLE parameter
   in COPYGROUP command 128
   in FORMDEF command 137
tumble printing
   example of basic N_UP printing 106
two sides, printing on 23
   See also duplex printing

U
unbounded-box fonts
   description 9
unformatted ASCII data
   basic controls 11
   description 7
   structured fields 12
units of measurement
   description 123
   specified with SETUNITS command
      in form definitions 145
      in page definitions 187
   specifying 123
unprintable area
   for 3900 20
UP parameter
   in FIELD command 164
   in PAGEDEF command 175
   in PAGEFORMAT command 178
   in PRINTLINE command 184
   in TRCREF command 190
UPC 223, 224, 228, 229, 230
user-access name
   definition 22

V
variable-length records, conditional processing 86
vertical parameter
   in FIELD command 162
VFYSETUP subcommand
   in FORMDEF command 144
   printer color setup in form definition 50
VM
   PPFA execution 195
   PPFA system dependencies 195
VSE
   PPFA execution 193

VSE (continued)
   PPFA system dependencies 193

W
WHEN subcommand
   at start of a page format 83
   in CONDITION command 154
   ordering 152
wide forms
   definition 29
width
   specifying in PAGEDEF command 173
   specifying in PAGEFORMAT command 176
WIDTH subcommand
   example 37
   in PAGEDEF command 173
   in PAGEFORMAT command 176
Readers' Comments — We'd Like to Hear from You

IBM Page Printer Formatting Aid: User's Guide
Publication No. S544-5284-02

Use this form to provide comments about this publication, its organization, or subject matter. Understand that IBM may use the information any way it believes appropriate, without incurring any obligation to you. Your comments will be sent to the author's department for the appropriate action. Comments may be written in your language.

Note: IBM publications are not stocked at the location to which this form is addressed. Direct requests for publications or for assistance in using your IBM system, to your IBM representative or local IBM branch office.

• Does the publication meet your needs? [ ] Yes [ ] No

• Did you find the information:
  [ ] Accurate?
  [ ] Easy to read and understand?
  [ ] Easy to retrieve?
  [ ] Organized for convenient use?
  [ ] Legible?
  [ ] Complete?
  [ ] Well illustrated?
  [ ] Written for your technical level?

• Do you use this publication:
  [ ] As an introduction to the subject?
  [ ] As a reference manual?
  [ ] As an instructor in class?
  [ ] As a student in class?

• What is your occupation?

Thank you for your input and cooperation.

Note: You may either send your comments by fax to 1-800-524-1519, or mail your comments. If mailed in the U.S.A., no postage stamp is necessary. For residents outside the U.S.A., your local IBM office or representative will forward your comments.

Comments:

Name

Address

Company or Organization

Phone No.
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17699 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 262 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17719 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 262 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17726 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 262 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17737 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 262 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17748 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 262 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17759 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 262 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17765 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 263 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17779 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 263 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17793 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 263 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17846 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 263 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17860 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 263 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17870 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 264 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17886 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 264 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17893 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 264 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17905 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 264 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17921 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 264 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17943 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 264 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17953 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 264 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17965 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 265 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17972 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 265 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17978 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 265 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 17996 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 265 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 18002 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 265 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 18024 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 265 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 18029 OF 'IB6P2MST'
+++EDF006W GT tag missing. (Page 265 File: IB6P2MST SCRIPT)
DSMMOM397I '.EDFGDEF' WAS IMBEDDED AT LINE 18041 OF 'IB6P2MST'