

Print Services Facility for z/OS  
Version 4.7

*User's Guide*



**Note**

Before using this information and the product it supports, read the information in [“Notices” on page 247](#).

This edition applies to the IBM® Print Services Facility Version 4 Release 7 Modification 0 for z/OS®, Program Number 5655-M32, and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces S550-0435-05.

© **Copyright International Business Machines Corporation 1983, 2022.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>Figures.....</b>	<b>ix</b>
<b>Tables.....</b>	<b>xiii</b>
<b>About this publication.....</b>	<b>xvii</b>
Using this publication.....	xvii
Understanding syntax notation.....	xviii
Related information.....	xviii
<b>How to send your comments to IBM.....</b>	<b>xxi</b>
If you have a technical problem.....	xxi
<b>Summary of changes.....</b>	<b>xxiii</b>
PSF for z/OS Version 4 Release 7.....	xxiii
PSF for z/OS Version 4 Release 6.....	xxiv
<b>Chapter 1. Introducing Advanced Function Presentation.....</b>	<b>1</b>
AFP architecture.....	1
AFP printing.....	2
Printing line data on AFP printers.....	3
Printing AFP data in different directions and character rotations.....	3
All-points addressability.....	5
<b>Chapter 2. Introducing Print Services Facility .....</b>	<b>7</b>
How PSF manages AFP printing.....	7
Components of PSF printing.....	7
PSF as an output writer (deferred-printing mode).....	9
PSF as an access method (direct-printing mode).....	9
Formatting data with PSF.....	10
<b>Chapter 3. Using resources.....</b>	<b>13</b>
Storing PSF resources.....	13
Searching for resources specified by a print job.....	14
Soft and hard resources.....	15
PSF system resource libraries.....	15
Fonts.....	19
Overview of AFP fonts.....	19
Obtaining and referencing fonts.....	23
Using AFP fonts.....	23
Form definitions.....	25
Using form definitions.....	26
Object containers.....	26
Object containers as data object resources.....	27
Common object containers supported by PSF.....	27
Overlays.....	30
Medium overlays.....	31
Page overlays.....	32
Using overlays.....	33
Merging data with an overlay.....	34

Using overlays on printers with different resolutions.....	35
Testing overlays.....	35
Medium overlays and page overlays on the same page.....	35
Page overlay rotation in page definitions and form definitions.....	36
Page definitions.....	36
Using page definitions.....	37
Page segments.....	37
Using page segments.....	38
Using multiple system page segment libraries.....	39
Testing page segments.....	39
Bar codes.....	39
Using BCOCA to produce bar code resources.....	41
Producing bar codes without BCOCA.....	41
Using bar code resources.....	41
Printing bar codes.....	41
Graphics.....	42
Using graphics.....	43
Printing graphic data.....	43
Image resources.....	43
Using image resources.....	44
Printing images.....	45
Printing images without IOCA.....	45
Compatibility among printers.....	46
Text.....	46
Using text.....	47
Printing text data.....	47
Combining character, image, graphics, and bar code data.....	48
Using resources with the distributed print function (DPF) of PSF.....	48
Using APSRMARK to mark resources.....	48
Using inline resources.....	49
PSF resources supplied by IBM.....	50

## **Chapter 4. Formatting and printing data..... 51**

Internal copy groups.....	51
Form definitions.....	52
Printing controls specified in copy groups.....	52
Using form definitions supplied with PSF.....	63
Page definitions.....	63
Defining page formats.....	63
Using page definitions supplied with PSF.....	69

## **Chapter 5. Printing different types of data..... 71**

Line data.....	71
Traditional line data.....	71
Record format line data.....	73
Supported encoding schemes for traditional and record format line data.....	73
Shift-out, shift-in (SOSI) codes.....	74
AFP structured fields included in line data.....	74
XML data.....	76
Element content.....	77
External entities.....	77
Supported encoding schemes and conversions.....	77
JCL parameters.....	78
MO:DCA-P data.....	78
Formatted data.....	79
Font selection.....	79
JCL parameters.....	79

AFP Conversion and Indexing Facility (ACIF).....	80
<b>Chapter 6. Using JCL for Advanced Function Presentation.....</b>	<b>81</b>
Determining printer defaults.....	81
Page printer defaults form.....	81
Creating JCL for direct-printing mode.....	82
Creating JCL for microfilm devices.....	83
Assigning OUTPUT statements to a DD statement.....	83
Specifying AFP parameters in the JCL.....	83
AFPSTATS.....	85
BURST.....	86
CHARS.....	86
CKPTPAGE.....	87
CKPTSEC.....	87
CLASS.....	87
COLORMAP.....	88
COMSETUP.....	88
CONTROL.....	89
COPIES.....	89
COPYCNT.....	90
DATAACK.....	90
DEST.....	91
DPAGELBL.....	92
DUPLEX.....	93
FCB.....	93
FLASH.....	93
FORMDEF.....	94
FORMLEN.....	95
FORMS.....	95
INTRAY.....	96
LINECT.....	96
NOTIFY.....	97
OFFSETXB.....	97
OFFSETXF.....	98
OFFSEYB.....	98
OFFSEYF.....	99
OUTBIN.....	100
OVERLAYB.....	100
OVERLAYF.....	100
PAGEDEF.....	100
PIMSG.....	101
PRMODE.....	102
PRTERORR.....	102
PRTQUEUE.....	103
RESFMT.....	103
SEGMENT.....	104
SUBSYS.....	104
SYSAREA.....	105
SYSOUT.....	105
TRC.....	106
UCS.....	106
USERLIB.....	107
USERPATH.....	107
Additional parameters to help in distributing output.....	108
<b>Chapter 7. Printing tasks and examples.....</b>	<b>111</b>
Printing on an AFP printer.....	111

Specifying a form definition.....	112
Using form definitions from a user library.....	113
Using inline form definitions.....	113
Using FORMDEF with COPIES or FLASH parameters in JCL.....	114
Specifying duplex printing.....	117
Specifying bins (paper source).....	117
Changing the paper source in a document.....	118
Using a forms flash on a 3800 printer.....	118
Printing with overlays.....	118
Printing a medium overlay.....	119
Printing a page overlay.....	119
Printing line data or XML data with page definition options.....	121
Specifying a page definition.....	121
Specifying print direction.....	123
Specifying lines per inch spacing.....	124
Specifying multiple-up printing.....	124
N_UP printing: printing multiple pages on a sheet.....	125
Suppressing print data.....	126
Specifying and selecting fonts.....	126
Changing formatting within a document.....	128
Using multiple copy groups or page formats.....	128
Printing page segments.....	130
Printing MO:DCA-P data.....	131
Specifying carriage control and table reference characters in line data.....	132
Using carriage control characters in line data records.....	132
Using table reference characters to select fonts.....	133
Merging data lines into a single print line.....	135
Example of merging data lines.....	135
Specifying shift-out, shift-in (SOSI) codes.....	135
Printing more than one copy.....	137
Bursting and stacking continuous-forms paper.....	139
Specifying whether you want error messages to be printed.....	139
Using TrueType and OpenType fonts.....	140
Processing Unicode Complex Text.....	141
Using extended code pages.....	141
Printing with resources from a user library.....	142
Printing with inline resources.....	143
Specifying that inline resources are stored above the bar.....	144
Specifying the AFPPARMS control statement on the OUTPUT statement.....	144
Specifying notification when the print job finishes printing.....	146
Inhibiting recovery of a print job.....	146
Specifying duplex-page offset.....	147
Transmitting a data set to an IBM i System.....	147
Specifying JCL parameters for microfilm jobs.....	148
Using microfilm setup resources from a user library.....	148
Using inline microfilm setup resources.....	148
Specifying color mapping tables.....	149
Using Color Mapping Table Resources from a User Library.....	149
Using Inline Color Mapping Table Resources.....	150
Specifying object container libraries in UNIX files.....	150
Finishing your output.....	151
Printing on printers that support multiple resolutions.....	152
Specifying printer checkpoints.....	153

## **Chapter 8. Using color mapping tables..... 155**

Understanding color mapping tables.....	155
Source groups.....	155

Target groups.....	156
Creating color mapping tables.....	156
Color Mapping Tool components.....	156
Using the Color Mapping Tool.....	156
Sample color mapping table source file.....	157
<b>Chapter 9. Using the AFP Reblocking Program.....</b>	<b>161</b>
How the AFP Reblocking Program works.....	161
Setting up the AFRREBLK profile.....	162
Uploading AFP files to z/OS.....	164
Reblocking data sets on z/OS.....	164
AFRREBLK command.....	164
Reblocking more than one data set at a time.....	165
Sample batch JCL.....	166
<b>Chapter 10. Obtaining AFP statistics.....</b>	<b>167</b>
AFPSTATS repository.....	167
Requesting an AFPSTATS report.....	167
Softcopy report.....	168
Format.....	168
Records.....	168
Example.....	171
Hardcopy report.....	171
Generating a hardcopy report.....	172
Example.....	172
<b>Chapter 11. Diagnosing incorrect printer output.....</b>	<b>173</b>
Messages are not visible.....	173
Messages do not help.....	173
<b>Appendix A. Form definitions supplied with PSF.....</b>	<b>175</b>
Form definition naming convention.....	175
Form definition for the 3800 printer.....	176
Form definitions for printers other than the 3800, PCL4, and PPDS printers.....	176
Form definitions for HP PCL4 and PPDS printers.....	179
Compatibility form definitions.....	180
Form definitions for special purpose jobs.....	181
Form definitions for finishing your output.....	182
Form definitions for printing PSF reports.....	183
<b>Appendix B. Page definitions supplied with PSF.....</b>	<b>185</b>
Page definitions for the 3800 printer.....	185
Page definitions for the 4224, 4230, 4234, 4247, and 6400 printers.....	187
Page definitions for HP-CL4 and PPDS printers.....	188
Page definitions for all other printers supported by PSF.....	189
Page definitions for printing PSF reports.....	192
Page definition line-spacing values and fonts.....	192
<b>Appendix C. AFPSTATS report.....</b>	<b>197</b>
Softcopy record details.....	197
Sample softcopy report.....	214
Sample hardcopy report.....	216
<b>Appendix D. Page-printer defaults form.....</b>	<b>221</b>
<b>Appendix E. Microfilm device considerations.....</b>	<b>223</b>

Images on a page.....	223
Graphics on a page.....	223
Form definitions.....	223
Using FOCA fonts.....	223
Page segments.....	224
Overlays.....	224
Specifying parameters.....	224
Additional parameters to help in distributing output.....	224
Printing constant forms.....	224
Other considerations.....	224
<b>Appendix F. Color and grayscale printing.....</b>	<b>225</b>
AFP color and grayscale solutions.....	225
Color printing without explicit color management.....	225
Resources included inline.....	225
Resources stored and managed centrally.....	226
Color printing concepts.....	226
Color spaces and ICC profiles.....	226
Gamut and rendering intent.....	227
Color mixing and calibration.....	228
Halftones and tone transfer curves.....	228
File size.....	229
Grayscale printing concepts.....	229
Color spaces and ICC profiles.....	229
Halftones.....	230
Tone transfer curve.....	230
Color management.....	230
ICC profiles.....	230
Rendering intents.....	230
Paper characteristics.....	231
AFP color management.....	231
Color management resources.....	232
Data objects.....	237
Resource library management.....	239
Tips and best practices.....	240
AFP resource installer.....	241
<b>Appendix G. Accessibility.....</b>	<b>243</b>
Accessibility features.....	243
Consult assistive technologies.....	243
Keyboard navigation of the user interface.....	243
Dotted decimal syntax diagrams.....	243
<b>Notices.....</b>	<b>247</b>
Terms and conditions for product documentation.....	248
IBM Online Privacy Statement.....	249
Policy for unsupported hardware.....	249
Minimum supported hardware.....	249
Trademarks.....	250
<b>Glossary.....</b>	<b>251</b>
<b>Bibliography.....</b>	<b>277</b>
Advanced Function Presentation (AFP).....	277
Text Processing.....	278
<b>Index.....</b>	<b>279</b>



---

# Figures

1. Components of AFP printing.....	3
2. Direction-rotation combinations.....	4
3. Copies of a memo that is printed in four inline directions.....	4
4. Direction-rotation combinations on a sample page.....	5
5. Printing two items in the same location on the page.....	5
6. PSF as an output writer.....	9
7. PSF in direct-printing mode.....	10
8. Resource storage.....	14
9. Sample page printed on an AFP printer.....	19
10. Example of text that uses outline fonts.....	20
11. Overlay on the sample page.....	31
12. Positioning of a medium overlay.....	32
13. Positioning of a page overlay.....	33
14. Electronic form (overlay) on the sample page.....	35
15. Medium overlays and page overlays on the same page.....	36
16. Page segments on the sample page.....	38
17. Bar codes on the sample page.....	40
18. Graphic on the sample page.....	42
19. Images on the sample page.....	44
20. Text on the sample page.....	47
21. Sample containing character data and other data.....	48
22. Relationship between the medium origin and the page origin.....	53
23. Duplex documents A and B specified as normal duplex.....	54

24. Duplex documents C and D specified as tumble duplex.....	55
25. N_UP printing partitions for various media.....	57
26. Copy group printed by using the constant-form function.....	59
27. Subgroups printed from one page of the data set.....	62
28. Pages printed in two directions on continuous-forms paper.....	65
29. One logical page divided into four subpages.....	68
30. Formatted and unformatted text.....	71
31. Traditional line data record.....	72
32. Record format line data record.....	73
33. Sample application program.....	76
34. Page-printer defaults form example.....	82
35. Additional JCL parameters for distributing output.....	85
36. Sample header page with additional distribution information.....	109
37. Output from three transmissions of a two-page data set.....	115
38. Output from four transmissions of a two-page data set.....	116
39. Positioning a page overlay.....	121
40. Printing four pages on two sheets.....	125
41. Output containing merged lines printed with a typographic font.....	135
42. IEBGENER example of merging two print lines.....	135
43. Sample output for COPIES=14.....	137
44. Sample output for COPIES=(, (1,3,2)).....	138
45. Example of the APFPARMS control statement.....	144
46. Sample JCL for Color Mapping Tool.....	157
47. Sample color mapping table source file (Part 1 of 3).....	158
48. Sample color mapping table source file (Part 2 of 3).....	159

49. Sample color mapping table source file (Part 3 of 3).....	160
50. Sample AFRREBLK profile.....	163
51. Reblocking numerous data sets in ISPF.....	166
52. APSWREBK sample batch job to execute AFRREBLK program.....	166
53. Placement of two subpages on a single physical sheet.....	187
54. Placement of multiple-up logical pages on the physical sheets.....	191
55. Softcopy record details (Part 1 of 18).....	197
56. Softcopy record details (Part 2 of 18).....	198
57. Softcopy record details (Part 3 of 18).....	199
58. Softcopy record details (Part 4 of 18).....	200
59. Softcopy record details (Part 5 of 18).....	201
60. Softcopy record details (Part 6 of 18).....	202
61. Softcopy record details (Part 7 of 18).....	203
62. Softcopy record details (Part 8 of 18).....	204
63. Softcopy record details (Part 9 of 18).....	205
64. Softcopy record details (Part 10 of 18).....	206
65. Softcopy record details (Part 11 of 18).....	207
66. Softcopy record details (Part 12 of 18).....	208
67. Softcopy record details (Part 13 of 18).....	209
68. Softcopy record details (Part 14 of 18).....	210
69. Softcopy record details (Part 15 of 18).....	211
70. Softcopy record details (Part 16 of 18).....	212
71. Softcopy record details (Part 17 of 18).....	213
72. Softcopy record details (Part 18 of 18).....	214
73. AFPSTATS softcopy report (Part 1 of 2).....	215

	74. AFPSTATS softcopy report (Part 2 of 2).....	216
■	75. AFPSTATS report (Page 1 of 8).....	217
■	76. AFPSTATS report (Page 2 of 8).....	217
■	77. AFPSTATS report (Page 3 of 8).....	217
■	78. AFPSTATS report (Page 4 of 8).....	218
■	79. AFPSTATS report (Page 5 of 8).....	218
■	80. AFPSTATS report (Page 6 of 8).....	219
■	81. AFPSTATS report (Page 7 of 8).....	219
■	82. AFPSTATS report (Page 8 of 8).....	220
	83. Page-printer defaults form.....	221

---

# Tables

1. SYS1.SAMPLIB members for PSF documentation updates.....	xviii
2. IBM operating systems for AFP printers and corresponding PSF products.....	7
3. PSF system resource libraries, resource types, and resource prefixes.....	16
4. AFP parameters in JCL.....	84
5. Carriage control characters.....	132
6. Allocation of AFPPARMS data set attributes.....	145
7. Data set values for default AFP object types.....	162
8. The basic format of the softcopy AFPSTATS report.....	168
9. Softcopy AFPSTATS records.....	169
10. Sections in the hardcopy AFPSTATS report.....	171
11. Form definition naming convention from left to right.....	175
12. Form definition for the 3800 printer.....	176
13. Form definitions for all printers other than the 3800, PCL4, and PPDS printers.....	176
14. Form definitions for printing envelopes on the 4028.....	177
15. Form definitions with a 0,0 offset.....	177
16. Form definitions for N_UP 2 printing.....	178
17. Form definitions for three-hole punched paper.....	178
18. Form definitions for rotating pages on the paper.....	178
19. Form definitions used to select print quality on a 64xx or 65xx printer.....	179
20. Form definitions for HP PCL4 and PPDS printers.....	179
21. Compatibility form definitions for AFCCU continuous-forms printers.....	180
22. N_UP compatibility form definitions for AFCCU continuous-forms printers.....	180
23. 3800 compatibility form definitions for cut-sheet printers.....	181

24. Form definitions supplied for special purposes.....	181
25. Form definitions for printers that support finishing.....	182
26. Additional form definitions for printers that support finishing.....	182
27. Form definitions for printing PSF reports.....	183
28. 3800 Model 1 FCBs and corresponding page definitions for 14.88 x 11-inch paper.....	185
29. Page definitions for 12 x 8.5-inch paper.....	185
30. Page definitions for 9.5 x 11-inch paper.....	186
31. Page definitions for 14.88 by 11-inch paper.....	186
32. Page definitions for multiple-up printing.....	186
33. Page definitions for continuous-forms paper 12 x 8.5 inches.....	187
34. Page definitions for continuous-forms paper 9.5 x 11 inches.....	187
35. Page definitions for continuous-forms paper 14.88 x 11 inches.....	188
36. Page definitions for A4 paper.....	188
37. Page definitions for B4 paper.....	188
38. Page definitions for cut-sheet letter paper.....	188
39. Page definitions for cut-sheet legal paper.....	189
40. Page definitions for A4 paper.....	189
41. Page definitions for B4 paper.....	189
42. Page definitions for letter and continuous-forms paper 12 x 8.5 inches or 9.5 x 11 inches.....	190
43. Page definitions for legal and continuous-forms paper 14.88 x 11 inches.....	190
44. Multiple-up page definitions.....	191
45. Page definitions for printing on three-hole punched paper.....	191
46. Page definitions for printing PSF reports.....	192
47. Cross-reference of line spacing and page definitions for the 3800 printer.....	192
48. Cross-reference of line spacing and page definitions for the 4224, 4230, 4234, 4247, and 6400 printers.....	193

49. Cross-reference of line spacing and page definitions for cut-sheet A4 paper for the PCL4 and PPDS printers.....	193
50. Cross-reference of line spacing and page definitions for cut-sheet B4 paper for the PCL4 and PPDS printers.....	193
51. Cross-reference of line spacing and page definitions for other printers.....	194
52. Cross-reference of line spacing and commonly used PSF monospaced fonts.....	195
53. AFP publications available from the AFP Consortium.....	277
54. AFP publications available from the IBM Documentation and the IBM Publications Center.....	277
55. AFP publications available from the IBM Publications Center.....	278
56. Text processing publications available from the IBM Publications Center.....	278





# About this publication

---

This publication provides information about using Print Services Facility (PSF) Version 4 Release 7.0 for z/OS (Program Number 5655-M32), from now on referred to as PSF.

This publication is written with the assumption that you are experienced with application programming and with Advanced Function Presentation (AFP) printers. In this publication, the word *printing* refers to presentation on paper, foils, labels, or microfilm.

## Using this publication

---

The information in this publication is for the application programmer and z/OS job submitter who use AFP resources, such as form definitions, page definitions, fonts, and color maps, and print on AFP printers.

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

- Chapter 1, “Introducing Advanced Function Presentation,” on page 1 summarizes AFP. It includes information about Mixed Object Document Content Architecture for Presentation (MO:DCA-P) architecture and printing AFP data.
- Chapter 2, “Introducing Print Services Facility,” on page 7 summarizes PSF. It includes information about deferred-printing mode, direct-printing mode, PSF data streams, and PSF resources. It also defines key PSF concepts.
- Chapter 3, “Using resources,” on page 13 describes the resources that PSF uses for printing.
- Chapter 4, “Formatting and printing data,” on page 51 describes form definitions and page definitions.
- Chapter 5, “Printing different types of data,” on page 71 summarizes how to print line data and XML records in a PSF format and how to code MO:DCA-P structured fields.
- Chapter 6, “Using JCL for Advanced Function Presentation,” on page 81 summarizes JCL commands and syntax rules to use when you are printing a job with PSF.
- Chapter 7, “Printing tasks and examples,” on page 111 presents detailed examples of PSF tasks you can use to print data sets.
- Chapter 8, “Using color mapping tables,” on page 155 specifies references to use when you are using color mapping tables for printers that support them and explains how to create color mapping tables.
- Chapter 9, “Using the AFP Reblocking Program,” on page 161 describes how to use the AFP Reblocking Program.
- Chapter 10, “Obtaining AFP statistics,” on page 167 describes how to obtain and print a report that contains statistics about the print file.
- Chapter 11, “Diagnosing incorrect printer output,” on page 173 describes how to diagnosis your problem when no messages are displayed or you do not understand the messages.
- Appendix A, “Form definitions supplied with PSF,” on page 175 describes form definitions that are supported and supplied with PSF.
- Appendix B, “Page definitions supplied with PSF,” on page 185 describes page definitions supported and supplied with PSF.
- Appendix C, “AFPSTATS report,” on page 197 contains softcopy record details for the AFPSTATS report and sample softcopy and hardcopy reports.
- Appendix D, “Page-printer defaults form,” on page 221 provides a page-printer defaults form for your use.
- Appendix E, “Microfilm device considerations,” on page 223 describes how you can use the same data set to print jobs on either paper output or microfilm.
- Appendix F, “Color and grayscale printing,” on page 225 describes the principles of color and grayscale printing, how various products can fit into color and grayscale solutions, and how you can integrate color and grayscale printing with your current operations or implement new color workflows.

- Appendix G, “Accessibility,” on page 243 describes the accessibility features available in z/OS.
- A notices, glossary, bibliography, and index are included. The notices contains important notices and a list of trademarks that are used in this publication. The Glossary contains terms and definitions that are related to PSF and the printing environment. The bibliography contains selected titles and order numbers of IBM publications that are related to PSF and the printing environment.

## Understanding syntax notation

The following rules apply to coding illustrations throughout this publication:

- Bold highlighting 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 DUPLEX and BLOCK.
- Variable data is printed in italics. Enter specific data to replace the characters in italics; for example, for PRT`nnnn` you might enter PRT0002. Italics also identify the names of publications.
- Monospacing 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 actually type.
- Do not enter the following symbols as part of a parameter or option:
  - Vertical Bar |
  - Underscore \_\_\_\_
  - Brackets [ ]
  - Braces { }
  - Ellipsis ...
- A vertical bar between two values means that you select only one of the values.
- An underscored value means that if an option is not specified, the underscored value, called the default, is used.
- Brackets around a value mean that you do not need to select the value; the value is optional.
- Braces around a value mean that you must select one of the mutually exclusive values. For example, { THIS | THAT }
- An ellipsis that is following a command or set of commands indicates the command or set of commands can be repeated.

## Related information

Publications that are referred to in this document or that contain more information about AFP and related products are listed in the “Bibliography” on page 277. For information about all z/OS product publications, see *z/OS Information Roadmap*.

For more information about z/OS and PSF for z/OS, see these web pages:

- [z/OS home page \(www.ibm.com/systems/z/os/zos\)](http://www.ibm.com/systems/z/os/zos)
- [IBM Documentation \(www.ibm.com/docs/en\)](http://www.ibm.com/docs/en)

To obtain the latest documentation updates for PSF for z/OS, see the appropriate SYS1.SAMPLIB members in Table 1 on page xviii.

Table 1. SYS1.SAMPLIB members for PSF documentation updates	
Member	Publication
<b>APSGADP7</b>	<i>PSF for z/OS: AFP Download Plus</i>
<b>APSGCUS7</b>	<i>PSF for z/OS: Customization</i>
<b>APSGDGN7</b>	<i>PSF for z/OS: Diagnosis</i>

<i>Table 1. SYS1.SAMPLIB members for PSF documentation updates (continued)</i>	
<b>Member</b>	<b>Publication</b>
<b>APSGDLG7</b>	<i><a href="#">PSF for z/OS: Download for z/OS</a></i>
<b>APSGMAC7</b>	<i><a href="#">PSF for z/OS: Messages and Codes</a></i>
<b>APSGSEC7</b>	<i><a href="#">PSF for z/OS: Security Guide</a></i>
<b>APSGUSR7</b>	<i><a href="#">PSF for z/OS: User's Guide</a></i>



## How to send your comments to IBM

---

We invite you to submit comments about the z/OS product documentation. Your valuable feedback helps to ensure accurate and high-quality information.

**Important:** If your comment regards a technical question or problem, see instead [“If you have a technical problem”](#) on page xxi.

Submit your feedback by using the appropriate method for your type of comment or question:

### Feedback on z/OS function

If your comment or question is about z/OS itself, submit a request through the [IBM RFE Community](http://www.ibm.developerworks/ref) (<http://www.ibm.developerworks/ref>).

### Feedback on IBM Documentation function

If your comment or question is about the IBM Documentation functionality, for example search capabilities or how to arrange the browser view, send a detailed email to IBM Documentation Support at [ibmdoc@us.ibm.com](mailto:ibmdoc@us.ibm.com).

### Feedback on the z/OS product documentation and content

If your comment is about the information that is provided in the z/OS product documentation library, send a detailed email to [mhvrcfs@us.ibm.com](mailto:mhvrcfs@us.ibm.com). We welcome any feedback that you have, including comments on the clarity, accuracy, or completeness of the information.

To help us better process your submission, include the following information:

- Your name, company/university/institution name, and email address
- The following deliverable title and order number: *PSF for z/OS: User's Guide*, S550-0435
- The section title of the specific information to which your comment relates
- The text of your comment.

When you send comments to IBM, you grant IBM a nonexclusive authority to use or distribute the comments in any way appropriate without incurring any obligation to you.

IBM or any other organizations use the personal information that you supply to contact you only about the issues that you submit.

## If you have a technical problem

---

If you have a technical problem or question, do not use the feedback methods that are provided for sending documentation comments. Instead, take one or more of the following actions:

- Go to the [IBM Support Portal](http://support.ibm.com) ([support.ibm.com](http://support.ibm.com)).
- Contact your IBM service representative.
- Call IBM technical support.



# Summary of changes

---

## PSF for z/OS Version 4 Release 7

---

This content includes terminology, maintenance, and editorial changes to information previously presented in *PSF for z/OS: User's Guide*, S550-0435-05. Technical additions or changes to the text and illustrations are indicated by a vertical line to the left of the change.

### General changes (December 2019)

- References to PSF for z/OS 4.5 and z/OS 2.1 are removed because those releases are no longer in service.

### New information (June 2021)

- APAR OA59773
  - [“Sample batch JCL” on page 166](#)

### New information (December 2019)

- Inline resource storage above the bar
  - [“Specifying that inline resources are stored above the bar” on page 144](#)
  - [“Specifying the AFPPARMS control statement on the OUTPUT statement” on page 144](#)
- MO:DCA support of references to Scalable Vector Graphics (SVG) secondary resources
  - [“Scalable vector graphics resource object” on page 30](#)

### Changed information (January 2022)

- APAR OA62033
  - [“FORMLEN” on page 95](#) is updated with a paragraph about using the FORMLEN parameter with Download for z/OS and AFP Download Plus.

### Changed information (June 2021)

- APAR OA59773
  - [“Using extended code pages” on page 141](#)
  - [“How the AFP Reblocking Program works” on page 161](#)
  - [“Softcopy record details” on page 197](#)
- Examples in these topics have been corrected:
  - [“AFP structured fields included in line data” on page 74](#)
  - [“Using multiple copy groups or page formats” on page 128](#)
  - [“Printing page segments” on page 130](#)
  - [“Printing MO:DCA-P data” on page 131](#)
  - [“Printing more than one copy” on page 137](#)

### Changed information (December 2019)

- Inline resource storage above the bar

- [“Using inline resources” on page 49](#)
- [“Specifying AFP parameters in the JCL” on page 83](#)
- [“Glossary” on page 251](#)
- AFP statistics report enhancements
  - [Table 9 on page 169](#)
  - [Table 10 on page 171](#)
  - [“Generating a hardcopy report” on page 172](#)
  - [Table 46 on page 192](#)
  - [Appendix C, “AFPSTATS report,” on page 197](#)
  - [“Softcopy record details” on page 197](#)
  - [“Sample softcopy report” on page 214](#)
  - [“Sample hardcopy report” on page 216](#)
  - [Table 42 on page 190 is updated.](#)
- MO:DCA support of references to Scalable Vector Graphics (SVG) secondary resources
  - [“Object containers as data object resources” on page 27](#)
  - [“Common object containers supported by PSF” on page 27](#)
  - [“Glossary” on page 251](#)
- APAR OA54593
  - [“Using inline resources” on page 49](#)

### Deleted information (June 2021)

- An obsolete link for downloading extended code page files is removed from [“Using extended code pages” on page 141](#).

## PSF for z/OS Version 4 Release 6

---

This content includes terminology, maintenance, and editorial changes to information previously presented in *PSF for z/OS: User's Guide*, S550-0435-04.

### General changes

- References to Ricoh products, including "InfoPrint Manager" and "Ricoch ProcessDirector" are removed.
- "Distributed Print Function (DPF)", "PSF Direct", and "Workstation Print Manager (WPM)" are changed to "distributed print function (DPF)", "PSF direct", and "workstation print manager (WPM)".
- "InfoPrint AFP Resource Installer" is removed or changed to "AFP resource installer".

### New information

- MO:DCA AFP/Archive (AFP/A) interchange set, MO:DCA AFP/A, IS/3 interchange set, and Graphics Arts Function Set are new. See [“Components of PSF printing” on page 7](#), [“MO:DCA-P data” on page 78](#), [“Printing MO:DCA-P data” on page 131](#), [Figure 58 on page 200](#), and [“Glossary” on page 251](#).
- A new page that lists metadata objects is added to the AFPSTATS report. See META-LIST record in [Table 9 on page 169](#), Processing Summary section in [Table 10 on page 171](#), [Figure 55 on page 197](#), [Figure 56 on page 198](#), [Figure 58 on page 200](#), [Figure 63 on page 205](#), [Figure 71 on page 213](#), [Figure 73 on page 215](#), [Figure 74 on page 216](#), [Figure 77 on page 217](#), and [Figure 81 on page 219](#).
- A new page that lists print file extension information is added to the AFPSTATS report. See PRTFILEX record in [Table 9 on page 169](#), Print File Extension Information section in [Table 10 on page 171](#), [Figure 71 on page 213](#), [Figure 73 on page 215](#), [Figure 75 on page 217](#), and [Figure 76 on page 217](#).



## Changed information

- The details in the AFP Statistics (AFPSTATS) report are updated in [Chapter 10, “Obtaining AFP statistics,” on page 167](#).
- The page definition in [“Generating a hardcopy report” on page 172](#) is updated. See also [Table 46 on page 192](#).
- RESOURCEDO record descriptions in [Figure 62 on page 204](#), an EVENT record description in [Figure 63 on page 205](#), and a SUMM-NAMEX record description in [Figure 67 on page 209](#) are updated.
- The form definitions and page definitions are updated in [Figure 73 on page 215](#).

## Deleted information

- References to the Compatibility Fonts feature of PSF are removed because it is no longer a feature in PSF 4.6. See:
  - [“FOCA fonts” on page 20](#)
  - [“FOCA fonts” on page 23](#)



---

# Chapter 1. Introducing Advanced Function Presentation

To understand what Print Services Facility (PSF) for z/OS can do for you, you must first understand the relationship between Advanced Function Presentation (AFP) and PSF.

AFP is an architected system of hardware and software for creating, formatting, viewing, retrieving, printing, and distributing information about a wide variety of printer and display devices. First introduced in 1984 to support the IBM 3800 Model 3 high-speed printer, AFP now supports improved printing technology and functions.

The AFP architecture governs the creation and control of data types (such as text, font, image, graphics, bar code, fax, color, audio, and multimedia) so that computer output is more readable and attractive. AFP's specific interchange architecture, called Mixed Object Document Content Architecture (MO:DCA), makes information interchange possible among different operating systems that use different protocols. These operating systems include:

- AIX®
- IBM i
- Linux®
- VM
- VSE
- Windows
- z/OS

## AFP architecture

---

AFP's MO:DCA architecture is a device-independent data stream that governs the interchange of documents. Without such an architecture, information exchange is difficult and unpredictable. A subset of MO:DCA is Mixed Object Document Content Architecture for Presentation (MO:DCA-P), which defines presentation documents.

A mixed object document is the collection of data objects that comprise the document's content and the resources and formatting specifications that dictate the processing functions done on the content. The term *mixed* in the MO:DCA-P architecture refers both to the mixture of data objects and the mixture of document constructs that comprise the document's components.

A MO:DCA-P document can contain a mixture of presentation data objects. Each data object type has unique processing requirements. An Object Content Architecture (OCA) is established for each IBM data object to define its respective syntax and semantics. MO:DCA-P documents can contain data and data objects that are governed by these OCAs:

**Bar Code Object Content Architecture (BCOCA)**

Describes and generates bar code symbols.

**Color Management Object Content Architecture (CMOCA)**

Supports the color management information that is required to render presentation data.

**Font Object Content Architecture (FOCA)**

Supports the digital presentation of character shapes by defining their attributes, such as shape definitions, shape dimensions, and positioning information.<sup>1</sup>

**Graphics Object Content Architecture (GOCA)**

Represents pictures that are generated by a computer, commonly referred to as computer graphics.

---

<sup>1</sup> Unlike the other OCAs, font objects are not carried inside the MO:DCA-P data stream. However, the MO:DCA-P architecture does provide and carry references to external font objects.

**Image Object Content Architecture (IOCA)**

Represents image information such as scanned pictures.

**Presentation Text Object Content Architecture (PTOCA)**

Defines text information.

MO:DCA-P documents can also contain or reference some non-OCA data objects that are registered in the MO:DCA-P architecture. Such data objects can be carried in a generic MO:DCA-P object envelope that is called an object container. Some examples of data objects that can be carried in an object container are image objects in Encapsulated PostScript (EPS) format, Portable Document Format (PDF) single-page and multiple-page objects, TrueType and OpenType fonts, and color mapping tables (CMT).

MO:DCA-P data is composed into pages before it is sent to the printer and includes data placement and presentation information (such as which font to use), along with the data to be printed. The data to be printed consists of embedded OCA objects and object containers, and included resources. An embedded object must be physically embedded every time that it is used, and you need a separate object for each size and rotation you use. A resource is included in the MO:DCA-P data stream through a reference to the object, so it is not physically embedded in the data stream. You can save BCOCA, GOCA, IOCA, and PTOCA objects as resources. For information about BCOCA, GOCA, IOCA, and PTOCA resources, see Chapter 3, “Using resources,” on page 13.

To generate MO:DCA-P output from an application program, you can use any of these methods:

- Use a text-formatting program that generates page data, such as Document Composition Facility (DCF).
- Use AFP Toolbox (Program Number 5655-A25) to generate MO:DCA-P data in your application program. You can use this licensed program to generate MO:DCA-P output from an application without having detailed knowledge of the MO:DCA-P data stream and syntax.
- Use one of many products from IBM Business Partners to generate MO:DCA-P output. With the provided design tools that are both flexible and easy to use, create personalized, customer-oriented documents.

For more information, see *AFP Application Programming Interface: Programming Guide and Reference*, *Mixed Object Document Content Architecture Reference*, or *PSF for z/OS: Introduction*.

## AFP printing

---

One common way to use AFP data is to print it. [Figure 1 on page 3](#) shows the main components of AFP printing: resources, data streams, and the printer driver.

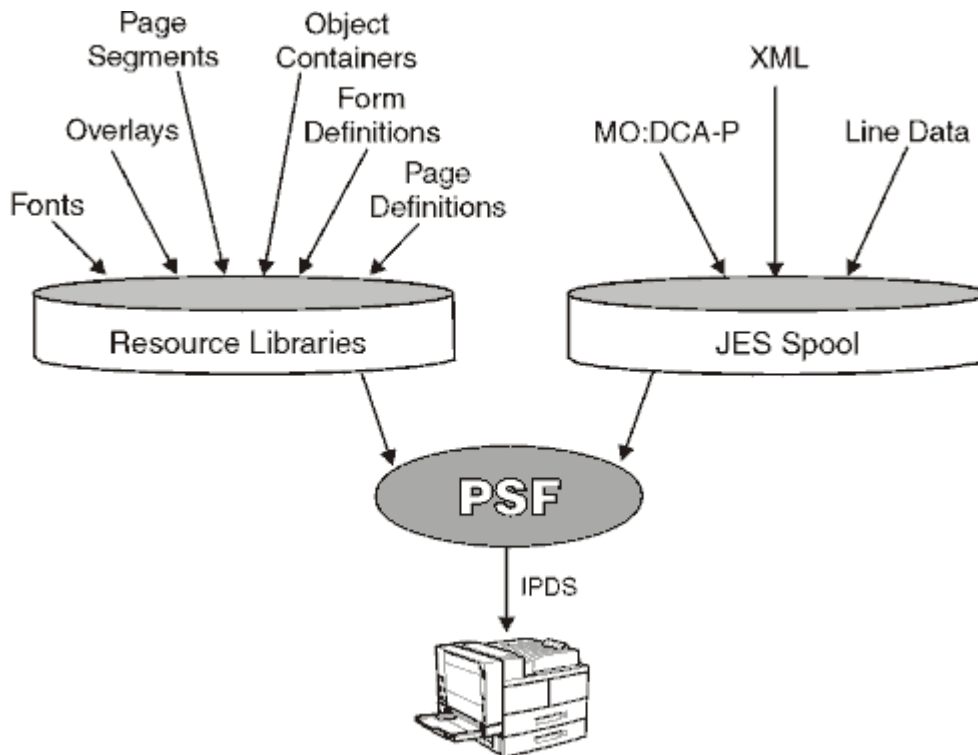


Figure 1. Components of AFP printing

1. Different types of resources are stored in resource libraries and can be retrieved by PSF for printing. A *resource* is a collection of printing instructions and sometimes data to be printed. It is bounded by delimiters that identify its type, such as graphics, image, or text.
2. Data streams can contain line data, MO:DCA-P data, and XML data. A *data stream* is data and control information that is transmitted through a data channel. The data streams are stored on the system spool, which is a storage area from which users can queue jobs to a device, such as a printer. The system spool for z/OS is Job Entry Subsystem (JES).
3. The data streams and resources are processed by the *printer driver*, which is a program that passes commands and resources with a data stream from the system spool to tell the printer how to print the data. The printer driver for z/OS is PSF. For more information about PSF, see [Chapter 2, “Introducing Print Services Facility,”](#) on page 7.

## Printing line data on AFP printers

All AFP printers are *page printers*. Page printers differ from line printers in that a page printer receives an entire page of data before it prints any data, whereas a line printer receives one line of data at a time and prints each line as it is received.

PSF can process line data so that AFP page printers can print it; therefore, a job that is prepared for printing on a line printer can be printed on a page printer with little or no change to the application. You can enhance line data by using electronic forms (overlays), typographic fonts, and other advanced-printing functions. For more information about line data, see [“Line data”](#) on page 71. Also, see [“Printing on an AFP printer”](#) on page 111.

## Printing AFP data in different directions and character rotations

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

The *inline direction* is the direction in which successive characters are added to a line of text. The four inline 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 for each inline direction are 0°, 90°, 180°, and 270°, and are measured clockwise around each inline direction. For example, the text in this paragraph is printed *across* the page, and its rotation is 0°. Figure 2 on page 4 shows the 16 possible combinations of these inline directions and character rotations. For information about the combinations that are supported by the printer you are using, see the documentation that is provided with the printer.

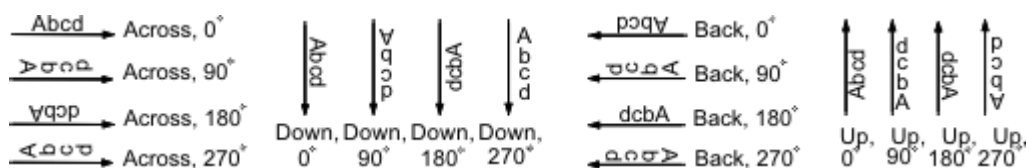


Figure 2. Direction-rotation combinations

PSF can process line data so that it can be printed in each inline direction with any character rotation. Figure 3 on page 4 shows four copies of a memo, each printed in a different inline direction with 0° character rotation. The memo was reformatted for each direction.

Pages are said to be printed in *portrait* page presentation when the shorter edges of the paper are the top and bottom of the page, and pages are said to be printed in *landscape* page presentation when the longer edges of the paper are the top and bottom of the page.

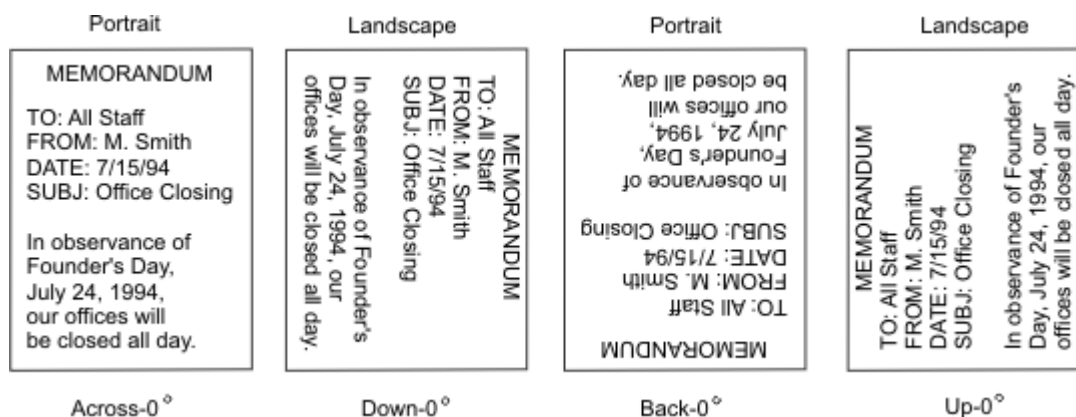


Figure 3. Copies of a memo that is printed in four inline directions

You can use a page definition to specify or select different inline directions and character rotations for different fields of data. Text-formatting programs can also produce different inline directions and character rotations for text. Figure 4 on page 5 shows the different combinations of inline directions and character rotations that are used on a sample page.

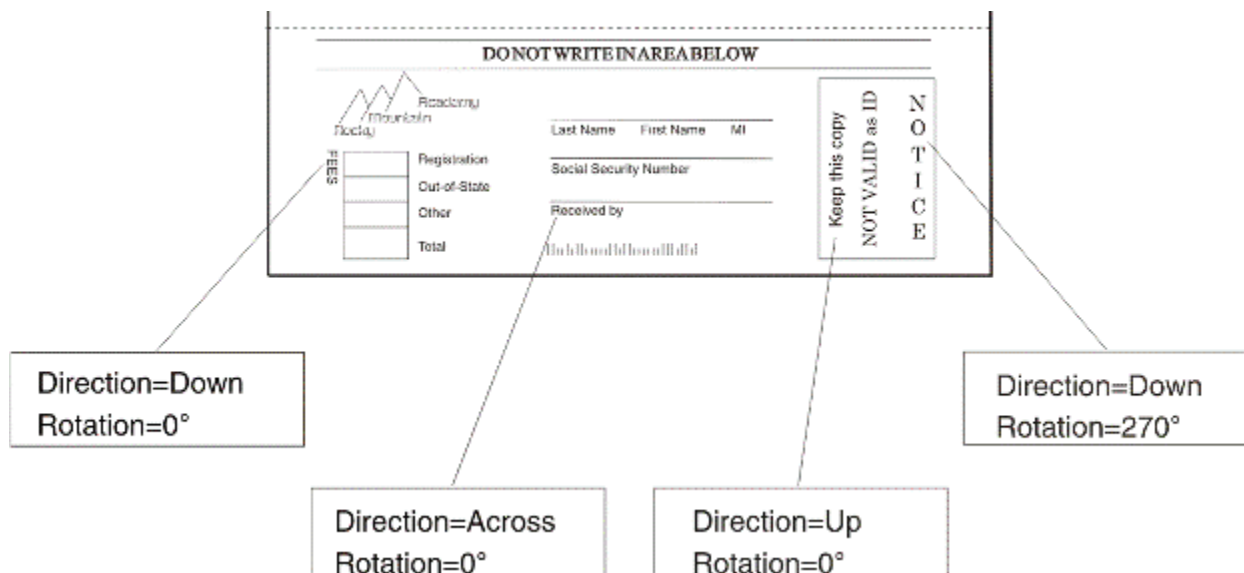


Figure 4. Direction-rotation combinations on a sample page

## All-points addressability

AFP page printers can print all addressable points, or *picture elements (pels)*, on a page. You can use *all-points addressable (APA)* printing to mix various type sizes, styles, images, and electronic forms on a single page. With all-points addressability, PSF can print different data at the same place on the page. Overlapping data in this manner does not produce bold print as it does with an impact printer. (To produce bold print with a page printer, you must use a bold font.) However, you can use these data-merging capabilities of PSF and page printers:

- Create composite characters by printing more than one character at the same location on the page. This capability is shown with the slash printed on top of the 0 in Example 1 in Figure 5 on page 5.
- Overlap characters so that parts of each character have a common area. This capability is shown with the asterisks (\*) printed on top of the characters 200.00 in Example 2 in Figure 5 on page 5.
- Merge a graphics object with character data. This capability is shown with the mountain graphic printed over the name of the academy in Example 3 in Figure 5 on page 5.

Example 1  
Composite Characters

080FE

Example 2  
Overstruck Characters

Cost \$200.00  
Reduced to \$150.00

Example 3  
Logo and Name Merged

Rocky Mountain Academy

Figure 5. Printing two items in the same location on the page

## **Pel density and placement**

The pel density is a major factor in determining the quality of a particular printer's output. The page printers that are supported by PSF have various pel densities. To learn about the pel density<sup>2</sup> of your printer, see the documentation provided with the printer. PSF sends the printer the address of a pel, along with the data to be placed there. With all-points addressability, you send data to the page printer in any sequence and position it anywhere on the printable area of the page.

For some print jobs, PSF must select addresses from formatting specifications. For others, addresses are already contained in the print jobs when PSF receives them.

## **Measurement unit conversion**

PSF accepts text, images, graphics, and bar code objects with values in the range of 1–32,767 for a 10-inch unit base. Some printers support only 2,400 units for every 10 inches and some support 14,400 units for every 10 inches. When the input values are not supported by the printer, PSF converts the input units for unit base values to values supported by the printer. This conversion can affect processing performance. Other printers support the full range of units for unit base values. For those printers, PSF does not convert the input units.

---

<sup>2</sup> Pel density is also called print-head resolution, or resolution.



## Chapter 2. Introducing Print Services Facility

Print Services Facility (PSF) is a licensed printer driver program. [Table 2 on page 7](#) shows the operating systems that run AFP printers and the PSF products they support.

Table 2. IBM operating systems for AFP printers and corresponding PSF products	
Operating System	PSF Product
<b>z/OS</b>	PSF for z/OS
<b>VM</b>	PSF/VM
<b>VSE</b>	PSF/VSE
<b>IBM i</b>	PSF/400

PSF has similar capabilities in all environments, plus differences unique to the operating system on which it is running. This information focuses on PSF on the z/OS system. On any operating system, PSF combines print data with resources to manage and control data transmitted to AFP printers. PSF accepts various data streams, transforms these data streams into the data stream that is required by each printer (by using processing and printing options that are specified by the user and the installation), and then transmits the data to the printer. In addition, PSF:

- Manages resources
- Verifies a part of the syntax of the input data stream (the printer verifies the rest of the syntax)
- Provides diagnostic aids, which provide information that can be used for problem determination
- Reports the status of a printer to the system operator
- Provides accounting information
- Provides error-recovery procedures for AFP printers, meaning that if a job does not print because of a printer problem, PSF retransmits the affected pages to the printer later

With PSF, you receive a starter set of resources. You can create more resources by using other programs.

### How PSF manages AFP printing

[Figure 1 on page 3](#) shows the basic components that are used to print data on AFP printers in an z/OS environment. The PSF printer-driver program processes the MO:DCA-P, line data, and XML data streams from the JES spool, combines the data streams with resources needed to print the data, converts the data into Intelligent Printer Data Stream (IPDS), and sends the result to the printer. Because MO:DCA-P and IPDS are part of the same architecture, this process is efficient for applications that produce MO:DCA-P.

### Components of PSF printing

The two main components of PSF printing are data streams and resources. PSF communicates with the printer to manage and control the data that is transmitted to the printers.

#### Data streams

The data streams placed on the JES spool are:

##### Line data

Application output that is prepared for printing but is not composed into pages. To print line data on page printers, a page definition is required to provide the data placement and presentation information. For information about printing line data, see [“Line data” on page 71](#).

**XML data**

Data that is identified as using Extensible Markup Language (XML) standards from the World Wide Web Consortium. XML does not describe data placement or presentation information. For printing on page printers, a page definition is required to provide the data placement and presentation information. The XML data that is processed by PSF can be encoded in EBCDIC, ASCII, UTF-8 or UTF-16. For more information about printing XML data, see [“XML data” on page 76](#).

**MO:DCA-P**

Data that is already composed into pages, including data placement and presentation information (such as which font to use). PSF supports MO:DCA Presentation Interchange Set (IS) data streams, including:

**MO:DCA AFP/Archive (AFP/A)**

MO:DCA AFP/A is an AFP document architecture interchange set that is used for long-term preservation and retrieval. This subset ensures page independence and eliminates images without clearly specified resolution, device default fonts, and external resources.

**MO:DCA Interchange Set 3 (IS/3)**

MO:DCA IS/3 is the first interchange set to achieve industry consensus through a rigorous open standards process. It improves existing functions and introduces new functions, such as Begin Print File (BPF) and End Print File (EPF) structured fields, and multiple image TIFF object support.

**MO:DCA AFP/A, IS/3**

MO:DCA AFP/A, IS/3 is an AFP document architecture interchange set that complies with the rules and restrictions of both the AFP/Archive and IS/3 interchange sets.

**MO:DCA Graphic Arts Function Set (GA)**

MO:DCA GA is an extension of MO:DCA IS/3 that adds PDF presentation object support.

For information about printing MO:DCA-P data, see [“MO:DCA-P data” on page 78](#).

**IPDS**

Data sent to the printer that contains both the data to be printed and the controls that define how the data is to be presented. For more information about IPDS, see [“Communication between PSF and the printer” on page 9](#).

**Mixed-mode**

A combination of line data and MO:DCA-P data. If MO:DCA-P data is already formatted into pages, it cannot be printed on the same logical page as line data. When PSF finds the beginning or the ending of a MO:DCA-P data page, it starts a new page.

**Resources**

PSF uses a combination of these resources to print AFP data:

**Fonts**

Graphic characters of a specific style that are used to present text. AFP fonts that PSF supports can be one of these types:

**FOCA**

Fonts that are defined by the Font Object Content Architecture (FOCA). A FOCA font is a paired character set and code page that can be used together to print a string of text characters.

**TrueType and OpenType**

Unicode-enabled fonts that are not defined by FOCA. TrueType and OpenType fonts consist of tables that identify the formatting information that is used to support Unicode encoding.

**Form definitions**

Information that defines the presentation of the logical page on the physical medium, such as where the page is placed on the medium and whether the data is printed on one or both sides of the paper.

**Object containers**

Object envelopes that contain certain types of non-OCA data objects.

## Overlays

Predefined data objects that can contain text, images, graphics, bar codes, object containers, and page segments and can be merged with application data for presentation. Overlays are often used as electronic forms.

## Page definitions

Information that is used to format line data and XML data into AFP pages.

## Page segments

Predefined data objects that can contain text, image, graphics, and bar code data objects that can be presented at any location on a page and are included during printing. Examples of items that can be page segments include logos, signatures, bar charts, and engineering drawings.

## Bar codes, graphics, images, and text

BCOCA, GOCA, IOCA, and PTOCA objects stored as resources.

For more information about resources, see [Chapter 3, “Using resources,” on page 13](#).

## Communication between PSF and the printer

PSF converts MO:DCA-P, line data, and XML data from the JES spool into IPDS. IPDS contains information about a printer, such as the characteristics of the printer, its resolution, what resources it has, whether it has sufficient memory, and whether it receives and prints a job. PSF communicates back and forth with the printer through IPDS to successfully manage and control the data that is transmitted to the printers.

## PSF as an output writer (deferred-printing mode)

In *deferred-printing mode*, PSF is the output writer that processes the spooled output from JES and sends a data stream to a page printer. All the printers that are supported by PSF can print in deferred mode, and PSF can process data for one or more page printers at a time.

In deferred-printing mode, you are not using PSF directly. Rather, JES sets up, starts, and controls each output writer and its system output devices. Output from an application program is spooled to JES, and printing is deferred until JES schedules PSF to print the spooled output data set. The components that are involved in generating formatted output on a system output printer are shown in [Figure 6 on page 9](#).

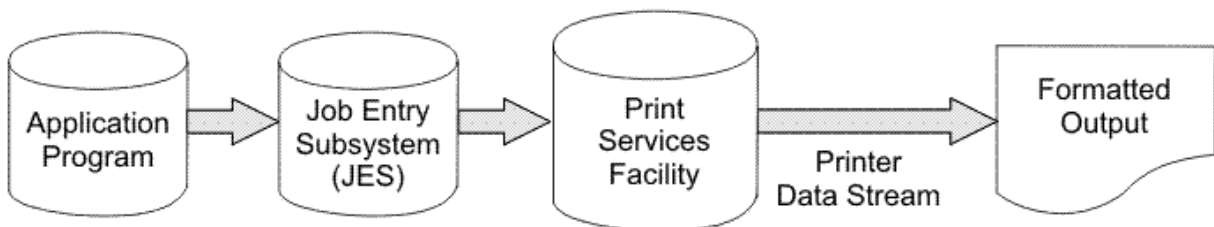


Figure 6. PSF as an output writer

Using information from Job Control Language (JCL) parameters and from the formatting specifications, PSF processes the print data set to generate the data stream from which the printer produces formatted output. For information about using JCL parameters to control the output, see [Chapter 6, “Using JCL for Advanced Function Presentation,” on page 81](#). For more information about deferred-printing mode, see [PSF for z/OS: Customization](#).

## PSF as an access method (direct-printing mode)

In *direct-printing mode*, PSF acts as an access method for the printer. PSF devotes the printer exclusively to the job, and the output is printed immediately. The components that are involved in direct-printing mode are shown in [Figure 7 on page 10](#).

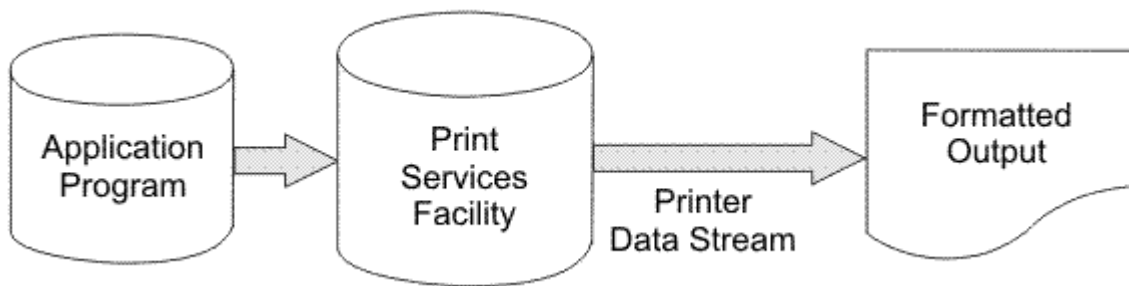


Figure 7. PSF in direct-printing mode

Only channel-attached printers can print in direct-printing mode. (Microfilm devices cannot be attached in direct-printing mode.)

PSF uses JCL parameters for direct-printing mode, but these parameters are different from the ones used in deferred-printing mode. Direct-printing mode does not support these deferred functions:

- User libraries
- System-assisted restart
- PSF repositioning
- Data set checkpointing
- Multiple data set processing
- JES operator commands to control the printer
- Job header and trailer pages
- Data set header pages
- Carrier-strip marking
- Mandatory page labeling
- System Management Facilities (SMF) type 6 processing
- Notification of print completion
- Restartable abends
- Redirection of message data sets
- PSF installation Exits 1–7
- Parameter specification in the Printer Inventory

For more information about direct-printing mode, see [PSF for z/OS: Customization](#).

## Formatting data with PSF

With PSF, you can format data in many ways. For example, by using page definitions to format the print lines into pages and form definitions to place the formatted pages on the media and to select media options, you can:

- Specify outline fonts to be scaled by the printer in any size or aspect ratio (see [“Outline fonts”](#) on page 20 and [“TrueType and OpenType fonts”](#) on page 22).
- Include page overlays and page segments anywhere on a page (see [“Page overlays”](#) on page 32 and [“Page segments”](#) on page 37).
- Include graphics and images on a page (see [“Graphics”](#) on page 42 and [“Image resources”](#) on page 43).
- Include electronic forms (see [“Merging data with an overlay”](#) on page 34).
- Specify data fields to be printed as bar codes (see [“Bar codes”](#) on page 39).
- Specify where the printer positions the page origin (see [“Page position”](#) on page 52).

- Specify duplex printing (see [“Duplex printing”](#) on page 54 and [“Specifying duplex printing”](#) on page 117).
- Select finishing options such as stapling or binding (see [“Finishing output”](#) on page 59 and [“Finishing your output”](#) on page 151).
- Select paper from multiple input bins, or route pages of the output to different output bins (see [“Paper source”](#) on page 53 and [“Specifying bins \(paper source\)”](#) on page 117).
- Change formatting on a page-by-page basis within a job (see [“Page definitions”](#) on page 63 and [“Changing formatting within a document”](#) on page 128).
- Print in any position on a page (see [“Defining page formats”](#) on page 63).
- Print in different orientations (see [“Printing AFP data in different directions and character rotations”](#) on page 3 and [“Print direction”](#) on page 64).
- Specify lines or fields to be printed in color (see [“Page format options for formatting fields”](#) on page 66).
- Select different fonts for lines or fields of data (see [“Fonts”](#) on page 67).
- Position print lines relative to other objects on a page (see [“AFP structured fields included in line data”](#) on page 74).
- Position multiple logical pages on a single sheet (see [“N\\_UP printing: printing multiple pages on a sheet”](#) on page 125).
- Specify data fields to be suppressed on some page copies of a document (see [“Suppressing print data”](#) on page 126).
- Include color resource objects for color fidelity management (see [“Specifying object container libraries in UNIX files”](#) on page 150).



---

## Chapter 3. Using resources

A *resource* is a collection of printing instructions and sometimes data to be printed. A resource is stored in a resource library and can be retrieved by PSF for printing.

This information describes where resources are stored and how PSF searches for resources, gives information about different types of resources, shows examples of resources that are used in a document, describes various ways to use resources, and lists some resources that are supplied by IBM. For information about managing resources, see [PSF for z/OS: Customization](#).

---

### Storing PSF resources

Resources can be stored in PSF libraries or as printer resources:

#### System libraries

System libraries, including security libraries, can be:

- A concatenation of partitioned data sets (PDS or PDSE) that contain members for one or more resources. Typically, a resource is built by an application or by an AFP licensed program and is stored in a library for other print jobs.
- A path or set of paths for system UNIX files (Hierarchical File System (HFS) or z/OS File System (zFS) files) that contain extended code pages or any resource installed in a resource access table (RAT), such as TrueType and OpenType font objects and data object resources.

#### Private user libraries

A private print-resource library can be:

- A PDS or PDSE that is owned by an individual user and is accessed only when an authorized job submitter specifies the data set name with the USERLIB OUTPUT JCL parameter.
- A path or set of paths for private UNIX files that contain extended code pages or any resource installed in a RAT, such as TrueType and OpenType font objects and data object resources. User path libraries are owned by individual users and are accessed only when an authorized job submitter specifies the path with the USERPATH OUTPUT JCL parameter.

#### Print data sets

Inline resources are stored in the print data set.

#### Printers

Some printers can store resources, such as fonts. Resources that are stored in a printer are considered *resident* or *printer-resident*. *Captured resources* are temporary printer-resident resources.

#### Distributed print function (DPF) libraries

Stored on a personal computer.

Printer-resident fonts and resources in DPF libraries are used only after a marked host resource is found. For more information, see [“Using printer-resident fonts” on page 24](#) and [“Using resources with the distributed print function \(DPF\) of PSF” on page 48](#).

Resources found in system libraries, user libraries, or inline are considered host resources. Host resources reside in the printer only during the print job. Printer-resident resources and DPF resources are stored in the printer or in an intermediate caching device.

[Figure 8 on page 14](#) shows where resources can be stored and the order in which PSF searches for them:

1. Resident in the printer or in DPF libraries
2. Among the inline resources for the data set
3. In any specified user libraries
4. In the system libraries

See “Searching for resources specified by a print job” on page 14 for the complete order that PSF searches for resources.

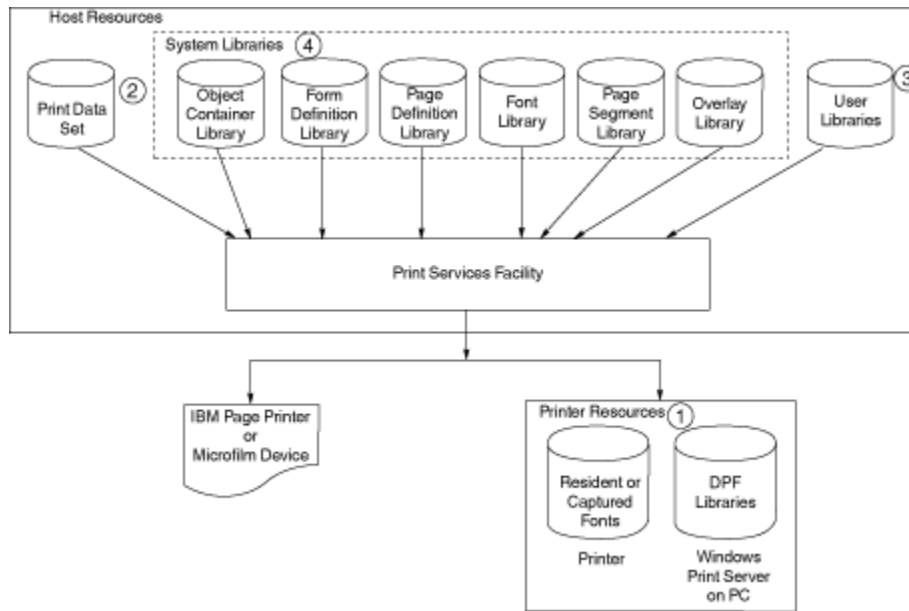


Figure 8. Resource storage

The resources that can be used depend on how the printer is set up, what licensed programs are installed, what libraries are assigned to each printer, and what data set is being processed.

When you select your printer by specifying JCL parameters, you are also selecting that printer's assigned system libraries. Check with your system-support group for what resources are available, what libraries the resources are in, and what libraries are assigned to each printer on your system.

## Searching for resources specified by a print job

The locations where resources are stored depend on the resource type. Resource types are stored in these locations:

### Inline in the print data set

All resource types

### System and user path libraries

- Extended code pages
- Object containers (also called data object resources)
- TrueType and OpenType fonts

### System and user PDS or PDSE libraries

- FOCA fonts
- Form definitions
- Object containers
- Overlays
- Page definitions
- Page segments

See Table 3 on page 16 for examples of resource types.

When a resource is referenced by a print data set, PSF only searches the locations where that resource type can be stored. PSF searches for the resource in this order:

1. If a resource can be activated by PSF, resources that are resident in the printer.



2. If PSF finds an Invoke Medium Map (IMM), internal copy groups immediately preceding IMM records in the print data set.
3. Inline resources in the print data set.
4. Resources in user path libraries.
5. Resources in user PDS or PDSE libraries in the order they are specified.
6. Resources in system path libraries.
7. Resources in system PDS or PDSE libraries in the order they are concatenated.

**Notes:**

1. PSF truncates the resource name to eight characters when searching PDS or PDSE libraries.
2. Only the security libraries (if PSFMPL is active) and the system libraries are searched for resources to be used on auxiliary data sets. For B1 security information, see *PSF for z/OS: Security Guide*.
3. An overlay called out in the Printer Inventory or the PRINTDEV is always loaded from the system libraries.
4. For soft IOCA resources, which are those not specified in the data stream with the MDR structured field, PSF searches the system PDS or PDSE libraries before searching the system path libraries.

## Soft and hard resources

---

Resources are considered *hard* or *soft* depending on how they are used.

**Soft resources**

Soft resources are resources that are included on a page but not mapped. Soft resources are integrated into the page data that is then sent to the printer.

**Hard resources**

Hard resources are sent to the printer in advance of the page as a separate resource object. They are mapped and only sent to the printer the first time they are referenced.

If a resource is used by more than a single page in the document, it might be beneficial for you to use the resource as a hard resource. Before it is used, a hard resource is referenced with a Map structured field in the active environment group of the page in which it is used.

## PSF system resource libraries

---

Resource libraries contain the resources needed for printing. Because some of the libraries (font, object container, and page segment) can contain multiple resource types, and because PSF does not enforce a prefix for most of the eight-character resource names, you can define a naming convention that identifies each type of resource in a library. IBM recommends that you use a two-character prefix naming convention for eight-character resource names

Table 3 on page 16 shows the PSF system resource libraries, the types of resources stored in each library, and the required and IBM-recommended prefixes.

Table 3. PSF system resource libraries, resource types, and resource prefixes

Library	Type of Resource	Description	Prefix	See
Font	All fonts			<a href="#">“Fonts” on page 19</a>
	FOCA font	Stored in PDS or PDSE font resource libraries:		<a href="#">“FOCA fonts” on page 20</a>
		240- and 300-pel character set	C0	
		3800 character set	C1–CG	
		Outline character set	CZ	
		Code page; extended code page	T1	
		240- and 300-pel coded font	X0 (required)	
		3800 coded font	X1–XG (required)	
		Outline coded font	XZ (required)	
		Stored in UNIX file path resource libraries:		
		Extended code page	T1	
	TrueType and OpenType font	Stored in UNIX file path resource libraries.	No prefix used	<a href="#">“TrueType and OpenType fonts” on page 22</a>
Form definition	Form definition	Stored in PDS or PDSE resource libraries.	F1 (required)	<a href="#">“Form definitions” on page 25</a>

Table 3. PSF system resource libraries, resource types, and resource prefixes (continued)

Library	Type of Resource	Description	Prefix	See
Object container	Color mapping table	Stored in PDS or PDSE resource libraries.	M1	“Object containers” on <a href="#">page 26</a>
	Encapsulated PostScript	Stored in PDS, PDSE, or UNIX file path resource libraries.	E1	
	Image (IOCA)	Hard and soft IOCA resources. IOCA resources that are identified in the data stream as hard resources with the Map Data Resource (MDR) structured field must be placed in the object container system library, a user library, or inline with the print data; soft IOCA resources can be placed in the object container or page segment library. Stored in PDS, PDSE, or UNIX file path resource libraries.	I1	
	IOCA tile	Stored in PDS, PDSE, or UNIX file path resource libraries.	IT	
	Microfilm setup	Stored in PDS or PDSE resource libraries.	H1	
	PDF resource	Stored in PDS, PDSE, or UNIX file path resource libraries.	PR	
	PDF single-page	Stored in PDS, PDSE, or UNIX file path resource libraries.	PP	
	Other data object resources	Stored in PDS, PDSE, or UNIX file path resource libraries. See the object type registry in <i>Mixed Object Document Content Architecture Reference</i> at AFP Consortium Publications ( <a href="http://afpcinc.org/publications">afpcinc.org/publications</a> ).	No prefix recommended	
Overlay	Overlay	Stored in PDS or PDSE resource libraries.	O1	“Overlays” on <a href="#">page 30</a>
Page definition	Page definition	Stored in PDS or PDSE resource libraries.	P1 (required)	“Page definitions” on <a href="#">page 36</a>

<i>Table 3. PSF system resource libraries, resource types, and resource prefixes (continued)</i>				
<b>Library</b>	<b>Type of Resource</b>	<b>Description</b>	<b>Prefix</b>	<b>See</b>
Page segment	Bar code (BCOCA)	Stored in PDS or PDSE resource libraries.	B1	<a href="#">“Bar codes” on page 39</a>
	Graphic (GOCA)	Stored in PDS or PDSE resource libraries.	G1	<a href="#">“Graphics” on page 42</a>
	IOCA	IOCA resources that are not identified with the MDR structured field (soft resources); stored in PDS or PDSE resource libraries.	I1	<a href="#">“Image resources” on page 43</a>
	Text (PTOCA)	PTOCA object with OEG; stored in PDS or PDSE resource libraries.	No prefix recommended	<a href="#">“Text” on page 46</a>
	Page segment	Stored in PDS or PDSE resource libraries.	S1	<a href="#">“Page segments” on page 37</a>

[Figure 9 on page 19](#) shows a page printed on an AFP printer by using PSF. This page shows examples of the kinds of data described in the following information.

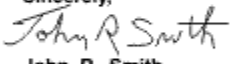
**Rocky Mountain Academy**  
**1234 Rocky Road**  
**Boulder, Colorado 80301**

February 25, 2003

Ms. Jane A. Doe  
 25 Park Avenue  
 White Rock, NY 10601

Dear Ms. Doe



Thank you for your interest in Rocky Mountain Academy. The admission application you requested is enclosed.

Sincerely,  
  
 John R. Smith  
 Director of Admissions

JRS/els  
 Enclosure

---

**DO NOT WRITE IN AREA BELOW**

 12345 Registration Out-of-State Other Total	<b>Doe</b>	<b>Jane</b>	<b>A.</b>
	Last Name	First Name	MI
	Social Security Number		
	Received by		
			

Keep this copy  
 NOT VALID as ID  
 NOTICE

Figure 9. Sample page printed on an AFP printer

## Fonts

A *font* is a collection of graphic characters that share the same type family, style, and weight. You can use a font for an entire data set or file, for an entire page, or for selected lines or fields of data on a page.

Page printers can print fonts with various point sizes,<sup>3</sup> styles, weights, and widths on a single line or on various lines on a page. Multiple fonts can be printed on a page. Before each page is printed, the fonts that are required for the page are sent to the printer (downloaded) if the printer does not already have them in its storage. The printer storage that is required for a font depends on the point size (for raster fonts), number of characters in the font, and whether it is double-byte or single-byte. Several publications that describe fonts are available in the [“Bibliography”](#) on page 277.

## Overview of AFP fonts

IBM AFP printers can use FOCA raster and outline fonts, which are single-byte or double-byte, or TrueType and OpenType outline fonts.

<sup>3</sup> A point size is a standard typographical measurement of the height of a font. One point is about 1/72 inch.

## FOCA fonts

Raster and outline fonts that are defined by the Font Object Content Architecture (FOCA) are called *FOCA fonts*. A FOCA font is a paired character set and code page that can be used together to print a string of text characters. FOCA fonts are stored in partitioned data sets (PDS or PDSE) in font resource libraries.

The different types of FOCA fonts that PSF supports are:

### Raster fonts

A raster font is a font that is created by a series of pels arranged to form an image. Raster fonts are created in a specific point size, so if you want to use different sizes of the same raster font, you must store multiple versions of the same font. Raster fonts can have 240-pel or 300-pel formats:

#### 240-pel fonts

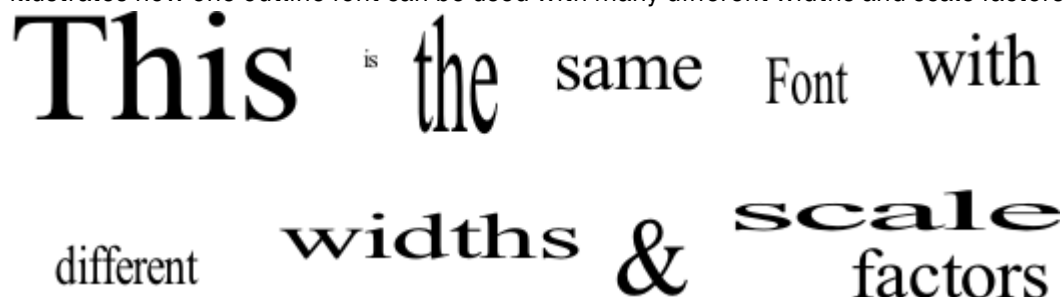
240-pel raster fonts can be bounded-box or unbounded-box. All IBM AFP printers except the 3800-3 use bounded-box fonts.

#### 300-pel fonts

Many IBM AFP printers print documents at 300-pel resolution. 300-pel fonts are provided in the AFP Font Collection, Program Number 5648-B33, or in the z/OS Font Collection, a base feature of z/OS, Program Number 5650-ZOS. Your system programmer can also convert any single-byte 240-pel font in the bounded-box format to a 300-pel font by using the font-conversion program distributed with PSF. This program, APSRCF30, is described in [\*PSF for z/OS: Customization\*](#).

### Outline fonts

PSF supports single- and double-byte outline font technology, in which the character shapes are represented by mathematical expressions. Because the font shape is defined without regard to size, outline fonts are scalable; therefore, you need to store only one version of an outline font, which increases your system storage space and enhances printing performance. [Figure 10 on page 20](#) illustrates how one outline font can be used with many different widths and scale factors.



This is the same Font with  
different widths & scale factors

Figure 10. Example of text that uses outline fonts

When you use an outline font you must specify a font size and a horizontal scale factor, which can be specified either in the MO:DCA-P data stream or in a coded font.

### Single-byte and double-byte fonts

Some page printers can print with both single-byte and double-byte fonts. A single-byte font can access up to 256 graphic characters from a character set. A double-byte font is used for printing languages whose base alphabets contain more than 256 graphic characters. To represent all the characters, more than 256 code points are needed. Therefore, two bytes are used: one byte to identify the section, and one byte to identify the code point in that section. An example of a writing system that requires double-byte fonts is kanji, in which the graphic characters are symbols in Japanese ideographic alphabets.

### DBCS simulation fonts

Double-byte character set (DBCS) simulation fonts are outline fonts that are positioned like the old DBCS raster fonts, which let you use outline fonts to print applications that use the old DBCS fonts without changing the application or changing its appearance. The DBCS simulation fonts are only used for an old application. Never mix the old DBCS raster fonts with the DBCS core raster or outline fonts because they are positioned differently.

FOCA fonts are stored in partitioned data sets (PDS or PDSE) in font resource libraries. A font resource library is composed of three library-member types:

- A *coded font member* associates a code page and a font character set as a pair. A single-byte coded font contains one code page and font character-set pair. A double-byte raster coded font, which requires two bytes to identify each graphic character, contains two or more code page and font character-set pairs; each pair is called a font section. A double-byte outline font contains one code page and font character-set pair. IBM requires a prefix of X0–XG and XZ for coded font resource objects.
- A *code page member* associates a code point and a graphic character identifier for each graphic character that is supported by the code page and specifies how code points that are not valid are to be processed. IBM recommends a prefix of T1 for code pages.

Extended code pages are code pages that can contain Unicode values, which a printer that supports extended code pages uses to print TrueType and OpenType fonts. Extended code pages can be stored in a partitioned data set (PDS or PDSE) in a font resource library or in a UNIX file in a font path library. When stored in UNIX files, extended code pages must have a .ECP file extension in uppercase format. For more information, see [“Using extended code pages” on page 141](#).

- A *font character set member* contains a graphic character identifier and a raster pattern or outline for each graphic character in the font or font section, and information about how the characters are to be printed. IBM recommends a prefix of C0–CG and CZ for font character sets.

PSF searches for FOCA fonts in any of these repositories:

#### **Resident in the printer**

If a font is resident or captured in a printer, PSF attempts to activate the font in the printer. See [“Using printer-resident fonts” on page 24](#) and [“Using captured fonts” on page 25](#) for more information.

#### **Inline in the print data set**

PSF looks for the specified font inline after first trying to activate the printer resident version of the font.

#### **User path library**

Extended code pages that are installed in a user path library can be accessed by PSF. PSF uses z/OS UNIX System Services to access the code pages in path libraries specified in the USERPATH parameter on the OUTPUT JCL statement. For more information, see [“USERPATH” on page 107](#).

#### **User library**

A font that is installed in a user library can be accessed by PSF. PSF accesses the fonts in PDS or PDSE libraries specified in the USERLIB parameter on the OUTPUT JCL statement. For more information, see [“USERLIB” on page 107](#).

PSF attempts to locate all fonts that are referenced in the user library after it looks for the printer resident version first and then the specified font inline. PSF looks for extended code pages that are referenced in the user library after it looks in the user path library.

#### **System font path library**

Extended code pages that are installed in a system font path library can be accessed by PSF. PSF uses z/OS UNIX System Services to access the code pages in path libraries specified in the FONTPATH parameter on the PRINTDEV statement of the PSF startup procedure. For more information, see [PSF for z/OS: Customization](#).

PSF attempts to locate referenced extended code pages in the system font path library after it first looks in the user path library and then the user PDS or PDSE library.

#### **System library**

A font that is installed in a system font library can be accessed by PSF. PSF accesses the fonts in PDS or PDSE libraries specified in the FONTDD, FONT240, or FONT300 parameters on the PRINTDEV statement of the PSF startup procedure. For more information about specifying the parameters on the PRINTDEV statement, see [PSF for z/OS: Customization](#).

PSF attempts to locate all fonts that are identified in the system font library after it looks for the printer resident version, the specified font inline, and in the user library. PSF looks for extended code pages that are referenced in the system library after it looks in the user path library, the user library, and the system font path library.

## TrueType and OpenType fonts

TrueType and OpenType fonts are outline fonts. They consist of tables that identify the formatting information that is used to support Unicode encoding. PSF has these limitations for supporting TrueType and OpenType fonts:

- PSF does not manage TrueType and OpenType fonts with installation Exit 7.
- PSF supports TrueType and OpenType fonts only on printers that support Unicode.

TrueType and OpenType fonts are stored in UNIX files (HFS or zFS files) in font path libraries. A font path library contains these TrueType and OpenType objects:

### Font

The basic TrueType and OpenType font element, which is a grouping of characters with the same typeface and style. A font object is stored as a single object or file.

### Collection

A group of TrueType and OpenType fonts collected together and stored as a single object or file. Font collections are created by the font vendor.

### Linked font

A group of TrueType and OpenType fonts that are associated by linking. One base font object is linked to other TrueType and OpenType font objects, which are known as linked fonts. When you are using WorldType fonts in Infoprint Fonts for Multiplatforms, the Font Installer for AFP Systems does font linking during font installation. When you are using the WorldType fonts in the z/OS Font Collection, the prebuilt resource access tables (RATs) contain linked fonts.

For PSF to use TrueType and OpenType fonts, the fonts must be in one of these repositories:

### Resident in the printer

If a TrueType and OpenType font is resident or captured in a printer, PSF attempts to activate the font from the printer. See [“Using printer-resident fonts” on page 24](#) and [“Using captured fonts” on page 25](#) for more information.

### Inline in the print data set

A TrueType and OpenType font can be contained inline in the print data set if it is wrapped in a MO:DCA-P object container construct. A wrapped TrueType and OpenType font is used to place the font inline in the MO:DCA-P data stream. PSF cannot interpret a TrueType and OpenType font data stream, unless it is contained in MO:DCA-P. PSF looks for the specified font inline after first trying to activate the printer resident version of the font.

### User path library

TrueType and OpenType fonts that are installed in a user path library can be accessed by PSF. These fonts do not contain any MO:DCA-P structured fields; therefore, they can be used by other print servers on multiple operating systems. PSF uses z/OS UNIX System Services to access the fonts in path libraries specified in the USERPATH parameter on the OUTPUT JCL statement. For more information, see [“USERPATH” on page 107](#).

PSF attempts to locate referenced TrueType and OpenType fonts in the user path library after it looks for the printer resident version first and then the specified font inline.

### System font path library

A TrueType and OpenType font that is installed in a system font path library can be accessed by PSF. These fonts do not contain any MO:DCA-P structured fields; therefore, they can be used by other print servers on multiple operating systems. PSF uses z/OS UNIX System Services to access the fonts in path libraries specified in the FONTPATH parameter on the PRINTDEV statement of the PSF startup procedure. For more information about specifying the FONTPATH parameter on the PRINTDEV statement, see [PSF for z/OS: Customization](#).

PSF attempts to locate referenced TrueType and OpenType fonts in the system font path library after it looks for the printer resident version, the specified font inline, and in the user path library.



## Obtaining and referencing fonts

The way that you obtain and reference fonts depends on whether they are FOCA fonts or TrueType and OpenType fonts.

### FOCA fonts

FOCA fonts are provided in the AFP Font Collection (Program Number 5648-B33), Infoprint Fonts for z/OS, outline fonts only (Program Number 5648-E76), and the z/OS Font Collection, a base feature of z/OS.

Double-byte 240 dpi raster fonts are available as a feature in the AFP Font Collection or in the z/OS Font Collection. Double-byte outline (scalable) fonts are available in the AFP Font Collection, in Infoprint Fonts for z/OS, or in the z/OS Font Collection. These fonts can be used on printers at any resolution, if the printer supports font scaling.

For MO:DCA-P data, you reference the fonts in the data stream with a Map Coded Font (MCF) structured field. Fonts that are used with line data or XML data are referenced in the page definition that is used to submit the job (see [“Fonts” on page 67](#)). Fonts that are used with traditional line data can also be referenced in the JCL used to submit the job. For information about selecting fonts in JCL, see [“CHARS” on page 86](#).

To reference FOCA fonts in a user library, see [“USERLIB” on page 107](#). To reference extended code pages in a user path library, see [“USERPATH” on page 107](#).

### TrueType and OpenType fonts

A basic set of TrueType and OpenType fonts is provided with WorldType Fonts for AFP Print Servers, an optional feature of Infoprint Fonts for Multiplatforms (Program Number 5648-E77), and the z/OS Font Collection, a base feature of z/OS V2R2 or later. To use TrueType and OpenType fonts in WorldType Fonts for AFP Print Servers, you need to use a resource installer program, such as the Font Installer for AFP Systems (an optional feature of Infoprint Fonts for Multiplatforms). The resource installer program installs and builds resource access tables (RATs) for TrueType and OpenType fonts that contain formal font names, paths, file names, and font attributes. The TrueType and OpenType fonts are mapped from their formal names to a path and file name that PSF can access. When you are using PSF with z/OS, a resource installer program is not necessary because the z/OS Font Collection contains prebuilt RATs for TrueType and OpenType fonts.

Keep these considerations in mind when you reference TrueType and OpenType fonts:

- For MO:DCA-P data, you reference TrueType and OpenType fonts in the print data set by using a Map Data Resources (MDR) structured field.
- To process complex text, you reference TrueType and OpenType fonts in the print data set. See [“Processing Unicode Complex Text” on page 141](#).
- For line data or XML data, you reference TrueType and OpenType fonts in a page definition that is referenced by the JCL for the application that generates the line data (see [“Fonts” on page 67](#)).
- When you reference a TrueType and OpenType font, you can optionally specify a FOCA code page that the printer uses to interpret the print data. Otherwise, you can specify the Unicode encoding of the data.
- When you reference a TrueType and OpenType font, you select the vertical font size, the horizontal scale factor, and the character rotation for the font.

To install TrueType and OpenType fonts in path libraries, see [“Using TrueType and OpenType fonts” on page 140](#).

## Using AFP fonts

AFP fonts can be mapped, used on printers with different resolutions, used to produce bar codes, used as printer-resident, and captured.

**Note:** When you are using fonts for a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

## Mapping FOCA fonts

Many outline fonts that are supplied in the AFP Font Collection, Infoprint Fonts for z/OS, and the z/OS Font Collection have equivalent raster fonts in selected sizes. If you specify outline fonts in your print jobs and are printing on a printer that supports only raster fonts, PSF tries to use equivalent raster fonts. You can use Type Transformer to create raster fonts from Type 1 outline fonts in the sizes you need. You can obtain Type Transformer at no extra charge from [IBM Type Transformer for Windows](http://www.ibm.com/support/docview.wss?uid=psd1P4000840) ([www.ibm.com/support/docview.wss?uid=psd1P4000840](http://www.ibm.com/support/docview.wss?uid=psd1P4000840)). Update PSF's user font mapping tables with information about these fonts.

For more information about using PSF's font mapping tables, see [PSF for z/OS: Customization](#) or contact your system programmer.

## Using FOCA fonts on printers with different resolutions

Some printers are manually configured to support a single raster resolution at a time, but can be switched between two different raster resolutions. Some printers can accept multiple raster resolutions and metric technology fonts at the same time; this function is sometimes called *automatic* or *auto resolution mode*. Ensure that the host font you want to use is supported by the printer. Some printers support only a single resolution. To determine which resolutions your printer supports, see the documentation provided with the printer.

Your system programmer can define multiple system font libraries (for example, 240-pel, 300-pel, and outline fonts) for switchable and auto mode printers. If the printer accepts only resources at a single resolution, PSF uses the printer's resolution to determine which system library to use. If the printer accepts multiple resolutions, you can use one of these methods to tell PSF which resolution system library to use:

- Your system programmer can specify a format resolution by using the Printer Inventory or the Resource Exit 7.
- Format the MO:DCA-P data by using an application that includes the Font Resolution and Metric Technology triplet on the MCF2 structured field. Two applications that currently support this triplet are Document Composition Facility (DCF) and Overlay Generation Language (OGL). See your application user's guide or to *Mixed Object Document Content Architecture Reference* for more information.
- Specify the RESFMT parameter on the OUTPUT statement. RESFMT tells PSF the resolution of the raster fonts that were used to format the data. See [“Specifying AFP parameters in the JCL”](#) on page 83 for more information about the RESFMT parameter.
- PSF can use the default system font library.

If the format resolution is specified in multiple places, PSF uses the first nonzero value that it finds in the methods previously listed.

## Using FOCA fonts to produce bar codes

Just as you can use FOCA fonts to print symbols for alphanumeric characters, you can use them to print symbols for BCOCA bar codes. Many different types of bar codes exist. The type of bar code is defined by its coding arrangement, or *symbolology*. Most bar codes are produced with BCOCA.

## Using printer-resident fonts

You can use printer-resident fonts only if they are supported on a printer. Some AFP printers, such as the network printers, InfoPrint 4000 and the InfoPrint 60, can use either host fonts or printer-resident fonts. Some printers, such as the 64xx, use only resident symbol sets, which are stored in the printer. Some printers, such as the 382x printer, 3835 printer, and 3900-001 printer, use only host fonts that are downloaded from PSF to the printer. Other printers do not support all the possible print-direction and character-rotation combinations for resident fonts and in some cases the characters in a printer-resident

font might not match the characters in the host version of the font. Your system programmer can tell you which printer-resident fonts are available on each applicable printer in your installation. For information about the fonts your printer supports, see the documentation provided with the printer.

If you are using a printer that supports only resident fonts, you are limited to printing with the font set that is stored on that printer. If you are using a printer that supports both resident and host fonts, you might want to use resident fonts instead of the equivalent host font. PSF can access the resident fonts in the printer instead of downloading them from the host font library, which saves transmission time. Your system programmer can identify or *mark* the host fonts by running the APSRMARK utility.

In order for a printer-resident font to be used, it must be the functional equivalent of the *marked* host version of the font. A host font that is marked as PUBLIC by using the APSRMARK utility is not sent to the printer if a matching printer-resident font is stored in the printer. If a font is marked as PRIVATE, PSF sends that font from the library to the printer even if a matching printer-resident font exists. If a font is not marked PUBLIC or PRIVATE, PSF assumes that the font is PRIVATE.

If you have AFP Font Collection or z/OS Font Collection, its fonts are already marked PUBLIC and you do not need to run APSRMARK to enable the printer-resident fonts. Also, the z/OS Font Collection or Font Installer for AFP Systems enables TrueType and OpenType fonts for capture.

If you update a font or create a new version of it by using an AFP utility, such as Type Transformer (see [IBM Type Transformer for Windows \(www.ibm.com/support/docview.wss?uid=psd1P4000840\)](http://www.ibm.com/support/docview.wss?uid=psd1P4000840)), the new version is not marked PUBLIC, and it is sent to the printer from host or user libraries. However, if you bypass IBM utilities to directly update the resource object code for a host-resident marked font, without changing the APSRMARK stamp that marks it as PUBLIC, PSF treats it as a PUBLIC font and does not download it to the printer; instead, PSF uses the earlier, printer-resident version. If you want to use the new version, you can mark the font PRIVATE or mark it PUBLIC with a new RRDATE and RRTIME. Some printers can capture these modified fonts and make them resident after they are correctly marked.

Do not mark a metric-only font character set PRIVATE, and do not mark code pages used with double-byte outline fonts PRIVATE. Because a metric-only font contains no font pattern, it must be marked PUBLIC, and print jobs must use the printer-resident version of the font. For more information about the APSRMARK utility, see [“Using APSRMARK to mark resources” on page 48](#).

## Using captured fonts

Some printers, such as the InfoPrint 60, InfoPrint 3000, and InfoPrint 4000, can capture downloaded fonts. Captured fonts become temporary printer-resident fonts, which improves performance for future jobs that use the same fonts. Printers can capture both raster fonts and outline fonts. A font that is marked PUBLIC, contains the appropriate date and time stamp, and resides in a system library is eligible for capture. Captured fonts remain resident in the printer if the storage is sufficient.

Do not mark sensitive fonts, such as signatures and MICR fonts, eligible to be captured because it is possible that an unauthorized person might access the captured font, even from another system. Improper use of font capturing might cause unpredictable results; therefore, only system administrators should handle the font capture feature. For more information about font capture, see the documentation provided with the printer.

### Notes:

1. FOCA fonts are enabled for printer-resident activation or capture by using the APSRMARK utility.
2. TrueType and OpenType fonts are enabled for printer-resident activation or capture by using the Font Installer for AFP Systems or z/OS Font Collection.

## Form definitions

---

A *form definition* is the resource that specifies the physical attributes of the printed output. The word *form* refers to a sheet of paper or any other print medium.

You must specify a form definition for each data set you want to print. You can specify a form definition by name, or you can use the default form definition that your installation sets up.

A form definition contains printing controls that specify these options, within the limitations of each printer:

- Page origin, which is the upper-left boundary for printing. When the printer is duplexing, page origin can be different for the front and back of the page.
- Sheets on which medium overlays are to be printed.
- Sheets on which medium preprinted form overlays are to be printed.
- N\_UP pages on which page preprinted form overlays are to be printed.
- Sheets on which a forms flash is to be printed.
- Printing with only overlays or forms flash and no variable data. This option is the constant-forms function.
- Number of copies of each page to be printed.
- Paper source (input bin of the printer) for printers with more than one paper source.
- Output bin for printers with more than one output stacker.
- Simplex printing (on one side of a sheet) or duplex printing (on both sides of a sheet).
- Data fields that are to be suppressed (not printed).
- Printed copy groups to be stacked offset from each other.
- Page presentation in either portrait or landscape position.
- Print-quality level for printers that support different levels.
- Horizontal adjustment in pels.
- N\_UP printing of multiple logical pages on a sheet. These pages can be MO:DCA-P pages (fully composed pages that contain data and the structured fields that control how the data is presented), line data, or XML data.
- Type of font fidelity. You can ensure the font that is used to format the data and the font that is used to print the data have the same resolution.
- Type of finishing; for example, corner stapling.

A form definition is required for every print job that you send to an AFP printer. PSF needs the form definition to position the logical page on the physical form. The form definition specifies the origin of the logical page as an offset from the origin of the physical form, or medium. The logical page is an area defined by the page definition for line data and XML data, or by structured fields for MO:DCA-P data.

For more information about medium origin and logical page origin, see [“Page position” on page 52](#).

**Note:** When you are creating form definitions for a microfilm device, see [Appendix E, “Microfilm device considerations,” on page 223](#).

## Using form definitions

IBM requires a prefix of F1 for form definitions.

- For more information about the function of a form definition in printing, see [“Form definitions” on page 52](#).
- For information about using a form definition, see [“Specifying a form definition” on page 112](#).

## Object containers

An *object container* is a MO:DCA structure that carries object data, which might or might not be defined by a presentation architecture. The required container structure depends on where the object is stored, how it is included in the data set, or both. If the object is stored in a resource library, the object can be included in its unaltered, original form; otherwise, the object must be wrapped with the appropriate MO:DCA container structured fields.

Object container resources are stored in object container libraries in one of these locations:

- Partitioned data sets (PDS or PDSE)

PSF accesses object container resources in PDS or PDSE libraries specified in the OBJCOND parameter on the PRINTDEV statement of the PSF startup procedure or the USERLIB parameter on the OUTPUT JCL statement. For more information, see [“USERLIB” on page 107](#).

- UNIX files (HFS or zFS files)

PSF uses z/OS UNIX System Services to access data object resources in path libraries specified in the OBJCPATH parameter on the PRINTDEV statement of the PSF startup procedure or the USERPATH parameter on the OUTPUT JCL statement. For more information, see [“USERPATH” on page 107](#).

For more information about specifying the parameters on the PRINTDEV statement, see [PSF for z/OS: Customization](#). For information about the order that PSF uses to search for object container resources, see [“Searching for resources specified by a print job” on page 14](#).

## Object containers as data object resources

Object container and image (IOCA) resources that PSF supports as hard or soft resources are called data object resources. Some of the object container resources that PSF supports as data object resources are:

- Color management resource
- Encapsulated PostScript (EPS)
- IOCA tile
- PDF resource
- PDF single-page
- PDF multiple-page
- Resident color profile
- Other object containers, such as GIF, JPEG, PCL, PCX, PNG, AFPC SVG subset, and TIFF, including TIFF multiple-image

See *Mixed Object Document Content Architecture Reference* for a complete list of data object resources. See Appendix F, [“Color and grayscale printing,” on page 225](#) for information about data objects used in color printing.

Hard data object resources are identified in the print data set by using the Map Data Resource (MDR) structured field and must be in a PSF system object container library, in a user library specified by the USERLIB or USERPATH parameter, or coded inline as part of the print data set. A soft data object resource is included by a page definition or by an IOB structured field in the input data set.

### Notes:

1. If an object container resource is inline as part of the print data set, the resource must be wrapped with MO:DCA structured fields.
2. If an object container resource is placed in a system or user library the resource might or might not be wrapped in MO:DCA structured fields:
  - For an object container resource wrapped in MO:DCA structured fields, the library is defined with carriage controls.
  - For an object container resource not contained in an object container, the library is defined without carriage controls. The library can be concatenated with libraries that do have carriage controls.

See [Table 3 on page 16](#) for the prefixes that IBM recommends for naming some data object resources.

## Common object containers supported by PSF

Some of the common object containers that PSF can send to the printer are:

- Color management resources
- Color mapping table resources

- Encapsulated PostScript resources
- IOCA tile resource objects
- Microfilm setup resources
- Multiple-image and multiple-page resources
- PDF single-page object resources
- PDF resource objects
- Resident color profile resource objects
- Scalable Vector Graphics (SVG)

For a complete list of object container types that might be supported by PSF and the printer, see registered object-type OIDs in the MO:DCA registry in *Mixed Object Document Content Architecture Reference*.

## Color management resource (CMR)

A color management resource is an object that provides color management in presentation environments.

Because CMR objects do not reside in partitioned data sets, a prefix is not needed for the name. For more information, see Appendix F, “Color and grayscale printing,” on page 225 or see *Color Management Object Content Architecture Reference*.

For print jobs that use CMRs, IBM recommends that you use outline fonts or TrueType and OpenType fonts. See [“Obtaining and referencing fonts”](#) on page 23.

## Color mapping table (CMT) resource

For some printers, such as the InfoPrint Color 100 and the InfoPrint Color 130 Plus, you can use a color mapping table (CMT) to map non-color fields to color; old color fields to new color fields; and new color fields to different new color fields. Thus, with a CMT, you can specify new color fields in existing applications and documents without changing the documents or applications. You can also use various color mappings with a single document to print the document, specifying color in various ways without changing the original document. CMT resources are stored in the object container library. IBM recommends a prefix of M1 for color mapping tables.

For more information, see [“COLORMAP”](#) on page 88 and [Chapter 8, “Using color mapping tables,”](#) on page 155.

## Encapsulated PostScript (EPS) resource

An EPS resource, which can contain any combination of text, graphics, and images, can be included along with MO:DCA-P data and printed on an IPDS printer that supports this object.

An EPS resource can be a soft or hard resource and is stored in the object container library. If the EPS resource is identified in the print data set by using a Map Data Resource (MDR) structured field, it is considered a hard resource. If the EPS resource is not identified by using the MDR structured field, but is included through an Include Object (IOB) structured field, it is considered a soft resource.

EPS data does not need to be wrapped in MO:DCA-P structured fields when it is stored as a resource. If EPS data is to be embedded in the data stream, it must be wrapped in MO:DCA-P structured fields.

Because an EPS resource is a type of object container resource, and PSF does not enforce a prefix for the eight-character name of an object container resource, your site can define a naming convention to avoid conflicts with other object container resources. IBM recommends a prefix of E1 for EPS objects. For more information, see [“Object containers as data object resources”](#) on page 27.

## IOCA tile resource object

An IOCA tile resource object is an IOCA FS45 tile resource. An IOCA tile resource object must be identified in the print data set by using an MDR structured field. An IOCA tile resource object does not



need to be wrapped in MO:DCA-P structured fields when it is stored as a resource. If this object is to be embedded in the data stream, it must be wrapped in MO:DCA-P structured fields.

Because an IOCA tile resource object is a type of object container resource, and PSF does not enforce a prefix for the eight-character name of object container resources, your site can define a naming convention to avoid conflicts with other object container resources. IBM recommends a prefix of IT for IOCA tile resource objects. For more information, see *Image Object Content Architecture Reference*.

## Microfilm setup resource

To print to a microfilm device, you need to specify a microfilm setup resource for your document. The microfilm setup resource is similar in concept to a form definition. The structure of the object container data field in the microfilm setup resource is defined by the manufacturer of the microfilm device. You can use microfilm setup resources to specify the processing commands for various microfilm device functions, such as titles, format, and extraction masks. Microfilm setup resources are stored in the object container library.

For information about how to use the utility that generates and packages the data in the microfilm setup resource, see [PSF for z/OS: Customization](#). The utility, provided with the microfilm device, places the setup resource information in an AFP object container, which is associated with the AFP print job by using JCL keywords. The resulting microfilm resource object container can be placed in a system library, in a user library, or inline in the data stream. PSF sends the object container setup resource with your data set to the microfilm device. If you do not specify a microfilm setup resource for your document, PSF uses the host default microfilm setup resource that your system programmer created, and your output might not appear as you expected.

For more information about the microfilm setup resource, see the publications provided with the microfilm device.

Because a microfilm setup resource is one type of object container resource, and PSF does not enforce a prefix for the eight-character name of the microfilm setup resource, your site can define a naming convention to avoid conflicts with other object container resources. IBM recommends a prefix of H1 for microfilm setup resources.

For more information about sending print jobs to a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

## Multiple-image and multiple-page resources

PDF multiple-page object resources, which can contain any combination of text, graphics, and images with PDF operators, and TIFF multiple-image object resources can be included along with MO:DCA-P data and printed on an IPDS printer that supports the objects. PDF multiple-page and TIFF multiple-image object resources can be transferred to the printer as one AFP object container, which optimizes file size, reduces processing time, and lets you individually reference and print one or more pages or images.

PDF multiple-page and TIFF multiple-image object resources can be stored inline with the print data, in a user library, or in the system object container library. When a multiple-page or multiple-image object is stored as a resource, it does not need to be wrapped in MO:DCA-P structured fields, but the page or image you want to reference must be included with an Include Object (IOB) structured field and the offset of the page or image must be specified. If the object offset is not specified, the first page or image within the object container is selected. When the object container is embedded in the data stream, it must be wrapped in MO:DCA-P structured fields.

## PDF single-page object resource

A PDF single-page object resource, which can contain any combination of text, graphics, and images with PDF operators, can be included along with MO:DCA-P data and printed on an IPDS printer that supports this object. A PDF single-page object resource can be a soft or hard resource that is stored inline with the print data, in a user library, or in the system object container library.

If the PDF single-page object resource is identified in the print data set by using an MDR structured field, it is considered a hard resource. If the PDF single-page object resource is not identified by using the MDR

structured field, but is included through an Include Object (IOB) structured field, it is considered a soft resource.

A PDF single-page object does not need to be wrapped in MO:DCA-P structured fields when it is stored as a resource. If this object is to be embedded in the data stream, it must be wrapped in MO:DCA-P structured fields.

Because a PDF single-page object resource is a type of object container resource, and PSF does not enforce a prefix for the eight-character name of an object container resource, your site can define a naming convention to avoid conflicts with other object container resources. IBM recommends a prefix of PP for PDF single-page object resources. For more information, see [“Object containers as data object resources”](#) on page 27.

## PDF resource object

A PDF resource object can be referenced by a PDF single-page or multiple-page object. Examples of PDF resource objects are fonts, font descriptors, and raster images. PDF resource objects must be identified in the print data set by using an MDR structured field.

A PDF resource object does not need to be wrapped in MO:DCA-P structured fields when it is stored as a resource. If this object is to be embedded in the data stream, it must be wrapped in MO:DCA-P structured fields.

Because a PDF resource object is a type of object container resource, and PSF does not enforce a prefix for the eight-character name of object container resources, your site can define a naming convention to avoid conflicts with other object container resources. IBM recommends a prefix of PR for PDF resource objects.

## Resident color profile resource object

A resident color profile resource object is a device-resident resource object that defines how device-dependent colors in a data object are related to device-independent colors. For example, many data object resources contain colors that are specified in the CMYK color space but tuned to one of a number of offset press standards that are geography-based. Some examples of resident color profiles are CMYK SWOP (US) and CMYK Euroscale (Europe). A resident color profile resource object defines how to render the colors in the data object resource that references it.

See the printer documentation to determine whether the printer has resident color profile resource objects and which ones are available.

## Scalable vector graphics resource object

A scalable vector graphics (SVG) resource object is a presentation object consisting of an SVG file that defines an XML-based file format for two-dimensional text, image, and graphics. PSF supports the AFP Consortium (AFPC) subset of this object defined in Presentation Object Subsets for AFP, available from the [AFP Consortium \(www.afpcinc.org\)](http://www.afpcinc.org) web page.

## Overlays

An *overlay* is a collection of predefined data that can be printed on a page by itself or merged with other data on a page as the page is printed.

Because an overlay can be printed on a page at the same time as the print data set is printed, overlays can be used as electronic forms to replace preprinted forms. The overlay on the sample page, which is shown in [Figure 11](#) on page 31, contains the information printed on the bottom of every letter. The print data set contains data that fills out the overlay.



Figure 11. Overlay on the sample page

An overlay can contain many different elements. Some of these elements are:

- Bar codes
- Boxes with and without shading
- Color
- Fonts, including fonts that are not used in the print data set
- Graphics and images
- Grids, arcs, and polygons
- Object containers
- Page segments
- Rules with different weights and thicknesses
- Text that is printed in different inline directions and character rotations
- Vertical, horizontal, and diagonal rules

An overlay cannot include another overlay, although multiple overlays can be printed on the same page.

PSF supports *medium overlays* and *page overlays*. Medium and page overlays have an identical object structure; the same overlay can be used as a page overlay or as a medium overlay. However, there is a difference in how to include the overlay on a page and in how PSF positions the overlay on the page. Therefore, when you create an overlay, its specified size and position might need to be different based on how you are using it.

You can identify medium overlays and page overlays as *preprinted form overlays*. A preprinted form or colored paper can be simulated with a regular medium or page overlay; however, because a regular overlay is applied first, some data might be hidden or unwanted data might be printed (such as a white-colored box on a yellow background). Preprinted form overlays better simulate preprinted forms or colored paper by changing the way the variable and overlay data are merged on the page; the overlay data is included after all other data is applied, and then the data is merged. Designating an overlay as a preprinted form overlay causes the printer to treat the preprinted form overlay data as if it is already printed on the paper before any other data printed.

You can create both medium and page overlays by using an IBM product such as Overlay Generation Language (OGL)/370 or by using various programs from IBM Business Partners. Many of these programs have easy-to-use PC-based layout editors so that you can see the overlay as you create it. For more information about coding overlays, see *Overlay Generation Language/370 User's Guide and Reference*.

**Note:** When you are using overlays on a microfilm device, see [Appendix E, "Microfilm device considerations,"](#) on page 223.

## Medium overlays

PSF positions a medium overlay at the media origin. The medium origin is constant throughout a document, but is printer-dependant.<sup>4</sup> Thus, a medium overlay is positioned at the same place on each

<sup>4</sup> For information about the medium origin for your printer, see the documentation provided with the printer.

page on which it is printed. You can print medium overlay data in different locations on different pages in these ways:

- Create different versions of a medium overlay.
- Use different medium maps to position the same medium overlay in different positions.

Figure 12 on page 32 shows the positioning of a medium overlay.

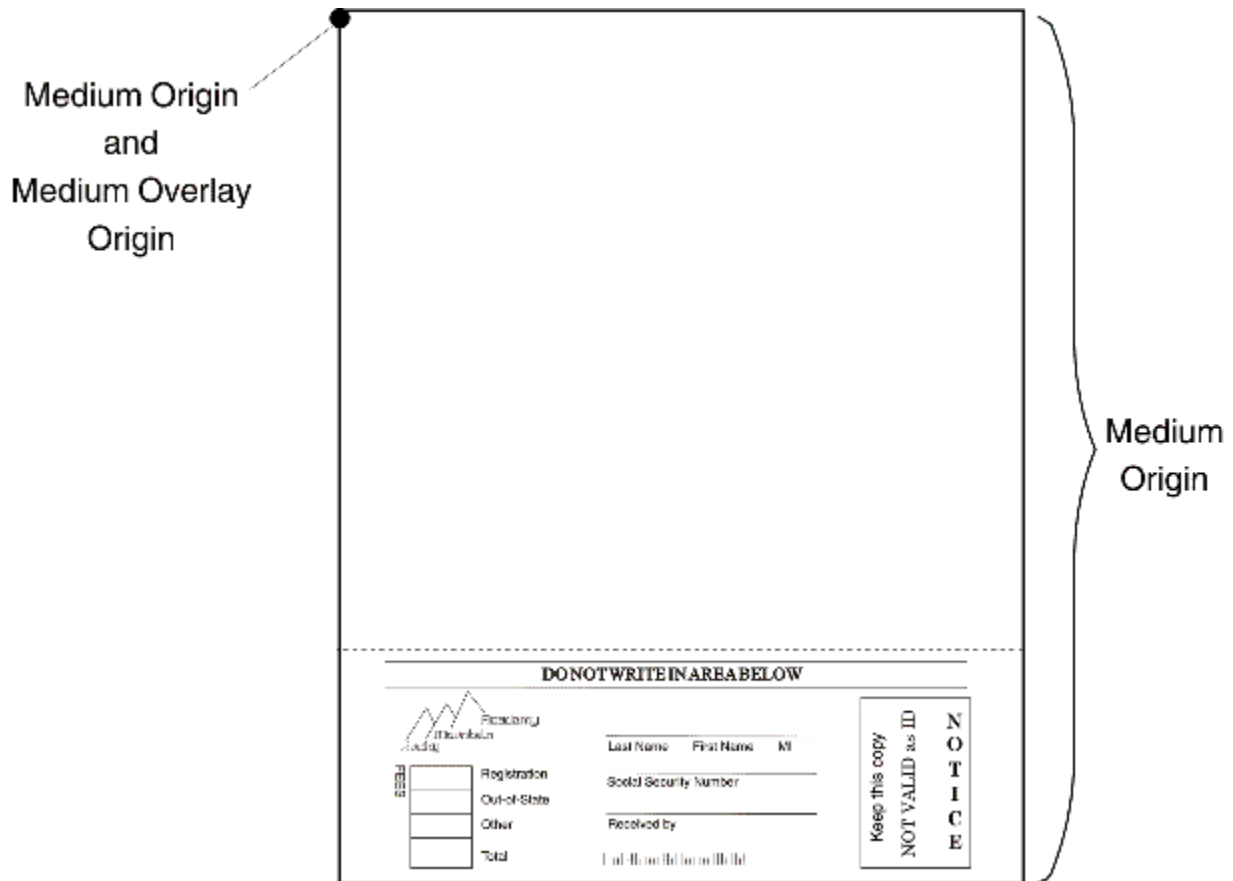


Figure 12. Positioning of a medium overlay

As Figure 12 on page 32 shows, the size of a medium overlay is generally the same as the page size. In this example, the positioning information in the overlay causes PSF to print the data in the overlay at the bottom of the page. You can control the size of a medium overlay; for example, the OGL OFFSET parameter increases the total size of the compiled overlay resource and offsets the overlay data within the expanded overlay boundaries.

## Page overlays

PSF can position a page overlay, like a page segment, at various locations on different pages. The position of a page overlay can be defined in the print data set relative to the logical page origin of each page on which the page overlay is to be printed. The logical page origin is defined in the form definition, or with the OFFSETXF and OFFSETYF, or the OFFSETXB and OFFSETYB keywords on the OUTPUT JCL statement.

For added flexibility in positioning a page overlay on a page, you can specify in the print data set that the page overlay is to be printed at the current print position on the page. Figure 13 on page 33 shows the positioning of a page overlay at specified horizontal (x-direction) and vertical (y-direction) distances from the logical page origin.

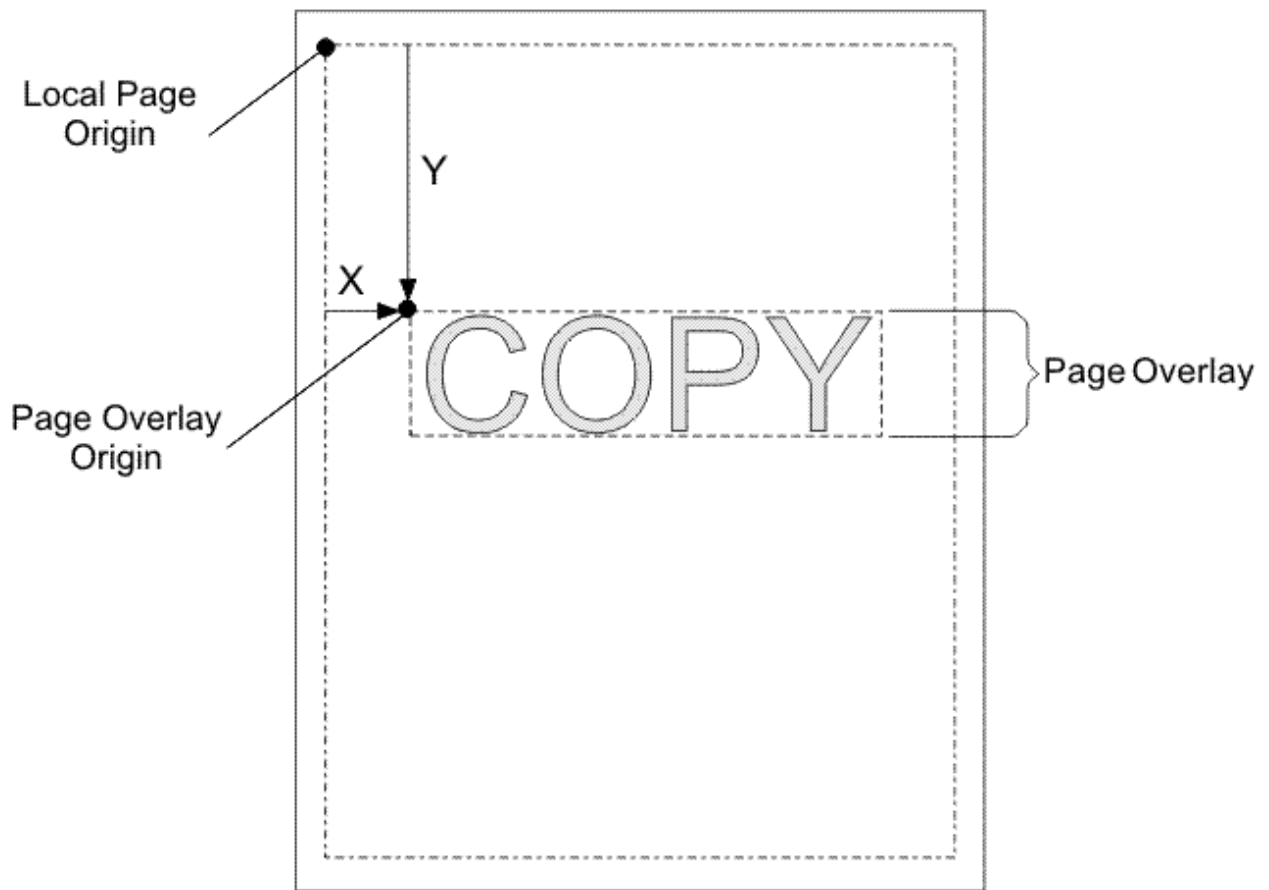


Figure 13. Positioning of a page overlay

The size of a page overlay can be the same as the page size; or, as [Figure 13 on page 33](#) shows, the size of a page overlay can be limited to the size of the overlay data. If you limit the size of the overlay to the size of the overlay data, you can position the overlay more easily in different locations on different pages.<sup>5</sup>

**Note:** Not all AFP printers support page overlays. If the printer does not support page overlays, PSF issues an error message and prints the page without the overlay.

## Using overlays

You can use medium overlays, page overlays, or both. The choice depends in part on the application. To print the same overlay in different positions on a page, you must use a page overlay. To print the same overlay in the same position on each page of a print data set, you can use either a medium overlay or a page overlay; however, for this kind of application, specifying a medium overlay might be easier. The two kinds of overlays are specified by different procedures. IBM recommends a prefix of 01 for all overlays.

You can use a form definition to identify one medium overlay or page overlay on a sheet as a preprinted form overlay.

### Medium overlays

To include a medium overlay on all sheets of a data set, you can specify the overlay name on the `OVERLAYF` or `OVERLAYB` keywords of the `OUTPUT JCL` statement. Or you can use the form definition to include a medium overlay on some or all sheets of a print data set. You do not need to modify the print data set to print a medium overlay on every sheet.

If you want to print a medium overlay without merging it with data from the print data set, use a utility such as IBM Page Printer Formatting Aid (PPFA) to specify the constant-forms function in the form definition. For example, in PPFA, use the `CONSTANT` subcommand. When the constant-forms function

<sup>5</sup> To position a page overlay by using OGL, use overlay `OFFSET` value of zero.

is specified, PSF prints the medium overlay on a page by itself. Thus you can, for example, print an overlay on the back side of each page in a print data set with no variable data on that side.

When you use a medium overlay, you can specify different overlays for different copies of a page. For example, you can print one copy of a page with an overlay, the second copy of the page with a different overlay, and the third copy with no overlay.

The number of medium overlays that you can use on a page depends on the complexity of the overlays, the amount of storage required for fonts and page segments in the overlays, and the amount of storage required for the page. You can specify a maximum of nine medium overlays in a subgroup—eight medium overlays and one preprinted form overlay.

### Page overlays

You can include a page overlay in line data or XML data by creating a page definition that calls in the overlay to print, relative to a print line (in traditional line data), a layout (in record format line data), or an xlayout (in XML data). For line data, you also can code an Include Page Overlay (IPO) structured field in the application to include the overlay on a particular page and to identify the print position of the overlay on that page. You must list the overlay name in the page definition that is used for printing the line data before you can call it by using an IPO structured field. You can use a utility such as PPFA to create a page definition that includes a page overlay or lists the overlay for inclusion by an IPO structured field. See [“AFP structured fields included in line data” on page 74](#) for more information about using structured fields.

To include a page overlay in a MO:DCA-P document, use the facilities that are provided by a document layout program, or specify the IPO structured field in the document. To use an IPO structured field, you must also list the overlay in a Map Page Overlay (MPO) structured field on the same page.

The number of page overlays that you can use on a single page depends on the complexity of the overlays, the amount of storage required for fonts and page segments in the overlay, and the amount of storage required for the page.

### Preprinted form overlays

Preprinted form overlays can be better than regular medium or page overlays for simulating preprinted forms or colored paper because the overlay data is included after all other data is applied, and then the data is merged. To identify a preprinted form overlay, create a form definition that identifies the overlay as a preprinted form overlay. A medium overlay is identified as a preprinted form overlay with a medium preprinted form overlay local ID keyword (X'D2') on the MMC structured field; a page overlay is identified as a preprinted form overlay with a Resource Object Include triplet (X'6C') on the PMC structured field that is specified with a PFO object type.

**Keep in mind:** Only one type of preprinted form overlay is allowed on each page on a sheet. For example, a page preprinted form overlay is ignored if a medium preprinted form overlay is identified. Also, if more than one of the same type of preprinted form overlay is identified, only one is used and the others are ignored.

For more information about preprinted form overlays, see *Mixed Object Document Content Architecture Reference*.

## Merging data with an overlay

In PSF, you can use an application to merge data with an electronic form, or *overlay*. An overlay contains constant information and can also contain blanks that can be filled in by the application. The print data set the application creates contains the variable data that is printed on the electronic form. PSF can print both the variable data and the form on a single sheet, eliminating the need to print forms before printing the variable data. PSF can also print an overlay on a blank page that contains no variable data. [Figure 14 on page 35](#) shows an electronic form that was used on the sample page. The application program supplied the name Jane A. Doe.

Figure 14. Electronic form (overlay) on the sample page

## Using overlays on printers with different resolutions

PSF supports multiple system overlay libraries, as follows:

- 240-pel library
- 300-pel library
- Default library

The respective system libraries would contain overlays that are designed to print at 240-pel resolution, 300-pel resolution, or at any resolution. If you know that your overlay prints best at a certain resolution, then you can store the overlay in the system library that is designed for it and specify this resolution to PSF.

You can use any of several methods to specify the format resolution:

- Your system programmer can specify a format resolution by using the Printer Inventory or the Resource Exit 7.
- Specify the RESFMT keyword on the OUTPUT statement. RESFMT indicates the resolution in which the overlays were formatted. See [“Specifying AFP parameters in the JCL”](#) on page 83 for more information.
- PSF can use the default system overlay library.

## Testing overlays

To make the testing of overlays easier, submit an overlay as print data by specifying the overlay name as the data set name in the job submission JCL. PSF treats the overlay as a page. Only one overlay at a time can be printed in this manner.

## Medium overlays and page overlays on the same page

PSF can print medium overlays and page overlays on the same page. [Figure 15 on page 36](#) shows the sample page with the medium overlay from [Figure 12 on page 32](#) and the page overlay from [Figure 13 on page 33](#) printed along with the variable data from the print data set.

**Rocky Mountain Academy**  
1234 Rocky Road  
Boulder, Colorado 80301

February 25, 2003

Ms. Jane A. Doe  
25 Park Avenue  
White Rock, NY 10601

Dear Ms. Doe:

Thank you for your interest in Rocky Mountain Academy. The admission application you requested is enclosed.

Sincerely,

*John R. Smith*  
John R. Smith  
Director of Admissions

JRS/eis  
Enclosure

---

**DONOT WRITE IN AREA BELOW**

	Registration	Doe	Jane	A.
	Out-of-State	Last Name	First Name	MI
	Other	Social Security Number		
	Total	Received by		
		0000000000000000		

Keep this copy  
NOT VALID as ID  
**NOTICE**

Figure 15. Medium overlays and page overlays on the same page

## Page overlay rotation in page definitions and form definitions

In PPFA, an orientation parameter is added to the PRINTLINE and LAYOUT parameters for page definitions and form definitions. You can create page definitions and form definitions that specify overlay orientation by using the PPFA OVROTATE keyword.

If you are writing your own MO:DCA-P objects, see the extension to the Resource Object Include (X'6C') triplet of the Page Modification Control (PMC) structured field in *Mixed Object Document Content Architecture Reference*, or the LND structured field in the *Advanced Function Presentation: Programming Guide and Line Data Reference*. You can use the PMC structured field or the LND structured field to specify the Resource Object Include triplet orientation parameter. You can rotate page overlays with this parameter.

If the printer does not support page overlay rotation, PSF ignores the Resource Object Include triplet parameter. PSF also ignores the Resource Object Include triplet orientation parameter if the triplet does not include an overlay. PSF does not generate messages for either of these conditions.

## Page definitions

A *page definition* is the resource that specifies how PSF formats line data or XML data into pages. PSF does not use a page definition for MO:DCA-P data because that data is composed into pages before PSF receives it. The page definition replaces the forms control buffer (FCB) used by line printers.

You can specify a page definition by name, or you can use the default defined by your installation.

A page definition contains formatting information that specifies:

- Page size (height and width)
- Print direction for the page of data

- Number of lines per inch
- Fonts to be used for printing the data
- Where data from each input record is to be printed
- Constant data to be printed
- Data fields that can be suppressed
- Data fields to be printed as a bar code
- Print position for carriage control characters or channel codes
- List of page segments used by this job
- List of page overlays used by this job
- List of object containers used by this job
- Conditional processing to change page formats, based on the data
- Color selection (for printers that support printing in multiple colors)
- Include and position page segments or overlays
- Rotation of included page overlays

## Using page definitions

IBM requires a prefix of P1 for page definitions.

- For more information about the function of a page definition in printing, see [“Page definitions” on page 63](#).
- For information about using a page definition, see [“Specifying a page definition” on page 121](#).
- When you are sending constant data to a microfilm device, see [Appendix E, “Microfilm device considerations,” on page 223](#).

## Page segments

---

A *page segment* is an object that can be merged with the variable data on a page that is being printed. It typically contains image data such as bar codes, signatures, logos, or graphics converted into image format. You can include a page segment in a print data set or in an overlay resource.

Page segments can contain character data that is formatted as MO:DCA-P data; however, limitations exist when you use page segments that contain text, and no IBM products create page segments that contain text.

You can create page segments by using any PC program in combination with the AFP WorkBench for Windows, or with the AFP Printer Driver for Windows. The AFP Printer Driver converts the PC application page into an AFP page segment, which can then be uploaded to z/OS.

The page segments on the sample page are shown in [Figure 16 on page 38](#).

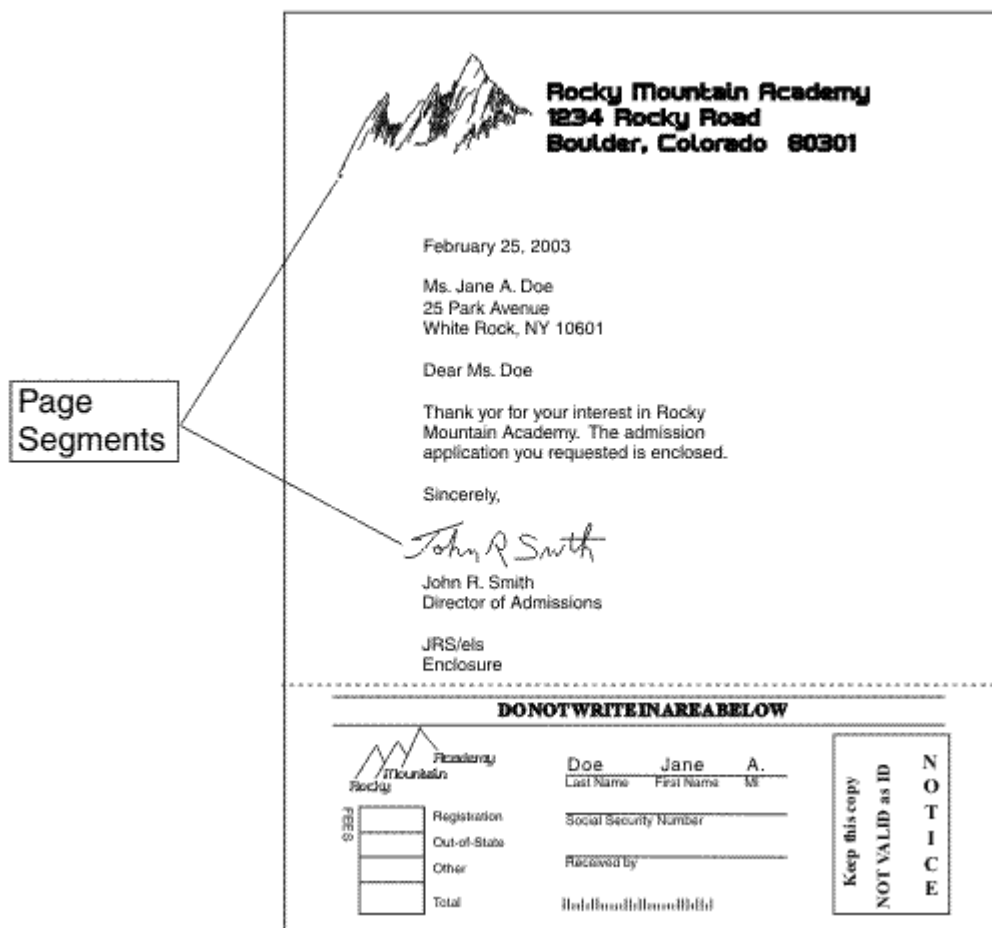


Figure 16. Page segments on the sample page

PSF can position a page segment at any location on a page. The position of a page segment is defined relative to the logical page origin, to the origin of an overlay, or to the page content that precedes it. (The logical page origin is defined in the form definition, or with the OFFSETXF and OFFSETYF keywords, or the OFFSETXB and OFFSETYB keywords on the OUTPUT statement.) For added flexibility in positioning a page segment, you can specify that PSF print the page segment at the current print position on the page.

**Note:** When you are using page segments on a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

## Using page segments

You can include a page segment in an overlay by using commands in a utility such as OGL. For line data and XML data, you can create a page definition to include a page segment positioned relative to a print line (in traditional line data), layout (in record format line data), or xlayout (in XML data). You can also include page segments in line data and in MO:DCA-P data by using MO:DCA-P structured fields. An Include Page Segment (IPS) structured field specifies the name and page position for a page segment. An Include Object (IOB) structured field can also specify size, rotation, and color for a page segment. IBM recommends a prefix of S1 for page segments.

To include a page segment in a print data set or in an overlay, you can use commands in an AFP resource utility (such as PPFA) or in a document layout application, to specify the name of the page segment and where to place it on the page.

Page segments included with an IOB structured field must be MO:DCA-P page segments. MO:DCA-P page segments consist of image (IOCA), graphic (GOCA), or bar code (BCOCA) object data. For more information about writing MO:DCA-P structured fields, see [“AFP structured fields included in line data”](#) on page 74.



When you are printing a page segment on more than one page of a print data set, you can improve performance by requesting that PSF retain the page segment in the printer while the file is printing. Page segments that are retained in printer storage are called *hard* page segments. To request that PSF retain the page segment in printer storage do this:

- For line data, specify the page segment in the page definition.
- For page data, identify the page segment as a *hard* page segment in a Map Page Segment (MPS) structured field in the print data set.

If a page segment is included in an overlay, PSF retains the page segment in printer storage if the overlay is retained.

Page segments included through an IOB structured field are always *soft* page segments, because the information in the object environment group must be changed for the override values specified in the IOB.

You can also include page segments in a print data set by coding structured fields in the print data set. *Advanced Function Presentation: Programming Guide and Line Data Reference* describes how to code the structured fields that are used to include page segments on a page.<sup>6</sup> For an example of how to use these structured fields, see [“Printing page segments” on page 130](#).

With IBM printer microcode for extended page segments, the maximum number of page segments is extended to a total of 32511 page segments per sheet. Extended page segments are not supported when the printer is connected with DPF or RPM3.

## Using multiple system page segment libraries

PSF can support multiple system page segment libraries. The system libraries can contain page segments that are designed to print at 240-pel resolution, 300-pel resolution, or at any resolution. If you know that your page segment prints best at a certain resolution, you can store the page segment in the system library that is designed for it and specify this resolution to PSF when you submit a print job.

You can use any of several methods to specify the image resolution:

- Your system programmer can specify a format resolution by using the Printer Inventory or the Resource Exit 7.
- Specify the RESFMT keyword on the OUTPUT statement. RESFMT indicates the resolution in which the page segments were formatted. See [“RESFMT” on page 103](#) for more information.
- PSF can use the default system page segment library, which can contain segments at 240 or 300 ppi, or at both resolutions.

## Testing page segments

To make the testing of page segments easier, submit a page segment as print data by specifying the page segment name as the data set name in the job submission JCL. PSF treats the page segment as a page. Only one page segment at a time can be printed in this manner.

**Note:** You can get unpredictable results when printing page segments that have text (PTOCA) included. The page segment does not contain enough information about the font. Therefore, you might get message APS818I posted and your text might be printed with an incorrect font.

For more information about page segments, see *Mixed Object Document Content Architecture Reference*.

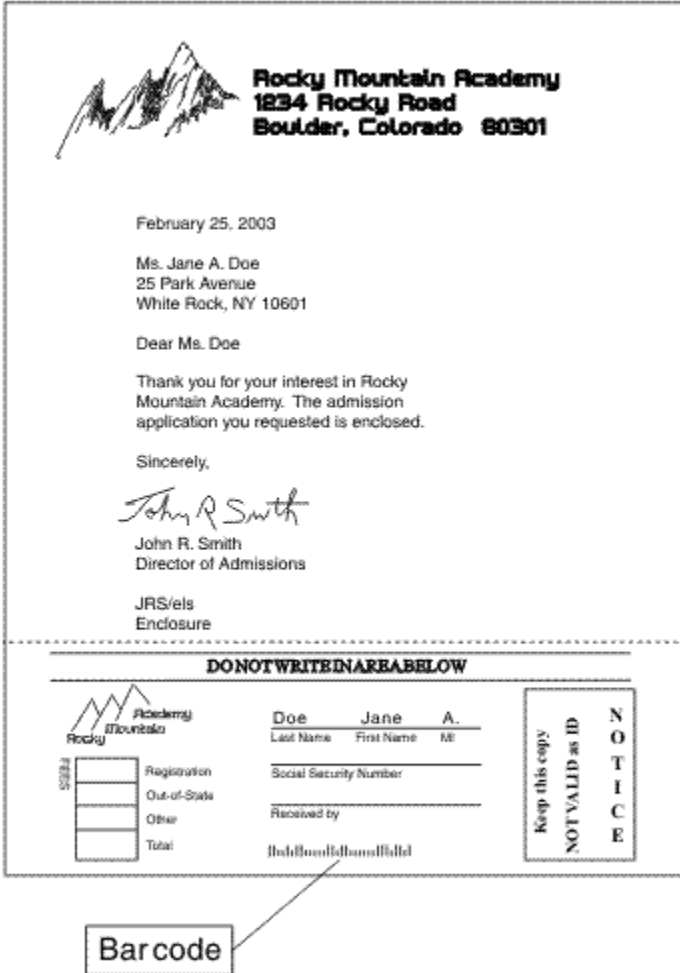
## Bar codes

A bar code resource (or BCOCA resource) is a Bar Code Object Content Architecture (BCOCA) object stored as a resource in the page segment library. Bar codes are soft resources.

---

<sup>6</sup> The Include Page Segment structured field is used to include a page segment on a particular page of a print data set or in an overlay. The Map Page Segment structured field is used in a page definition or a page print data set to name the page segment for resource management.

Bar codes represent characters by sets of parallel bars of differing thickness and separation, multiple rows of parallel bars, or a matrix of data cells that is typically composed of squares or circles. These codes can represent, for example, product numbers, part numbers, and manual numbers. Bar codes can be read optically by transverse scanning. [Figure 17 on page 40](#) shows the bar code printed on the sample page.



**Rocky Mountain Academy**  
1234 Rocky Road  
Boulder, Colorado 80301

February 25, 2003

Ms. Jane A. Doe  
25 Park Avenue  
White Rock, NY 10601

Dear Ms. Doe

Thank you for your interest in Rocky Mountain Academy. The admission application you requested is enclosed.

Sincerely,  
*John R. Smith*  
John R. Smith  
Director of Admissions

JRS/eis  
Enclosure

---

**DO NOT WRITE IN AREA BELOW**


	Registration	Doe Jane A.	<b>Keep this copy NOT VALID as ID NOTICE</b>
	Out-of-State	Last Name First Name MI	
	Other	Social Security Number	
	Total	Received by	
		Barcode	

Figure 17. Bar codes on the sample page

Many different kinds of bar code coding arrangements, or *symbolologies*, are developed for specific applications. PSF supports both linear symbolologies and two-dimensional symbolologies. Some of these codes are:

- 2 of 5 Codes: Industrial, Interleaved, Matrix
- Codabar
- Code 39 (Code 3 of 9)
- Code 93
- Data Matrix and MaxiCode
- EAN 8 and EAN 13
- MSI/Plessey
- PDF417
- Postnet
- QR Code
- UPC A and UPC E

**Note:** To see whether a printer supports bar code encodings, such as Data Matrix, GS1 DataBar, Royal Mail RED TAG, and USPS Intelligent Mail Container, use the display printer information function in PSF. See [PSF for z/OS: Diagnosis](#) for more information.

A bar code can be created several different ways, but if it is created by using a set of MO:DCA-P structured fields that is called a BCOCA object, the object can then be stored as a resource for easier use.

## Using BCOCA to produce bar code resources

By using a set of structured fields that is called a bar code data object or BCOCA object, you can direct some PSF-supported printers to produce bar codes. A bar code data object specifies the type of bar code (the symbology), its size, and positioning information. Many different bar code symbologies can be produced by use of bar code data objects.

A bar code data object can be included in a print data set or in an overlay resource, or it can be created from bar code specifications in a page definition. PSF accesses resources and sends them to the printer with the print data set.

When you create a bar code data object, you have these options:

- One of several bar code types (symbologies) can be selected.
- The bar code elements can be of any height and width, within the limitations of the symbology.
- The bar code can be printed in one of several colors (on printers that support printing in more than one color). However, you cannot specify color when you are specifying bar codes in a page definition.
- The bar code can be placed at any position on the page.
- The bar code can be rotated.

If the bar code is stored as a resource, you can change some of these properties at print time. For more information about coding bar code data objects and about the options you can specify, see *Mixed Object Document Content Architecture Reference* or *Bar Code Object Content Architecture Reference*. For information about coding bar code specifications in page definitions, see [Page Printer Formatting Aid: User's Guide](#).

## Producing bar codes without BCOCA

If your printer does not support BCOCA, you can print bar codes by using a program that produces non-BCOCA bar codes, such as the IBM Document Composition Facility (DCF) program. DCF uses the MO:DCA-P Presentation Text Data (PTX) structured field. This structured field contains codes that instruct the printer to draw horizontal or vertical rules of different lengths and thicknesses. These bar codes cannot be stored as resources. For more information about using DCF to produce bar codes, see *Document Composition Facility: Bar Code User's Guide*.

You can also use fonts to produce non-BCOCA bar codes. These bar codes are stored as text. For more information about using fonts to produce bar codes, see to [“Using FOCA fonts to produce bar codes”](#) on page 24.

## Using bar code resources

Bar code resources are stored in the page segment library, included through an Include Object (IOB) structured field and are embedded in the data stream at print time. Because bar code resources are stored in the page segment library, IBM recommends a prefix of B1 for bar code resources, so that you can find all the bar code resources easily. Each time that you use a bar code resource, you can define different properties, such as size and rotation.

## Printing bar codes

Not all printers support BCOCA. If your printer does not support BCOCA, you can still print bar codes by producing them with a program that does not give BCOCA output, or by using fonts.

Printers that support bar codes support different bar code symbologies and symbology parameters. Your printer can use default values for unsupported parameters; therefore, verify that the bar codes that are printed by your printer are suitable for your purposes. Bar codes that are generated on printers with different resolutions can differ in length because of the resolution correction that is needed to round the bar code widths to a specific pel size. Each printer can use a different algorithm for this calculation; therefore, you need to test on each type of printer to be used to print bar codes.

For information about the bar code support that your printer provides, see the documentation provided with the printer.

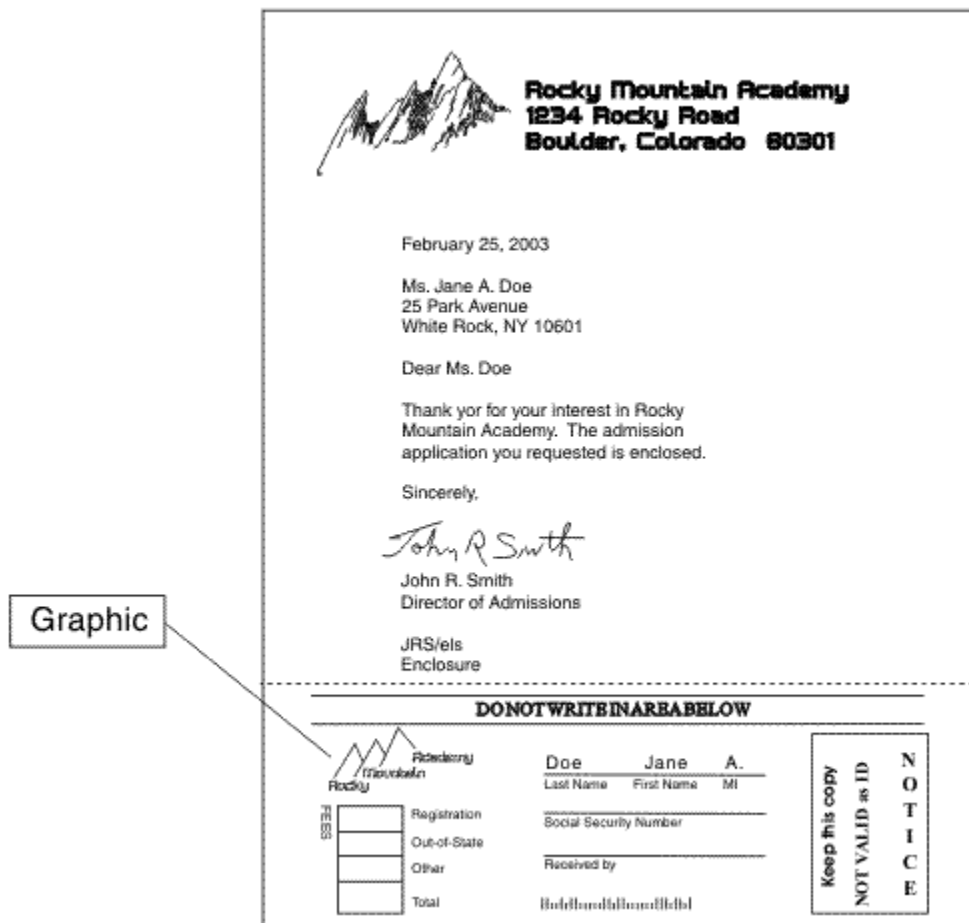
## Graphics

A graphic resource (or GOCA resource) is a Graphics Object Content Architecture (GOCA) object stored as a resource in the page segment library. Graphic resources are soft resources.

Graphic data contains commands to draw lines, arcs, and circles and can be used to represent something as complex as a three-dimensional engineering drawing. It is created by a program and stored in a set of MO:DCA-P structured fields called a GOCA object. This object can then be stored as a resource for easier use.

Along with drawing commands, a graphics object can contain image and character data. Whenever image data is included in a graphics object, code the Image Resolution field in the Graphics Data Descriptor (GDD) to provide image resolution information to the printer, which lets the printer print the image at the correct size. When character data is included in a graphic, all fonts that are used must be mapped in the print data set or overlay or page segment that includes the graphic data object.

Figure 18 on page 42 shows the graphic data printed on the sample page.



**Rocky Mountain Academy**  
1234 Rocky Road  
Boulder, Colorado 80301

February 25, 2003

Ms. Jane A. Doe  
25 Park Avenue  
White Rock, NY 10601

Dear Ms. Doe

Thank you for your interest in Rocky Mountain Academy. The admission application you requested is enclosed.

Sincerely,

*John R. Smith*  
John R. Smith  
Director of Admissions

JRS/eis  
Enclosure

---

**DONOTWRITEINAREABELOW**

**Rocky Mountain Academy**

Registration  
Out-of-State  
Other  
Total

Doe Jane A.  
Last Name First Name MI  
Social Security Number  
Received by  
H0dH0ndH0dH0ndH0dH0dH0d

Keep this copy  
NOT VALID as ID  
N  
O  
T  
I  
C  
E

Figure 18. Graphic on the sample page

## Using graphics

Graphic resources are stored in the page segment library, included through an Include Object (IOB) structured field and are embedded in the data stream at print time. Because they are stored in the page segment library, IBM recommends a prefix of G1 for graphics, so that you can easily find all the graphic resources.

A graphic resource can be included in a print data set, in an overlay resource, in a page segment resource, or saved and used as a resource. For information about the data format for AFP graphics, see *Graphics Object Content Architecture for AFP Reference*.

**Note:** When you are using graphics on a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

## Printing graphic data

Each time that you reference a graphic resource, it can be manipulated in these ways:

- Placed in any position on the page.
- Scaled to any size; that is, enlarged or reduced, by the printer.
- Rotated by the printer.
- Clipped so that only part of the graphic is printed.

For more information about how to code graphics resources and what options can be specified, see *Mixed Object Document Content Architecture Reference*.

Because graphic data is stored in vector representation, it is always resolution-independent. That is, graphic data can be printed at the same size on any printer that supports graphic data objects, regardless of the resolution of the printer.

Most printers, such as the InfoPrint 70, InfoPrint 2000, and InfoPrint 4100, support graphics. To determine whether your printer does, see the documentation provided with the printer.

## Image resources

An image resource<sup>7</sup> is an IOCA object, which is also called a data object resource. If the IOCA resource is to be stored in a system library, you can store it in either the page segment library or the object container library, depending on whether the IOCA resource is a hard or soft resource. If the IOCA resource is identified in the print data set by using a Map Data Resource (MDR) structured field, it is considered a hard resource and must be stored in the object container library. If the IOCA resource is not identified by using the MDR structured field, but is included through an IOB structured field, it is considered a soft resource. As a soft resource it can be stored in either the page segment library or the object container library. IBM recommends that all new IOCA resource objects be stored in the object container library.

Image resources contain a series of picture elements (pels) arranged in rows and columns. Image objects also specify where the image is placed on the page. [Figure 19 on page 44](#) shows the images printed on the sample page.

---

<sup>7</sup> An image resource is also called an IO image object, a raster pattern, or an IOCA resource.

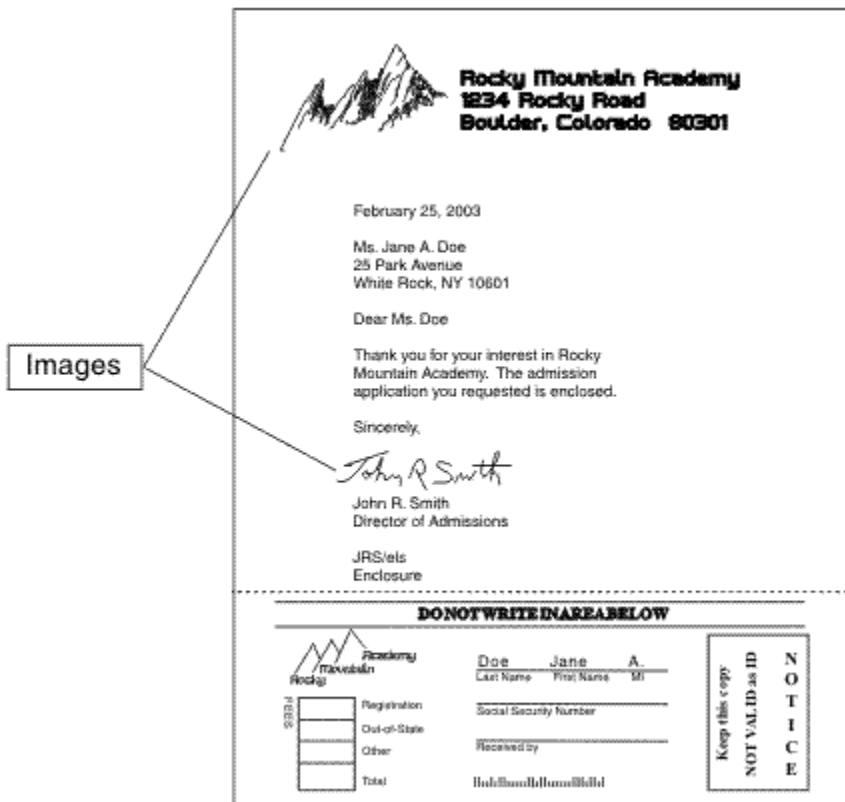


Figure 19. Images on the sample page

Image resources are created by a scanning device or a program and stored in a set of MO:DCA-P structured fields called an IOCA object. The object can then be stored as a resource for easier use.

## Using image resources

Soft image resources can be stored in the page segment library or object container library, can be included through an Include Object (IOB) structured field, and are embedded in the data stream at print time. PSF looks for a soft image resource in this order:

1. Inline in the print data
2. In a user library
3. In the system page segment library
4. In the system object container resource library

If you use the image resource in more than a single page of the document or in multiple documents, it might be better to use the MDR structured field to identify the image resource as a hard resource in the print data set. By using the MDR structured field, PSF can download the image resource once and use it repeatedly in the print data set. If the image resource is being used as a hard resource, it must be stored in the object container library. PSF looks for a hard image resource object in this order:

1. Inline in the print data set
2. In a user library
3. In the system object container library

For the complete order that PSF uses to search for image resources, see [“Searching for resources specified by a print job”](#) on page 14.

Because an image resource is one type of resource in a library of multiple resource types, and PSF does not enforce a prefix for the eight-character name of the image resource, your site can define a naming convention to avoid conflicts with other resources. IBM recommends a prefix of I1 for image resources.

The image resource can be included in a print data set, in an overlay resource, in a page segment resource, or stored and used as a resource. An image resource can be manipulated in these ways:

- It can be placed in any position on the page.
- It can be stored and transmitted to the printer in compressed form, saving storage and transmission time.
- It can be resolution-independent, so that an image created with one resolution (pel density) can be printed at the same size on a printer with a different resolution.
- It can be scaled to any size (enlarged or reduced) by the printer.
- It can be rotated by the printer.
- It can be clipped so that only part of it is printed.

For more information about coding IOCA image data objects and the options you can specify, see *Mixed Object Document Content Architecture Reference* or to *Image Object Content Architecture Reference*.

PSF provides a MO:DCA-P document called IOCAMMR. IOCAMMR contains an IOCA image that is compressed by the IBM MMR compression algorithm. You can print this image on any IPDS printer that supports IOCA. You can also use this document as a test file or print it in hexadecimal to assist an image application developer in understanding the structure of a MO:DCA-P document that contains an IOCA image object. If an image is compressed by use of the CCITT Group4 compression algorithm, you would code it similarly.

**Note:** When you are using images on a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

## Printing images

Most printers, such as the InfoPrint 70, InfoPrint 2000, and InfoPrint 4100, support image resources. To determine whether your printer supports image resources, see the documentation provided with the printer.

Ordinarily, image resources print only on printers that support IOCA images. However, with PSF you can print some image resources even on a printer that supports only IM image data objects. For more information about IM images, see [“Printing images without IOCA”](#) on page 45. These restrictions apply:

- The resolution of the image resource must be the same as the resolution of the printer.
- The image resource must be stored in uncompressed format.
- The image resource must contain only one segment.
- The image resource must specify one of these mapping options (these mapping options do not require scaling or correction of resolution):
  - Image point to pel
  - Image point to pel with double-dot
  - Scale-to-fit, with the input image space that is the same as the output space

When printed on a printer that supports image resources, an image can be resolution-independent. That is, every such printer prints the image at the same size, even if the resolution of the image is different from the resolution of the printer. However, because of differences in the scaling algorithms that are used by various printers, exact fidelity of the image is not guaranteed. To be resolution-independent, you must specify a parameter that tells the printer to correct the resolution. To learn about the mapping options a printer has that can correct the resolution, see the documentation provided with the printer.

## Printing images without IOCA

To print an image on a printer that does not support IOCA, you can use an IM image data object. An IM image data object specifies the content of a raster image and its placement on a page. You can manipulate an IM image in these ways:

- It can be placed in any position on the page.

- It can be enlarged to twice its size, which is the double-dot function.

## Compatibility among printers

All the printers supported by PSF support IM image data objects. However, an IM image might or might not print at the same size on a printer with a different resolution; it depends on whether the printer supports IOCA, as follows:

- If the printer does not support IOCA, IM images that are created with one resolution shrink or expand.
- If the printer does support IOCA, PSF transforms the IM image data object into an IOCA image, taking advantage of the resolution independence available with IOCA image data objects. Thus, the IM image prints at the correct size. However, because of differences in the scaling algorithms that are used by various printers, exact fidelity of the image is not guaranteed.

## Text

---

A text resource is a Presentation Text Object Content Architecture (PTOCA) object with Object Environment Group (OEG) that is stored as a resource in the page segment library. PTOCA objects with OEG resources are soft resources.

A PTOCA object with OEG defines the text in the presentation space and defines the position, rotation, and size of the object area on a MO:DCA page. PTOCA objects with OEG are referenced on an Include Object (IOB) structured field or embedded in an overlay, page, or line data. All fonts that are used must be mapped in the page or overlay that includes the text object.

[Figure 20 on page 47](#) shows the text printed on the sample page.



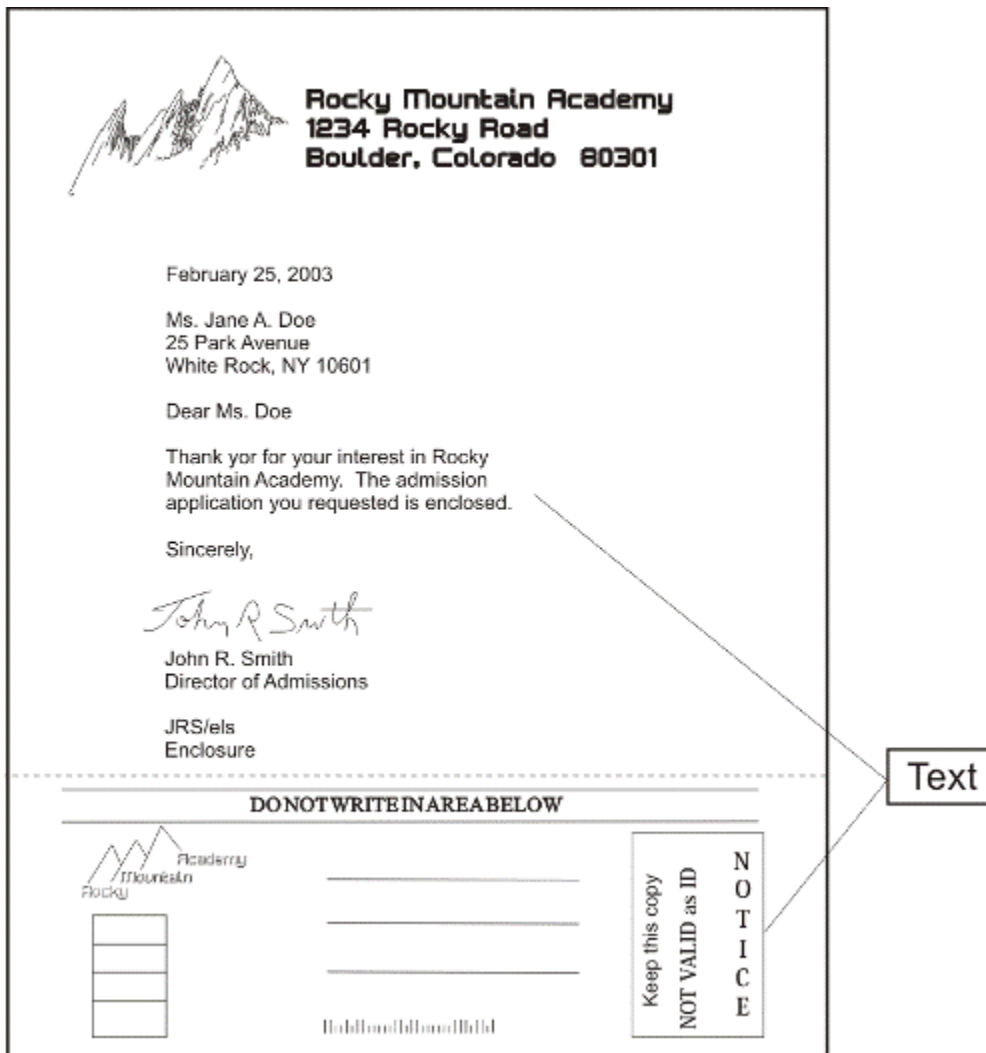


Figure 20. Text on the sample page

## Using text

Text resources are stored in the page segment library. A text resource can be:

- Included through an IOB structured field in a page or overlay.
- Embedded in a page, overlay, or line data, but not in a page segment.
- Included in an inline resource group.

For information about the data format for AFP text, see *Presentation Text Object Content Architecture Reference*.

## Printing text data

Each time that you reference a text resource, it can be manipulated in these ways:

- Placed in any position on the page.
- Rotated by the printer.

A text resource can use a CMR in the OEG to specify color management. For more information about how to code text resources and what options can be specified, see *Mixed Object Document Content Architecture Reference*.

PSF supports text fidelity controls that you use to indicate whether PSF continues or ends processing of a print file when an unsupported PTOCA control sequence is encountered. You can also indicate whether text fidelity errors are reported (see Chapter 11, “Diagnosing incorrect printer output,” on page 173).

## Combining character, image, graphics, and bar code data

PSF can merge variable data from application programs with print resources and can print the data at any location on a page. PSF can merge text, image, graphics, and bar code data to create a composite output of different data types.

Figure 21 on page 48 is a sample page shown in two parts: character data and print resources (graphic, image, and bar code data) merged with it.

February 25, 2003

Ms. Jane A. Doe  
25 Park Avenue  
White Rock, NY 10601

Dear Ms. Doe


Thank you for your interest in Rocky Mountain Academy. The admission application you requested is enclosed.

Sincerely,

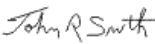
John R. Smith  
Director of Admissions

JRS/els  
Enclosure

Doe   Jane   A.




**Rocky Mountain Academy**  
1234 Rocky Road  
Boulder, Colorado 80301



---

**DONOTWRITEINAREBELOW**



Registration  
Out-of-State  
Other  
Total

Last Name	First Name	MI
Social Security Number		
Received by		
<div style="border-bottom: 1px solid black; height: 15px; width: 100%;"></div>		

Keep this copy  
NOT VALID as ID

**NOTICE**

Character Data
Other Data

Figure 21. Sample containing character data and other data

## Using resources with the distributed print function (DPF) of PSF

When data and resources are sent from a host system (z/OS or IBM i), the distributed print function (DPF) of PSF stores them on a Windows print server on a local area network (LAN) or a wide area network (WAN). DPF drives printers that are attached to the Windows server to print data that is sent from the host.

DPF can store PSF for z/OS and PSF/VSE resources in its libraries. Thus you save the time and expense of having to transmit the resources from the host libraries. PSF for z/OS tells DPF not to store inline resources or resources found in user libraries. Inline resources can be captured on DPF if this function is requested by the system programmer by using Exit 7 or the Printer Inventory. See [PSF for z/OS: Customization](#) for more information.

## Using APSRMARK to mark resources

Resources are marked as PUBLIC or PRIVATE by using a PSF utility called APSRMARK. A mark of PUBLIC instructs PSF to look for a printer-resident version of the resource first, and, if none is found, to download the host resource. A mark of PRIVATE (or no mark at all) instructs PSF to download the host resource.

System programmers generally use APSRMARK to mark new or updated resources. In addition to marking resources such as overlays, page segments, and fonts for DPF, APSRMARK also can be used to mark fonts for resident font activation or capture by the printer.

**Note:** APSRMARK is not used to mark extended code pages.

If you edit a marked resource by using an AFP utility program, such as OGL, but do not alter the original mark, the new version is not downloaded. If you edit a marked resource and delete the original mark, the new version of the resource is considered PRIVATE and is downloaded. If you use APSRMARK to mark the new version of the resource as PUBLIC, that resource is time- and date-stamped by APSRMARK. Therefore, the next time that a job uses that resource, the new host version has a more recent stamp than the current resident version, and the version is downloaded and saved again.

Because the time- and date-stamp for PUBLIC resources are always compared, you are assured of getting the most recent host version of a resource if you use standard IBM utilities for creating your resources. However, if you do not use an IBM utility to directly update the resource object code for a marked resource, you can circumvent the process. If you update the resource without deleting or changing the APSRMARK time- and date-stamp, PSF cannot differentiate between your updated resource and the resident version and uses the resident version instead of the updated version.

For more information about using APSRMARK, see *PSF for z/OS: Customization*, or see your system programmer.

## Using inline resources

---

Resources can be placed in a line data or MO:DCA-P print data set by an appropriate application. Resources cannot be placed in an XML print data set because the resource records would contain carriage control characters (X'5A'), which are not valid in XML data. A resource that is in a print data set is called an *inline resource*. For information about how to code inline resources, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

PSF can use all types of resources sent inline:

- Fonts
- Form definitions
- MO:DCA resources (BCOCA, GOCA, IOCA, and PTOCA with OEG)
- Object container resources
- Overlays
- Page definitions
- Page segments

When the name of the inline resource matches the name of a resource that is used by the print data set, PSF uses an inline resource to print the data set. For example, if the print data set references a page segment that is named S1LOGO, and an inline page segment is named S1LOGO, PSF uses the inline page segment. If the inline resource is marked PUBLIC, PSF might use a printer-resident version of the resource.

PSF stores inline resources temporarily and uses them only for the data set in which they are sent inline. After the data set finishes processing, PSF deletes the resources to prevent any other data set from using them.

The temporary inline resource library exists in virtual storage. This storage can be in either of these areas:

- The region storage area (31-bit storage), the default. If your job has inline resources, it might end because of insufficient virtual storage in the region area.
- Above the bar (64-bit storage). The system programmer can define that all inline resources are stored above the bar. You can also specify that inline resources are stored above the bar for a specific job that ends when inline resources are stored in the region area.

For more information, see [“Specifying that inline resources are stored above the bar” on page 144](#).

PSF searches the temporary inline resource library for a resource before it searches for the resource in the user libraries or system libraries. However, when you use a resource from a security library, PSF searches only the security library. For information about using security resources, see [\*PSF for z/OS: Security Guide\*](#).

PSF can use inline resources (fonts, page segments, and overlays) that are marked for use with resident fonts or for use with the DPF resource library. Your system programmer must mark these resources, by using the APSRMARK program, before you can use them with resident fonts or with the DPF resource library. For more information about using APSRMARK, see [“Using APSRMARK to mark resources” on page 48](#) and [“Using resources with the distributed print function \(DPF\) of PSF” on page 48](#).

Even though resident resources can be activated by inline resources that are marked PUBLIC, PSF tells the printer not to capture inline resources. If you want them made resident in the printer, you must directly install them into the printer (if the printer is capable) or you must put them in a system library. PSF allows only PUBLIC resources from system libraries to be captured, with one exception. Exit 7 can be coded to allow capture of inline resources for printers that are attached through DPF. For more information about the system programmer's role in using resources that are stored in the DPF library or for information about Exit 7, see [\*PSF for z/OS: Customization\*](#).

See [“Printing with inline resources” on page 143](#) for guidelines and examples of specifying inline resources.

## PSF resources supplied by IBM

---

IBM supplies these resources with PSF:

- Form definitions, which are listed in [Appendix A, “Form definitions supplied with PSF,” on page 175](#).
- Page definitions, which are listed in [Appendix B, “Page definitions supplied with PSF,” on page 185](#).

You can also use other IBM licensed programs to create and tailor PSF resources. For information about these programs, see [\*PSF for z/OS: Introduction\*](#).

For information about the fonts that IBM supplies for PSF, see these publications:

- [\*z/OS Font Collection\*](#) for outline, raster, and WorldType fonts contained in the z/OS Font Collection, a base feature of z/OS, Program Number 5650-ZOS.
- [\*IBM AFP Fonts: Font Summary for AFP Font Collection\*](#) for fonts contained in the AFP Font Collection, Program Number 5648-B33. Samples of fonts are shown in [\*IBM AFP Fonts: Font Samples\*](#).
- [\*IBM Infoprint Fonts: Font Summary\*](#) for AFP fonts contained in Infoprint Fonts for z/OS, Program Number 5648-E76, and TrueType and OpenType fonts contained in the WorldType Fonts for AFP Print Servers optional feature of Infoprint Fonts for Multiplatforms, Program Number 5648-E77.

---

## Chapter 4. Formatting and printing data

Every data set that is printed on a page printer requires one or more *copy groups*. A copy group (also called a medium map) controls the printing of a form (a physical sheet of paper). If you are printing on both sides of a sheet, a copy group controls the printing of both sides. If you change copy groups, PSF automatically ejects to a new sheet before it uses the controls in the next copy group.<sup>8</sup>

Copy groups are contained in a *form definition* or one or more *internal copy groups*. PSF uses a form definition or an internal copy group to control the modifying and printing of forms. PSF uses a *page definition* to compose pages from line data.

This information describes internal copy groups, form definitions, and page definitions. You can create form definitions and page definitions by using the IBM Page Printer Formatting Aid (PPFA) licensed program or by using some other products, which might differ from the products described in this information.

---

### Internal copy groups

An internal copy group is a copy group that you define within a print data set instead of within a form definition. You can use internal copy groups to dynamically change any of the functions controlled by a copy group without modifying the form definition. To use an internal copy group:

1. Define the copy group in the print data set, starting with a Begin Medium Map (BMM) structured field and ending with an End Medium Map (EMM) structured field.
2. Immediately follow the copy group definition with an Invoke Medium Map (IMM) structured field to call it.

You can define and call an internal copy group before the first page of data and between pages of data. Each time that you want to call an internal copy group, you must define the copy group and follow it with an IMM structured field, even though you called the same internal copy group previously in the data set.

In any one print data set, PSF can use a combination of internal copy groups and copy groups defined within the form definition. PSF uses the internal copy group instead of one in the form definition if all of these conditions are met:

- The copy group is defined within the print data set.
- An IMM structured field immediately follows the copy group.
- The name of the copy group in the IMM structured field matches the name of the internal copy group that precedes it.

If these conditions are not met, PSF uses a copy group in the form definition, either the first copy group in the form definition or the copy group named in the IMM structured field.

Just as job submitters can specify JCL parameters to override values in copy groups in form definitions, job submitters can also override values in internal copy groups. For example, a job submitter can override the duplex option in all copy groups, including internal copy groups, by specifying the DUPLEX JCL parameter. Some JCL parameters, such as FLASH and COPIES group values, have special considerations that depend on whether the copy group is found within a user-specified or system-default form definition. For these JCL parameters, PSF treats internal copy groups as if they were defined in a user-specified form definition. For more information about these JCL parameters, see [“Using FORMDEF with COPIES or FLASH parameters in JCL” on page 114](#).

#### Notes:

1. You can use conditional processing to select copy groups in form definitions, but you cannot use conditional processing to select internal copy groups.

---

<sup>8</sup> When you are using N\_UP printing, changing copy groups might not cause PSF to eject to a new form.

2. You cannot define internal copy groups in XML data.
3. An internal copy group can be selected only by a structured field in a print data set.

## Form definitions

---

A form definition must contain one or more copy groups. Many of your form definitions contain a single copy group, which means that all the pages of the print job are printed according to the same form specifications. However, if your job requires different specifications for different pages of output, you can use a form definition that contains multiple copy groups.

To begin printing a data set, PSF selects the first copy group in a form definition, unless the data set begins with a control record that includes instructions to select a different copy group. A copy group within a form definition can be selected by a structured field in a print data set or by conditional processing in a page definition. For more information about selecting copy groups in a form definition, see [“Using multiple copy groups or page formats” on page 128](#). For more information about conditional processing, see [“Conditional processing” on page 67](#).

This information describes:

- Specifying printing controls in copy groups
- Using form definitions supplied with PSF

## Printing controls specified in copy groups

The printing controls that can be specified in a copy group include:

- [“Page position” on page 52](#)
- [“Paper source” on page 53](#)
- [“Duplex printing” on page 54](#)
- [“Duplex-page offsets” on page 56](#)
- [“N\\_UP printing” on page 56](#)
- [“Constant forms” on page 57](#)
- [“Page-presentation compatibility” on page 59](#)
- [“Finishing output” on page 59](#)
- [“Offset stacking” on page 60](#)
- [“Print quality level” on page 60](#)
- [“Horizontal-adjustment for the 3800 printer” on page 61](#)
- [“Subgroup modifications” on page 61, including:](#)
  - Overlays
  - Data suppression
  - Paper source
  - Output bin selection
  - Forms flash for 3800 printers

**Note:** When you are sending output to a microfilm device, see [Appendix E, “Microfilm device considerations,” on page 223](#).

## Page position

The copy group assigns horizontal and vertical offsets to position the upper left corner of the logical page on the physical form (sheet). The page position is relative to the medium origin.

The medium origin is the upper left corner of the physical piece of paper (or other medium) being printed on, as seen by the printer microcode. For cut-sheet printers, the medium origin is always the upper

left corner of a sheet viewed with the short side as the top. For continuous-forms printers that support page-presentation compatibility, such as the 3835 and 3900 printers, the medium origin is always the upper left corner of the narrow edge of the form. For more information about medium origin your printer supports, see the documentation provided with the printer.

The page origin is the starting position of the logical page, which contains the user's print data. The logical page is defined in the page definition for line data, or in structured fields for MO:DCA-P data. All the data in the print data set must fit within the boundaries of this logical page.

The form definition positions this logical page on the physical form. The page origin is specified as a horizontal offset and a vertical offset from the medium origin. The page might be positioned at the medium origin, but it is typically offset to avoid unprintable areas or areas too near the edges of forms. For information about printable areas for your printer, see the documentation provided with the printer.

Figure 22 on page 53 shows the relationship between the medium origin and the page origin. The size of the logical page and its print direction are defined in the page definition for line data or in structured fields for MO:DCA-P data. The offset of the page origin is defined in the form definition. In the figure, the horizontal offset is labeled X, and the vertical offset is labeled Y.

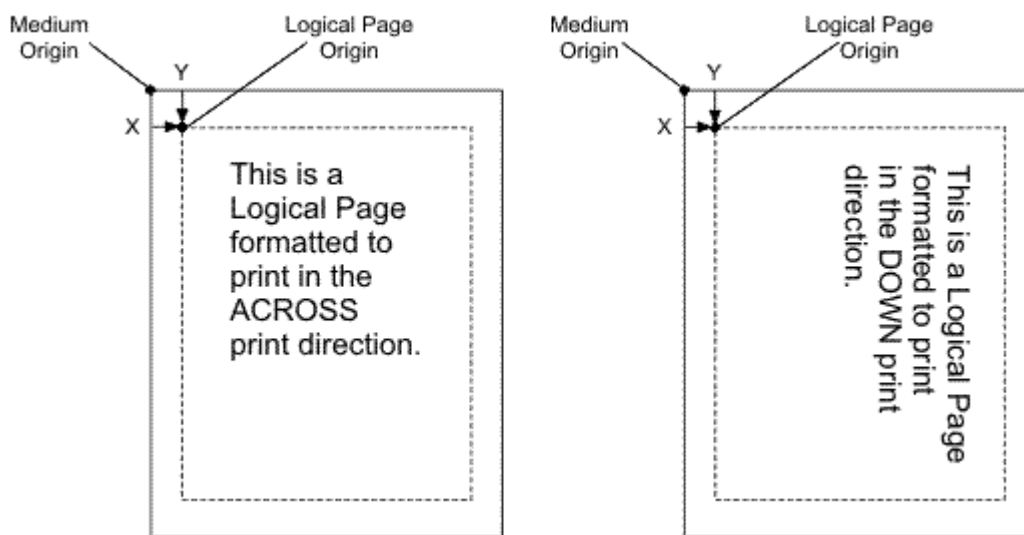


Figure 22. Relationship between the medium origin and the page origin

Page origin is not affected by the print direction specified in the page. As Figure 22 on page 53 shows, changing the page print direction from across to down does not change the position of the page origin.

You can specify different logical page offsets for the front and the back of a duplex printed page. For an additional explanation, see [“Duplex-page offsets” on page 56](#).

## Paper source

To specify the paper source, insert a control in the copy group or a subgroup of it<sup>9</sup> (see [“Subgroup modifications” on page 61](#).) This control indicates whether paper is to be fed from the primary source or from an alternative source. Printers can have up to 255 alternative paper sources. To see what paper sources your printer supports, see the documentation provided with the printer.

**Note:** Some printers support disabled mechanisms. For example, if a form definition specifies the primary source, but that source is disabled, the operator can print the job from an alternative source. To see whether your printer supports disabled mechanisms, see the documentation provided with the printer.

For continuous-forms printers, PSF ignores copy controls for paper source because these printers have only one paper source.

<sup>9</sup> To specify the paper source in the form definition, use "BIN". For an example that shows how to specify the paper source, see [“Specifying bins \(paper source\)” on page 117](#).

## Duplex printing

Some printers can print on one side of the sheet (simplex printing) or on both sides (duplex printing). To see whether your printer can print in duplex mode, see the documentation provided with the printer.

You specify duplex or simplex printing in a control in the copy group. These choices are available:

- To print on only one side, specify no duplex.
- If the sheets are to be bound on the long edge of the paper, as Document A and Document B are in [Figure 23 on page 54](#), specify normal duplex.
- If the sheets are to be bound on the short edge of the paper, as Document C and Document D are in [Figure 24 on page 55](#), specify tumble duplex.

[Figure 23 on page 54](#) shows Document A and Document B printed with normal duplex. Notice that the pages are printed in the *portrait position* for Document A and in the *landscape position* for Document B.

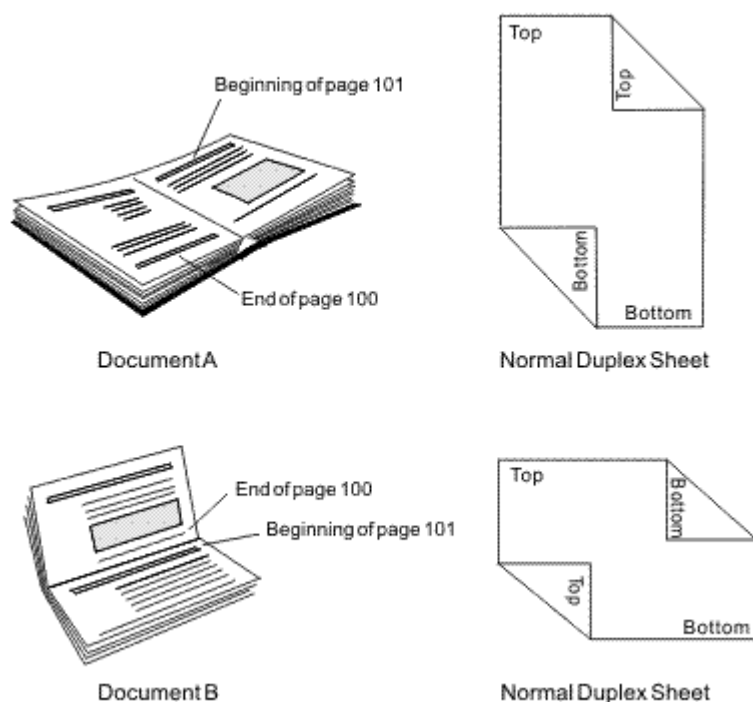


Figure 23. Duplex documents A and B specified as normal duplex

[Figure 24 on page 55](#) shows Document C and Document D printed with tumble duplex. Notice that the pages are printed in the *portrait position* for Document C and in the *landscape position* for Document D.



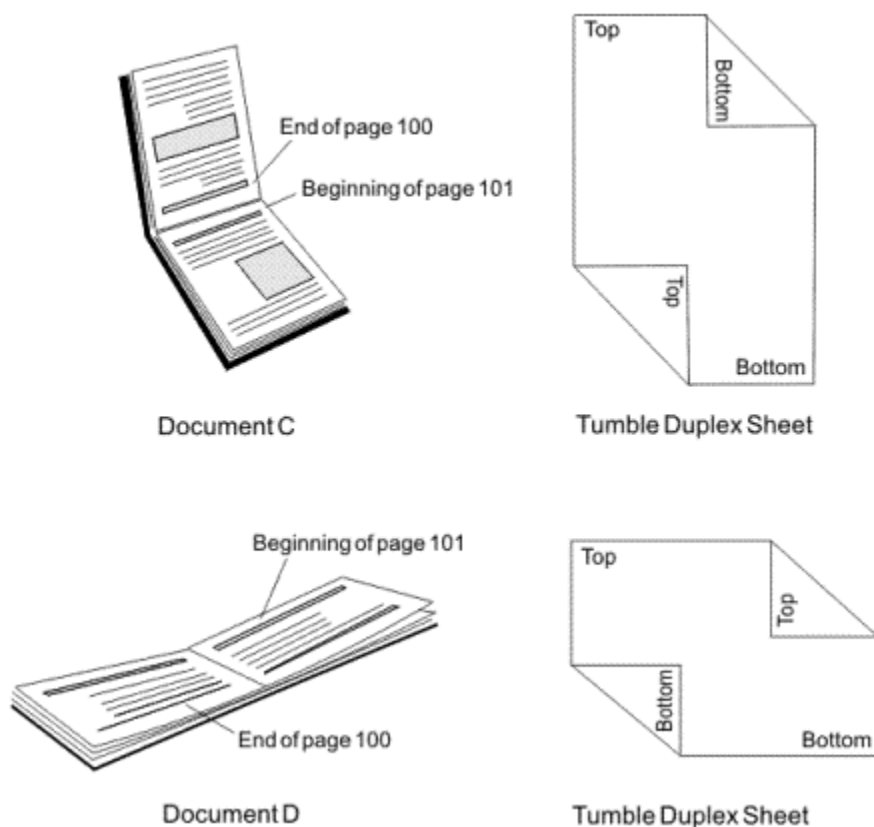


Figure 24. Duplex documents C and D specified as tumble duplex

Some printers can print despite a disabled duplex paper path; other printers are unable to print in duplex printing mode. PSF has the following disabled mechanism support for some PSF-supported printers (your installation might define print classes or destinations that do not have this support for disabled mechanisms):

- If duplex printing is disabled or is not supported for a specific printer, the printer prints in simplex.
- If the primary paper source is disabled, the printer selects paper from one of the alternative paper sources.
- If offset stacking is disabled, no offset stacking occurs.

If a mechanism is disabled, your output might not be as expected, and you might want to reassign output to another printer.

For a printer that does not support printing in duplex or that has a disabled duplex path, PSF processes the duplex control in these ways:

- For no N\_UP printing, PSF processes two consecutive pages of data as a pair. The first page is processed as the front of the sheet, and any controls that are specified in the form definition for the front are applied. The second page is processed as the back of the sheet, and any controls that are specified in the form definition for the back are applied. If you request multiple copies in the form definition, all the copies of front pages are printed first, and then all the copies of back pages are printed.
- For basic N\_UP printing, PSF processes the first  $n$  consecutive pages of data as the front of the sheet, applying any controls specified in the form definition for the front. PSF processes the next  $n$  consecutive pages of data as the back of the sheet, applying any controls specified in the form definition for the back. If you request multiple copies in the form definition, all the copies of the front  $n$  pages are printed first, and then all the copies of the back  $n$  pages.
- For *enhanced N\_UP printing*, PSF is unable to process the duplex control because page buffering and page reordering might be required to simulate duplex printing. In this case, the data cannot be printed.

To see whether your printer can support disabled mechanisms and print in duplex printing mode, see the documentation provided with the printer.

## Duplex-page offsets

You can specify different logical page offsets for the front and back of a duplex sheet. You might want to specify different page positions when, for example, the printed pages are bound or when you are printing on three-hole punched paper. In these cases, you might want to specify a smaller left margin for the back of the sheet to leave room on the right side of the sheet for the binding or the holes.

## N\_UP printing

PSF supports *N\_UP printing*, a printer media-handling support that is specified in the form definition. With N\_UP printing, you can place multiple pages in partitions on a sheet, which enables you to print much more data on a sheet, saving printer-use costs, paper, and storage space.

N\_UP printing differs from multiple-up printing, in that N\_UP printing is specified in the form definition and works with MO:DCA-P data (page data) and with line data. You can use N\_UP printing to place multiple MO:DCA-P pages or line data on a sheet, and to format each of the N\_UP pages differently. In contrast, multiple-up printing is activated in a page definition and works only with traditional line data. The entire multiple-up impression is formatted with a single page format and only appears to have multiple logical pages.

To use N\_UP printing, you need a form definition that specifies N\_UP. (IBM enhanced Page Printer Formatting Aid (PPFA) so you can create form definitions that specify the N\_UP subcommand.) These new form definitions specify the number of pages on a sheet in addition to the other form definition options, such as duplex and page offsets.

The form definition does not control the size of the pages on the sheet. The size of the pages on the sheet is controlled in the page definition for line data, or in the structured fields in a MO:DCA-P page.

PSF supports two levels of N\_UP: *basic* N\_UP and *enhanced* N\_UP. To see whether your printer can support N\_UP, see the documentation provided with the printer.

With *basic N\_UP printing*, you can use a form definition to print up to four pages on one side of a sheet of paper in simplex mode. You can also use a form definition that contains the PPFA N\_UP subcommand to print up to eight pages in duplex mode. With basic N\_UP, you accept the default placement of the pages in the partitions. The partitions are all the same size and are placed one to four per side, depending on the number specified in your N\_UP subcommand. The page must be the correct size to fit within the partition area. For basic N\_UP, the valid printable area (VPA) is the intersection of the partition and the current logical page. [Figure 25 on page 57](#) shows the equal partitions created on a side of a sheet by including the basic N\_UP subcommand in a form definition. The figure shows continuous forms, and cut-sheet forms with both wide and narrow leading edges, feeding into the printer.

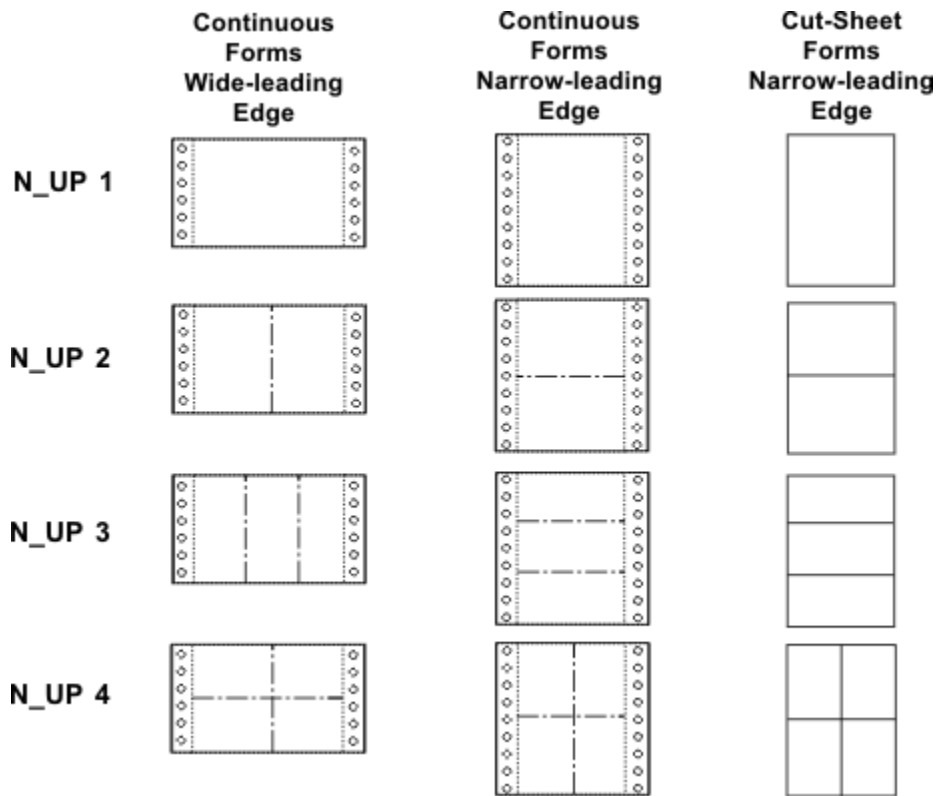


Figure 25. N\_UP printing partitions for various media

With *enhanced N\_UP printing*, you can place pages at any location on either side of the sheet. You can also:

- Place a page relative to any partition origin on either side of the sheet, in any orientation, and in any size that fits on the sheet. You can place multiple pages relative to the same origin, when the total number of pages does not exceed the N\_UP limit for that sheet.
- Place overlays relative to any partition origin, with or without variable page data from the application program.
- Specify a different rotation for each page.
- Specify one or more different overlays for each page.
- Specify a different offset for each page.

*Page Printer Formatting Aid: User's Guide* describes how to use N\_UP printing.

**Note:** When you are sending N\_UP data to a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

## Constant forms

You can use the constant-forms function to print medium overlays or a forms flash (3800 printer only) on blank pages without adding blank pages to your print data set; PSF generates the blank pages on which to print the overlays or the forms flash. These pages that are generated by PSF are called *constant forms* because no variable data from the print data set is printed on the page.

For example, you can print an overlay that contains constant text on the back of each page of a print data set without modifying the data set; you specify the constant-forms function in the form definition. You specify the constant-forms function for an entire copy group; you identify the overlays and form flashes in the subgroups of a copy group. See [“Subgroup modifications”](#) on page 61.

**Note:** When you are printing constant forms to a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

Using the constant-forms function, you can request that PSF generate and print the constant form as the front and/or back side of each sheet in the copy group, as follows:

- For the front side of each sheet:

PSF prints the constant form as the front side of each sheet.

If duplex printing is specified for the copy group, PSF prints the pages from the print data set on the back side of each sheet. The print data set must contain at least one page that is printed by use of this copy group; otherwise, PSF does not generate any constant forms for this copy group.

If simplex printing is specified for the copy group, the print data set must not contain any pages to be printed by use of this copy group; for subsequent pages in the print data set, a different copy group must be used.

- For the back side of each sheet (duplex printing only):

PSF prints the constant form as the back of each sheet, and the pages from the print data set on the front of each sheet.

The print data set must contain at least one page that is printed by use of this copy group; otherwise, PSF does not generate any constant forms for this copy group.

- For both the front and back side of each sheet (duplex printing):

PSF prints the constant form as both the front and back sides of each sheet.

The print data set must not contain any pages to be printed by use of this copy group; a different copy group must be used for subsequent pages in the print data set.

Figure 26 on page 59 shows two pages that are printed by use of a copy group that specifies the constant-forms function for the back side of each sheet. A subgroup in that copy group specified overlay O1CODES for the back side of the sheet.

**Note:** The print data set contains only two pages; PSF generates the pages that are printed as the back sides of the sheets.

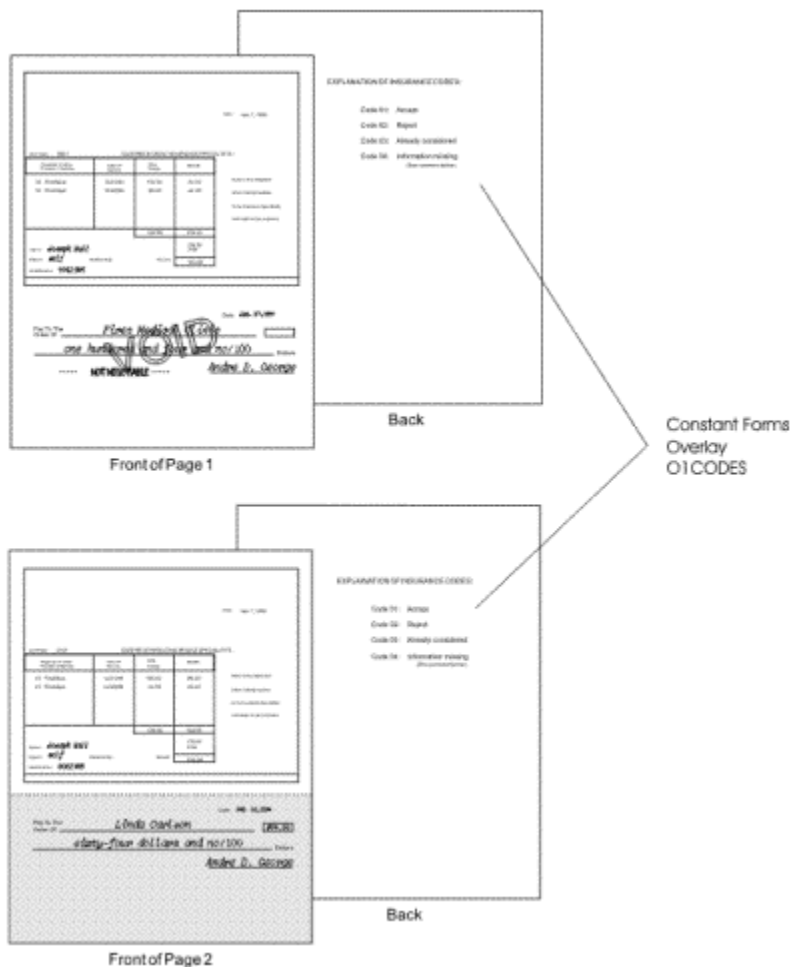


Figure 26. Copy group printed by using the constant-form function

If the constant-forms function is specified in a copy group, but no overlays or forms flashes are specified in the subgroups of that copy group, PSF generates a blank page. In [Figure 26 on page 59](#), the back side of each sheet is blank.

## Page-presentation compatibility

PSF-supported printers can have different hardware (default media) origins. As a result, for compatibility across PSF-supported printers, you might need to use form definitions that contain page-presentation controls. For a description of the default medium origins for your printer, see the documentation provided with the printer.

You can also build other form definitions for page-presentation compatibility by using a tool such as PPFA. For more information about using PPFA, see [Page Printer Formatting Aid: User's Guide](#).

## Finishing output

Finishing refers to an operation you can do with a finisher, such as the InfoPrint 60 finisher. The capabilities of your finisher determine the available types of finishing. Some examples of finishing types include:

- Corner stapling
- Edge stitching
- Saddle stitching
- Z-folding

Specify the finishing request to PSF in a MO:DCA-P Medium Finishing Control (MFC) structured field in the form definition. By specifying the finishing request in an MFC structured field in a copy group, you can control finishing on a collection of pages. For example, you can staple a collection of pages without the requirement that subsequent pages are collected for stapling.

For more information about how to code finishing in a form definition, see *Page Printer Formatting Aid: User's Guide*. For an example of how to request finishing, see [“Finishing your output” on page 151](#). For sample form definitions, see [Table 25 on page 182](#).

## Offset stacking

In offset stacking, the sheets that are printed according to one copy group are stacked offset to the sheets printed according to the preceding copy group. If you want your printed output to be offset-stacked, specify offset stacking in the copy group.

**Note:** If the form definition specifies offset stacking, but the stacker is disabled, the operator can print the job without offset stacking on some printers. To see whether your printer supports disabled mechanisms, see the documentation provided with the printer.

If you want the printed output from a continuous-forms printer to be offset-stacked, the printer must be equipped with a burster-trimmer-stacker (BTS) feature or an equivalent post-processing device (see [“Bursting and stacking continuous-forms paper” on page 139](#)). If your continuous-forms printer does not have a BTS feature but supports edge-marking, the printer changes the edge-markings on the sheets that you specify to be offset-stacked.

The following steps show how to specify offset stacking. Assume that you have a five-sheet data set to be printed with the same set of printing controls, but that you want sheet 3 offset from sheets 1 and 2, and you want sheets 4 and 5 offset from sheet 3.

1. Create a form definition that contains two copy groups that are identical except that copy group A does not specify offset, and copy group B does.

### Notes:

- a. The printer offsets a copy group relative to the previous copy group. If you specify copy group A (which does not specify offset) for sheets 4 and 5, those sheets are not offset from sheet 3.
  - b. Because the same offset is used for all the sheets in a group, they are in one stack.
2. Copy group A is the default copy group. PSF selects it for sheets 1 and 2, which are not offset from any sheets that precede them.
  3. For page 3, offset stacking is specified by either:
    - An Invoke Medium Map (IMM) structured field that specifies copy group B, which is inserted at this point in the data set
    - Conditional processing, which is specified in the page definitionWhichever method is used, sheet 3 is offset from sheets 1 and 2.
  4. At the start of sheet 4, copy group B (the copy group that specifies offset stacking) is selected again, by either of the methods already described. Sheets 4 and 5 are offset from sheet 3, and are stacked together.

## Print quality level

With some printers, you can select different levels of print quality, such as draft or near-letter quality. For higher print quality, printing speed is slower.

To specify the print-quality level, insert a control in the copy group of the form definition.

If you specify a quality level for a printer that supports only one quality level, and if the quality level you specify is not that level, PSF sends an error message and ignores that specification. For information about the levels of print quality that your printer supports, see the documentation provided with the printer.

## Horizontal-adjustment for the 3800 printer

Specify the horizontal-adjustment value as a control in the copy group. This value indicates both the starting print position and the amount of space by which the 3800 printer operator can adjust the position of the printed data to the left or right. Specify the adjustment value as a number of picture elements (pels). For information about the adjustment values for the 3800 printer, see the documentation provided with your printer.

## Subgroup modifications

A copy group contains one or more subgroups, each of which can contain specifications for different versions of a page. In a subgroup, you can specify what modifications are to be made to a page and how many copies of each version are to be printed. The sum of the number of copies that are specified in the subgroups is the total number of copies of each page to be printed.

For simplex printing, you can code a subgroup to specify the modifications for a single page in that subgroup. For duplex printing, you can code a subgroup to specify one set of modifications for both sides of the sheet, or you can code two subgroups to specify a different set of modifications for each of the two sides.

In a subgroup, you can specify these modifications:

- Overlay identification
- Data suppression
- Paper source
- Output bins for paper destination
- Forms flash for the 3800 printer

### ***Overlay identification***

You can identify the names of up to eight medium overlays and one medium preprinted form overlay, a total of nine medium overlays, for each side of a sheet. In [Figure 27 on page 62](#), the INVOICE overlay was printed for the first subgroup, and the PACKLIST overlay was printed for the second subgroup.

### ***Data suppression***

You can identify up to eight suppression names (names that identify fields that are not to be printed), provided they are defined in the page definition. In [Figure 27 on page 62](#), no suppression names were specified for the first subgroup, and three suppression names were identified for the second subgroup: Salesperson, PRICE, and AMOUNT.

### ***Paper source***

In a form definition that is created by using the PPFA program, you can include the BIN subcommand in the subgroup to specify the paper source. Use this subgroup subcommand only for printers with more than one media source and that support bin selection in a subgroup. You can also select the paper source by specifying a component ID (such as 50 for letter-size paper), media name (such as LETTER for letter-size paper), or both. Selecting the paper source frees you from knowing which bin has the paper you want to use. For more information, see [Page Printer Formatting Aid: User's Guide](#).

PSF uses information that is specified in the form definition to determine which paper source to use, according to this hierarchy:

1. Component ID
2. Media name
3. Bin number
4. Default paper source

For example, if a component ID is specified in the form definition, PSF uses it to select the paper source. If no component ID is specified or it is not found in the printer, PSF uses the media name.

Some printers support only bin numbers, some support bin numbers and media names, some support bin numbers and component IDs, and some support bin numbers, media names, and component IDs. By specifying the paper source all ways, you can free yourself from knowing which bin a paper type is loaded in on those printers with media name or component ID support. By specifying the paper source, you can also use a bin number to print on those printers that only support bin numbers.

### ***Output bins for paper destination***

You can use a form definition to include the OUTBIN subcommand in the subgroup to specify the paper destination. Use the OUTBIN subcommand only for printers that have more than one media destination and support output bin selection in a subgroup.

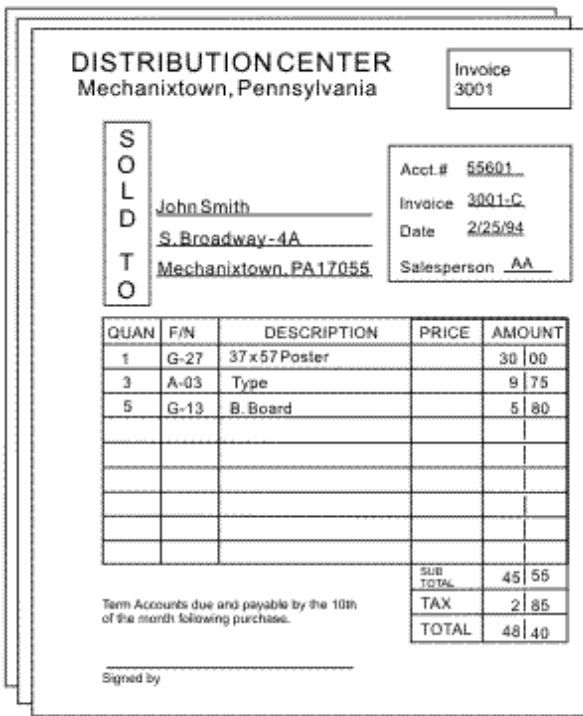
If you specify the OUTBIN parameter in the OUTPUT JCL statement (see “OUTBIN” on page 100), it takes precedent over any OUTBIN specifications in the form definition for the job. If the bin specified in the JCL OUTBIN keyword is not available, or bin selection is not supported by the printer, the printer default bin is used, even if the bin selection was specified in the form definition.

### ***Forms flash for the 3800 printer***

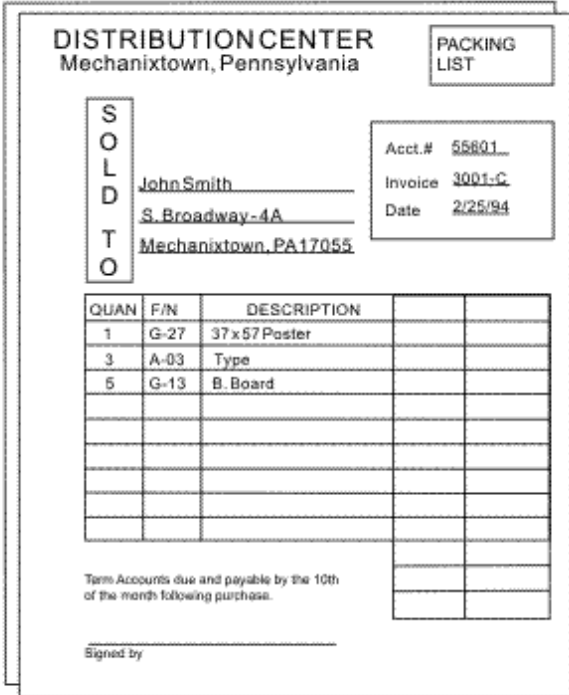
You can specify whether the 3800 printer forms-flash unit prints its negative on each sheet in a subgroup. For the other page printers, PSF ignores this control.

You can also specify that PSF is to print overlays or forms flashes without printing any variable data on certain pages of your output. For example, you can specify that a constant overlay is to be printed on the back of each page of a duplex print job. See “Constant forms” on page 57.

As an example, Figure 27 on page 62 shows a page in a data set that is printed by using a copy group that contains two subgroups. In the first subgroup, three duplicate sheets were printed with a set of modifications; in the second subgroup, two duplicate sheets were printed with a different set of modifications. The two sets of sheets were generated from the same page of information in the data set but were modified differently.



**First Subgroup**



**Second Subgroup**

Figure 27. Subgroups printed from one page of the data set



For more information, see [“Using FORMDEF with COPIES or FLASH parameters in JCL” on page 114.](#)

## Using form definitions supplied with PSF

PSF provides general-purpose form definitions, described in [Appendix A, “Form definitions supplied with PSF,” on page 175.](#) These form definitions specify a page position at the top of the page; one copy of each page is printed, on one or both sides.

The source modules are stored in the SYS1.SAPSPDFD library. You can use the source code to customize form definitions and page definitions for your organization.

## Page definitions

---

A *page definition* is the resource that contains formatting specifications for line data or XML data. This resource is required for any data set or any part of a data set that is not already composed into pages when PSF receives it. No page definition is used for data that is already composed into pages. If you specify a page definition for MO:DCA-P data, PSF ignores it.

A page definition contains one or more *page formats* (also known as *data maps*), each of which contains a complete set of page formatting specifications.

A page format controls the printing of an entire page (a physical sheet of paper) if you are printing on only one side, or one side of a sheet of paper if you are printing in duplex (on both sides of the sheet). If you change page formats, PSF automatically ejects to a new page before it uses the controls in the next page format.

Many page definitions have only one page format. All the pages of the print job are printed to the same specifications. If your job requires different specifications for different pages of output, you can use a page definition that contains multiple page formats.

To begin printing a data set, PSF selects the first page format in a page definition. To specify a page format, insert a structured field in a data set or use conditional processing in the page definition. For more information about conditional processing, see [“Conditional processing” on page 67.](#)

To change from one page format to another, you can use the Invoke Data Map (IDM) structured field, as described under [“Using multiple copy groups or page formats” on page 128.](#)

This information describes:

- Defining page formats
- Using page definitions supplied with PSF

For information about specifying a page definition, see [“Specifying a page definition” on page 121.](#)

## Defining page formats

A page format contains formatting controls for your data set that indicate where and how text, and optionally, page overlays and page segments are to be placed on the page. The page format is defined relative to the origin of the sheet specified in the form definition. A page format can specify one of three types of data:

- Traditional line data, which might contain CCs and TRCs
- Record format line data, which contains record IDs and might contain CCs
- XML data

All page formats in the page definition must specify the same type of data. If you do not specify a type of data, the page format assumes that you are using traditional line data. For information about using types of line data, see [“Line data” on page 71](#) or see *Advanced Function Presentation: Programming Guide and Line Data Reference*. For information about using XML data, see [“XML data” on page 76.](#)

A page format can contain the following information:

- Size of the page area to be formatted, specified as the width and height of the page (required)

- Print direction
- Relative print line positioning
- Options for already formatted print records
- Formatting options for individual fields in print records
- Fonts
- Conditional processing

## Page size

The page size, width, and height are defined in the page format. This area is called the *logical page*. All the text and images that are contained in your print data set must fit within the boundaries of this logical page.

The logical page must cover the entire area of the physical form on which your data can print. If you are printing *multiple-up* applications (in which two or more pages of application data are formatted on the same side of a physical form), you must make your logical page large enough to contain all the pages of application data (this applies only to multiple-up formatting for which a page definition is used; it does not apply to N\_UP printing).

Characters or images that extend outside the boundaries of the logical page cannot be printed. Whether you receive error messages that the characters or images were not printed depends on the value in the DATAACK parameter. For more information, see [“DATAACK” on page 90](#).

Be careful when positioning text near the top or bottom of the logical page area. The position that you specify for character data in your page definition is the position at which the baseline of the characters are printed.<sup>10</sup> When you position character data in a page definition, be sure to leave room for the characters and their ascenders and descenders. For example, never place a character 0 inches down from the top of the page.

Also, be careful not to extend data off the right side of the page, which can happen if your print lines are too long, or if you used a font that is too large to fit within your page area.

In most cases you do not need to be concerned with placing characters too near the left margin of the page. Characters are positioned from the front or leading edge of their baseline so that a character can be placed at the exact left edge of the logical page. However, italic fonts can extend off the left edge of the page because they are *kerned* to tuck under the adjacent characters. Some italic characters extend a few pels to the left of their character space origin. Therefore, when you print with an italic font, be sure to position the text line a few pels to the right of the left edge of the logical page.

For more information about character baselines and positioning, see *IBM AFP Fonts: Font Summary for AFP Font Collection* or [z/OS Font Collection](#).

## Print direction

The page format assigns a print direction to the lines of text in the logical page. The print direction can be one of four inline directions supported by AFP printers:

- ACROSS: oriented 0° from the page origin
- DOWN: oriented 90° from the page origin
- BACK: oriented 180° from the page origin
- UP: oriented 270° from the page origin

In addition to print direction, AFP text can be printed in four character rotations: 0°, 90°, 180°, and 270°. Each of these character rotations is specified relative to the print direction. Unless you specify otherwise, a 0° rotation is used with any print direction you select for your page definition. For more information

---

<sup>10</sup> The baseline is the imaginary line on which characters sit. The character extends above the baseline. Characters such as the lowercase "g" or "y" have pieces that are called *descenders*, which extend below the baseline.

about print direction and character rotation, see [“Printing AFP data in different directions and character rotations”](#) on page 3.

Not all AFP printers can print in every combination of print direction and character rotation. For information about the print orientations your printer supports, see the documentation provided with the printer.

Figure 2 on page 4 shows the different print directions possible on AFP printers. For information about the relationship between print direction and page origin, see [“Page position”](#) on page 52.

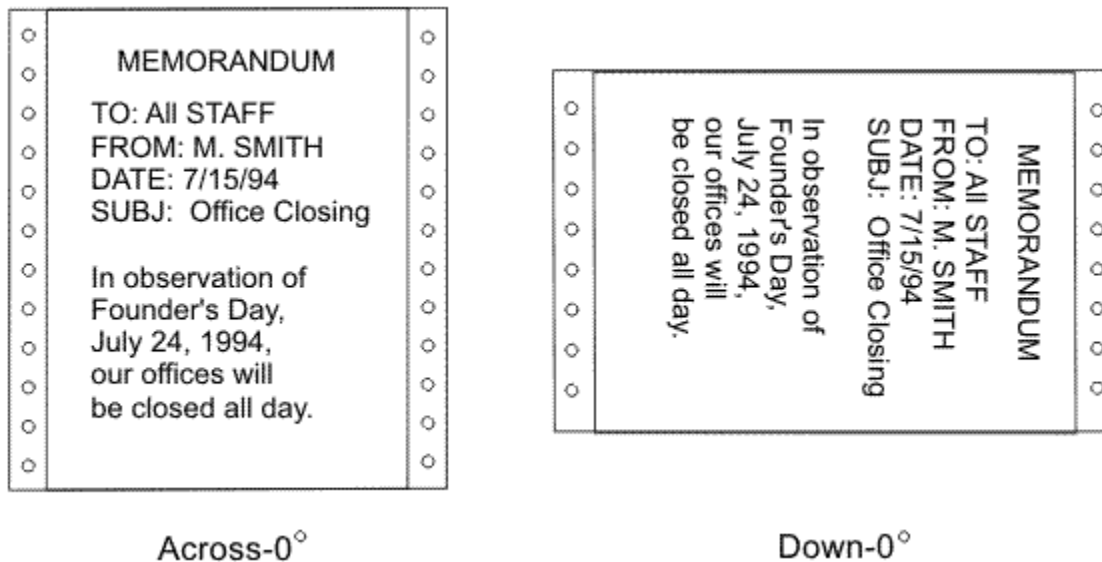


Figure 28. Pages printed in two directions on continuous-forms paper

## Relative print line positioning in a page definition

Relative print line is a function of the page definition that lets formatting float on the page. In some earlier releases of PSF, all page definition formatting was tied to a specific position on the page. With relative print line, values in the print data carriage control byte can determine different formatting to be applied to the current page position. With this function, the application can change record formatting, print line spacing, fonts, overlays, and page segments at any page position, without writing MO:DCA-P structured field commands in the print data.

Relative print line provides increased formatting flexibility. It also preserves the benefit of using page definition formatting, which lets you make changes to the printed output without changing the application program that generates that output.

In many cases, using a page definition to format can reduce the time that is required to develop and test an application. However, relative print line does not eliminate application development requirements: the application must write print records that can take advantage of the formatting capability.

## Page format options for formatted print records

When the print data set contains records that are already formatted into output print lines, you can specify more formatting options in the page format. For this type of data stream, you can use the page format to:

- Specify the number of print lines per page
- Specify the number and starting position for a group of lines on a page
- Specify how carriage controls in the data are to be processed
- Specify the position on the page at which channel codes in the data are to start
- Change line spacings for groups of lines within the page
- Select fonts for the entire page or for groups of lines or fields on a page

- Provide a list of fonts for selection by table reference characters (TRCs) in the data
- Change print directions of one or more lines
- Change the character rotation of a font for one or more lines
- Define and place fields of constant text on each page with the variable data
- Define, place, and select fonts for fields of variable data
- Indicate variable data fields to be generated as bar codes
- Define conditional processing tests on the input record to effect formatting changes
- Include page segments or overlays anywhere on a page
- Provide a list of page segments to be mapped in the printer
- Provide a list of page overlays to be included by the print data set
- Select multiple colors, if the printer supports multiple colors

**Note:** Not all printers support multiple colors. For information about color support for your printer, see the documentation provided with the printer.

- Specify an absolute or relative position for records, fields, or objects

## Page format options for formatting fields

You can use the page definition not only to place records that are formatted into print lines, but also to format individual fields within print records. The page format specifies the position, the print direction, and the font for each individual field that you want printed. You can also prevent fields in the print record from printing by not specifying them in the page format. You can use field formatting to change the output for formatted print records that you are currently printing, or you can use field formatting to print unformatted print records that are not formatted into print lines but that contain only the data to be printed.

This capability of AFP enables an application program to generate only the variable data to be printed. The data fields in the print record can be generated in any sequence by the program. With specifications in the page format, you can use each field more than one time, if you so choose, and place a field at any position on the page. Because the format of the printed output can be changed without affecting the application program, maintaining applications is easier. In addition, if all of the necessary data fields are contained in the print records, new report formats can be created from an existing application by creating a new page definition or page format.

When you format fields from application print records, the page format can be used to:

- Define conditional processing tests based on individual fields in the data.
- Specify the starting position and length in the print record for each field to be formatted.
- Place each field on the page with absolute or relative positioning.
- Specify different print directions for fields within the page, and, optionally, character rotation.
- Select fonts for each field.
- Identify fields that can be suppressed (that is, not printed) on some copies of output or in some transmissions of the output data set.
- Identify fields to be generated as bar codes.
- Define and place fields of constant text on each page with the variable data.
- Select a color for each field, if the printer supports multiple colors.

**Note:** Not all printers support multiple colors. For information about color support for your printer, see the documentation provided with the printer.

Formatted and unformatted print records can be used together in an application program to add formatting flexibility.

## Fonts

You can specify fonts in a page format in several ways:

- You can name a single font for printing all the text on the page.
- You can name different fonts for use with groups of lines, a single line, or data fields in a record.
- You can add a character rotation parameter to a specified font to rotate characters in the inline print direction, as described in [“Printing AFP data in different directions and character rotations” on page 3](#).
- You can specify fonts in a font list for selection by including table reference characters (TRCs) in the data.

If your JCL specifies a page definition that specifies one or more fonts, those fonts are used instead of fonts named in the JCL CHARS parameter (see [“CHARS” on page 86](#)) or UCS parameter (see [“UCS” on page 106](#)). If you want to select FOCA fonts with the JCL CHARS parameter, do not use a page definition that names fonts.

**Note:** You cannot specify TrueType and OpenType fonts with the JCL CHARS parameter. See [“TrueType and OpenType fonts” on page 23](#).

If you are using the default page definition for a printer, you can select fonts with the JCL CHARS or UCS parameter, even if the default page definition specifies a font.<sup>11</sup> A default page definition that is modified by the CHARS or UCS parameter is classified as a modified-default page definition.

If you are using TRCs to select fonts, you can specify the fonts either in a font list in the page format, or in the JCL CHARS parameter. For rules that might affect your decision about where to specify fonts, see [“Using table reference characters to select fonts” on page 133](#).

You can use the PPFA FONT command to select either raster fonts or outline fonts. You can also specify scaling information and size for outline fonts. Coordinate the line spacing with the size of the font you are using.

When the page definition specifies record format processing, font specifications external to the page definition are ignored.

## Conditional processing

You can use the conditional processing function to:

- Define tests to be done on fields in selected input records. These tests are called *conditions*.
- Specify actions that you want PSF to do when certain conditions are met.

The actions PSF can conditionally do are:

- Activate a different copy group. You can do this to change options such as offset stacking and bin selection.
- Activate a different page format. You can do this to change options such as print direction and line spacing.
- Start a new page.
- Start a new sheet.

You can also specify when PSF is to do the conditional actions:

- Before the current line is formatted
- Before the current subpage is formatted
- After the current line is formatted
- After the current subpage is formatted

---

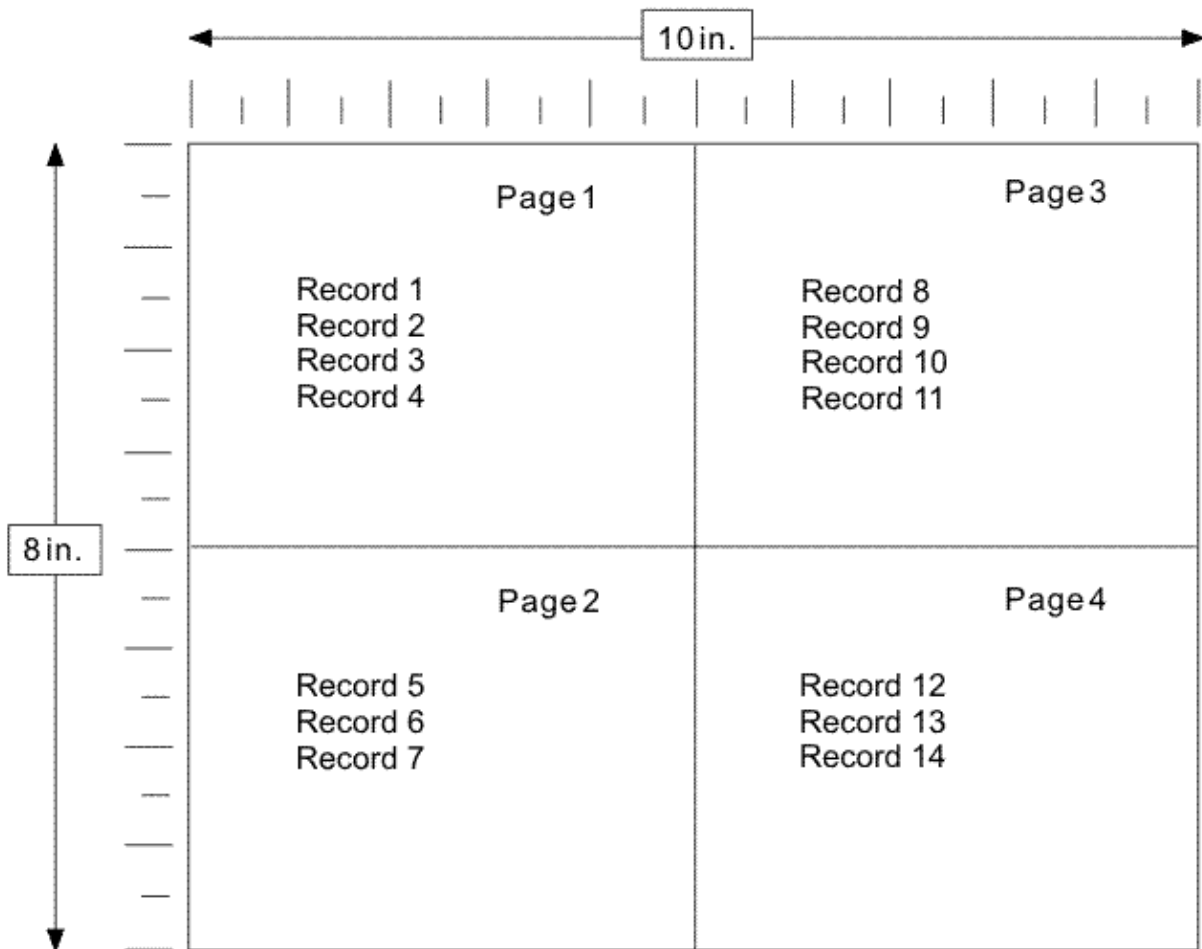
<sup>11</sup> This statement has one exception: If an FCB is named in the JES initialization statements for the printer, and your system programmer created a page definition by the same name, which names a font, that page definition is the printer default, and the font that is named in it cannot be overridden by CHARS or UCS in JCL.

**Note:** A subpage is a logical page or in multiple-up applications, a subpage can be part of a logical page. For a description of subpages in multiple-up applications, see [“Conditional processing for multiple-up applications”](#) on page 68.

Because you can specify that PSF is to change processing options before the current line or subpage, you can test an input record and change the processing options before that record is printed. For example, you can specify that when the contents of a certain field in a record are different from the contents of the corresponding field in the preceding record, PSF starts printing on a new page.

### ***Conditional processing for multiple-up applications***

The printable portion of a sheet can be divided into sections called *subpages*, each having the appearance of a smaller printed page, as in Figure 29 on page 68. Subdividing the printed page in this way is called *multiple-up* printing. In the figure, the traditional line data is printed with subpages to give the appearance of four separate pages. Do not confuse multiple-up printing with N\_UP printing, which places up to four actual logical pages of data on one side of a sheet.



*Figure 29. One logical page divided into four subpages*

You can use multiple-up processing with or without defining sections of the page as subpages. However, you must define subpages to do conditional processing either before or after one of the subdivisions (subpages) of a multiple-up page. You can use only multiple-up printing with traditional line data.

With the IBM Page Printer Formatting Aid program, you can identify the input records that form a subpage. For more information, see [Page Printer Formatting Aid: User's Guide](#).

## Multiple conditions

With conditional processing, you can specify more than one condition and corresponding action for the same input record. You can also specify conditions and actions for more than one line in a subpage. If your page definition contains such multiple conditions, more than one of the conditions can be met.

PSF handles multiple conditions as follows:

- If you define multiple conditions for the same input record, PSF does the specified action for the first condition that is met. PSF does not do any other conditional actions for that input record.
- If you define conditions for more than one input record in the same subpage, and if a condition is met that specifies that PSF is to take an action after the current subpage is formatted, PSF does not do any other conditional actions for subsequent records in that subpage.

## Reprocessing

*Reprocessing* occurs when PSF is requested to do an action either before the current line is formatted or before the current subpage is formatted. In these cases, PSF must reprocess one or more input records, perhaps using a new copy group or page format.

While reprocessing records, PSF does not do some conditional actions that would require additional reprocessing. Because of these restrictions, you might get unexpected results.

When PSF reprocesses records because a condition specified that PSF was to do an action before the current line, this restriction applies:

- PSF does not do actions that are specified to occur either before the line or before the subpage. This restriction is in effect only for the one input record that is being reprocessed.

When PSF reprocesses records because a condition specified that PSF was to do an action before the current subpage, these restrictions apply:

- PSF does not do actions that are specified to occur before the current subpage. This restriction is in effect for all of the input records in the current subpage.
- PSF does not do actions that are specified to occur before the current line. This restriction is in effect only for the first input record in the current subpage.

### Notes:

1. The reprocessing restrictions apply even if you specify that PSF is to change to a new page format before it reprocesses the input records.
2. Do not use the JCL SEGMENT parameter (see “[SEGMENT](#)” on [page 104](#)) and conditional processing at the same time. The JCL SEGMENT parameter causes a data set to be broken up into multiple small data sets, and conditional processing is not supported across data set boundaries. If you use the functions at the same time, the results might be unpredictable.

## Using page definitions supplied with PSF

General-purpose page definitions for printing traditional line data on some of the more common paper sizes are provided with PSF. These page definitions are described in [Appendix B, “Page definitions supplied with PSF,” on page 185](#).

Also supplied are page definitions converted from the FCB modules that were provided with the 3800 Printer Model 1 (FCB3STD1, FCB3STD2, FCB3STD3, FCB36, and FCB38). FCB modules are used with line printers to define the vertical format of printed output: lines per inch, skipping and spacing, and length of form. If you created customized FCB modules for your line printers, you must convert the FCBs to page definitions for use by a page printer. For information about the page definitions in your system libraries, consult your system-support group.

The source modules are stored in the SYS1.SAPSPDFD library. You can use the source code to customize form definitions and page definitions for your organization.





# Chapter 5. Printing different types of data

PSF can print data sets formatted as line data, as XML data, as Mixed Object Document Content Architecture Presentation (MO:DCA-P) data, or as a mixture of line data and MO:DCA-P data. This information describes how to use PSF to print the different data types. It includes this information:

- “Line data” on page 71 contains information to help you create line data records and create MO:DCA-P structured fields that can be mixed with line data.
- “XML data” on page 76 describes how a page definition uses XML start tags to process XML data.
- “MO:DCA-P data” on page 78 describes how to create and use MO:DCA-P data.
- “AFP Conversion and Indexing Facility (ACIF)” on page 80 describes how you can use ACIF with different types of data.

## Line data

This information is intended to help you create line data records in the format expected by PSF. It also contains information that you can use to create MO:DCA-P structured fields. The structured fields are considered general-use programming interfaces and are documented in *Advanced Function Presentation: Programming Guide and Line Data Reference*; see that publication before you create structured fields in your program.

*Line data* is application output to be printed that is not in MO:DCA-P format. To compose pages for the page printer from line data, PSF separates the incoming print records into pages according to specifications in a page definition. A page definition is always required for printing line data with PSF. You can create your own page definition or use a page definition provided with PSF. For more information about page definitions, see Chapter 4, “Formatting and printing data,” on page 51. There are two types of line data: traditional and record format. When the term “line data” is used in this publication, the term applies to both types of line data, unless otherwise specified.

The line data input to PSF can consist of records that are fully formatted; it can consist of records that contain only the fields of data to be printed; or it can consist of records of both types. You can use the page definition resource to format fields of line data outside of the application program.

Figure 30 on page 71 shows the difference between formatted and unformatted line text.

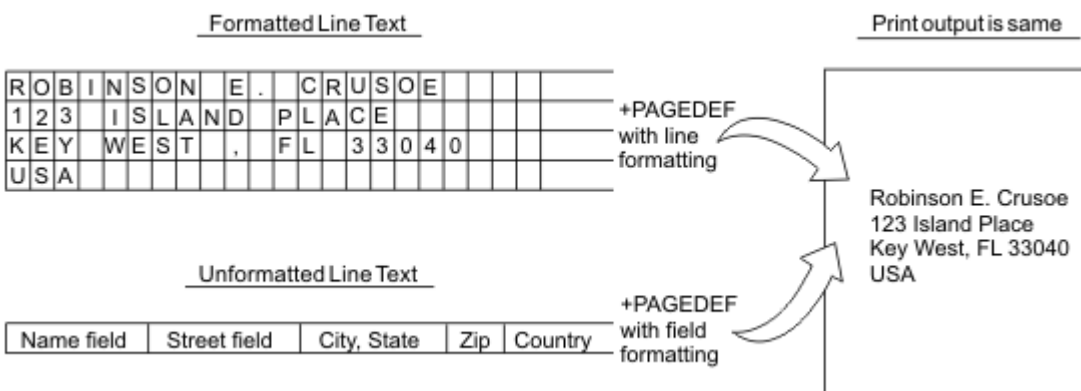


Figure 30. Formatted and unformatted text

## Traditional line data

Traditional line data is data that is formatted for printing on a line printer. Fully formatted line data can be printed on a line printer without a page definition, but all line data needs a page definition to be printed on a page printer.

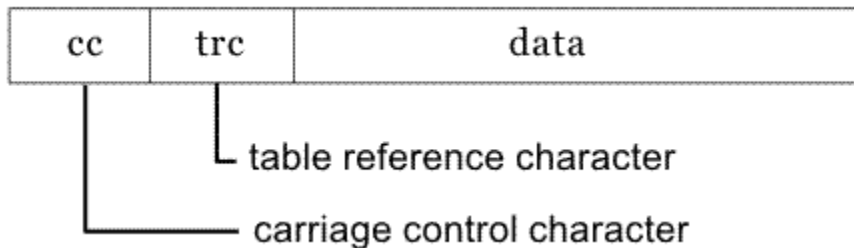
A traditional line data record can contain a 1-byte carriage control (CC) character and a 1-byte table reference character (TRC) followed by the data to be printed.<sup>12</sup> Both characters are optional and are defined as follows:

**cc**

Carriage control character, which defines the positioning, write, space, or skip operation

**trc**

Table reference character, which identifies the font with which the line is to be printed



*Figure 31. Traditional line data record*

Existing applications<sup>13</sup> that generate data that consists entirely of line data records can be printed on a page printer. However, a page definition is required in place of the forms control buffer (FCB) used for line printers. For information about printing jobs that are generated for a 3800 line printer and for jobs that contain merged lines, see [“Merging data lines into a single print line” on page 135](#).

As was noted in [“Using page definitions supplied with PSF” on page 69](#), PSF page definitions are provided for these FCB modules: FCBSTD1, FCBSTD2, FCBSTD3, FCB36, and FCB38. If the name of the page definition is the same as the name of the FCB used with a line printer, no JCL changes are required because the FCB parameter is interpreted as the name of a page definition. However, if the name is different, you must specify the name of the page definition with either the JCL PAGEDEF parameter (see [“PAGEDEF” on page 100](#)) or the FCB parameter (see [“FCB” on page 93](#)).

TRCs are supported for compatibility with 3800 line printers. For traditional line data applications in which the number of lines between font changes is constant, IBM recommends selecting fonts in the page definition. For font selection when the number of lines between font changes is variable, TRCs can provide more flexibility.

## Carriage control characters and table reference characters

Traditional line data can contain CC characters, TRCs, both, or neither. CC characters are used to control writing, spacing, and skipping operations as the data is being formatted. TRCs are used to select the font that prints the record text that contains the TRC.

For more information about using CC characters and TRCs, see [“Specifying carriage control and table reference characters in line data” on page 132](#).

## Data lines merged into a single print line

PSF can merge multiple input data records to print a single print line. When two or more lines of data that contain printable characters are printed in the same line space, the printed line can cause two or more characters to be superimposed. Thus you can print composite characters with line data. You can also use this function to create printed lines in which different fields are printed in different fonts.

<sup>12</sup> With a line printer, the maximum number of data bytes in a single input record is 208. With a page printer, the maximum number is 32768 bytes.

<sup>13</sup> "Existing applications" are applications that are originally designed for formatting and printing data with line printers. For considerations that apply when these applications are run on a page printer, see the documentation provided with your printer.

Some traditional line data applications merge data lines with lines that contain mostly blanks and a few vertical bars. With line printers, this is one way to merge tabular application data with a traditional line data electronic form. This method can more than double the amount of data per page, but at some cost in performance. On AFP printers, you can greatly improve performance by using an overlay to print electronic forms. For more information, see [“Overlays”](#) on page 30.

For more information about merging data lines, see [“Merging data lines into a single print line”](#) on page 135.

## Record format line data

A type of line data that is supported by PSF and formatted by a page definition is record format line data. With this format, each data record contains a record identifier, which selects the record descriptor (RCD) in a record format page definition that is used to format the line data, and might contain a carriage control (CC) byte. Record identifiers are 1–250 bytes. However, within a data map, all record identifiers must be the same length. You can use blanks to make the record identifiers the required length. [Figure 32](#) on page 73 shows the form of record format line data:

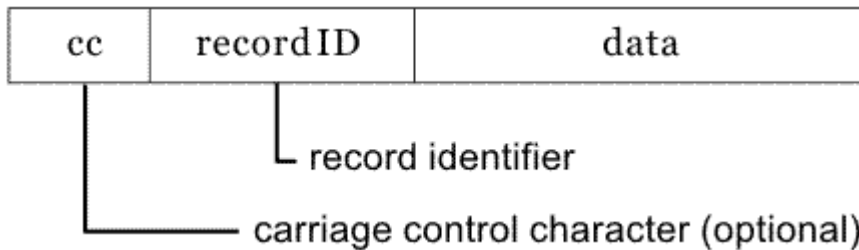


Figure 32. Record format line data record

The CC byte is required when record format data is mixed with MO:DCA-P data, but is ignored. The CC byte is optional for record format line data at all other times, but if you put it in, you must inform PSF that it is there. For information about specifying the CC byte, see [“Specifying carriage control and table reference characters in line data”](#) on page 132.

Many functions that are used in the line descriptor (LND) to format traditional line data are used in the RCD to format record format line data. Others, such as header and trailer processing, are unique to RCDs. For more information about formatting line data, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

Traditional line data is similar to record format line data in that neither is formatted into pages. However, traditional line data can be printed with line printers while record format line data cannot.

## Supported encoding schemes for traditional and record format line data

The encoding schemes that are supported by PSF for traditional and record format line data are:

- EBCDIC (single-byte)
- EBCDIC (double-byte)
- ASCII (single-byte only)
- UTF-8
- UTF-16

If your line data is entirely UTF-8 or UTF-16, the UDType parameter must be specified with the PPFA PAGEDEF command so that PSF looks for a Byte Order Mark (BOM). For UTF-8 data, the BOM is the first 3 bytes of the line data. For UTF-16 data, the BOM is the first 2 bytes of the line data and tells PSF if the data is in big endian order or little endian order. The BOM is required if the line data encoding is UTF-16 in little endian order. Otherwise, the BOM is optional.

All supported line data encodings can be printed with FOCA fonts and TrueType and OpenType fonts. However, single-byte UTF-8 data can be printed only with FOCA fonts and an ASCII code page that you

specified for the font. For more information about using TrueType and OpenType fonts with the supported encodings, see *Using OpenType Fonts in an AFP System*.

## Shift-out, shift-in (SOSI) codes

A data set that contains both single-byte and double-byte character codes can necessitate coding the structured fields in the data set to change from one kind of character code to the other. To avoid doing this, you can instruct PSF to provide special processing by specifying SOSI1, SOSI2, SOSI3, or SOSI4.

The data set can then contain the shift-out code, X'0E', and the shift-in code, X'0F', to indicate to PSF when a font change is required. PSF converts the shift-out and shift-in codes to Set Coded Font Local text controls, described in *Presentation Text Object Content Architecture Reference*.

The conversion for SOSI1 is as follows:

1. Each X'0E' is replaced with a blank (X'40'), followed by a PTOCA structure that contains a Set Coded Font Local text control for the second font.
2. Each X'0F' is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the first font, followed by a blank (X'40').

The conversion for SOSI2 is as follows:

1. Each X'0E' is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the second font.
2. Each X'0F' is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the first font.

The conversion for SOSI3 is as follows:

1. Each X'0E' is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the second font.
2. Each X'0F' is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the first font, followed by two blanks (X'4040').

The conversion for SOSI4 is the same as for SOSI2.

PSF assumes that each line starts with a single-byte font, which means that the data is scanned for the codes 1 byte at a time. After PSF processes a shift-out code, PSF scans the data 2 bytes at a time, examining only the first byte of each pair. Because the scan starts 1 byte at a time on each line, if a line starts with double-byte font data, the first byte in the line must be a shift-out code so that scanning is done 2 bytes at a time.

See [“Specifying shift-out, shift-in \(SOSI\) codes” on page 135](#) for information about the methods of specifying SOSI codes and examples of specifying SOSI codes.

## AFP structured fields included in line data

For more flexibility in formatting your line data applications, you can include certain AFP control records and you can embed certain AFP structured fields in the data. By mixing structured fields with line data, you change the formatting of selected pages in a data set, or include images or blocks of page data on a page, such as:

- Change a copy group to change electronic overlays, duplexing, paper source, or field suppressions.
- Change a page format to change print direction, data formatting, or conditional processing specifications.
- Include page segments from a library.
- Include page overlays from a library.
- Include image data inline with the application print data.
- Use text control sequences to include MO:DCA-P data, or to draw vertical and horizontal rules on a page.

For a description of the structured fields that can be mixed with line data, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

When you mix structured fields with line data records, these apply:

- All records in the data set must contain either the X'5A' control character or one of the valid carriage control characters listed in [Table 5 on page 132](#). If no control characters are specified, the data set can contain only line data records.
- The RECFM subparameter of the DCB parameter must specify "A" for American National Standard control characters or "M" for machine-code control characters.
- If the length of the structured field records is greater than or equal to the logical record length defined for the print data set, the record format of the data set must be specified as variable. These record formats can be specified:
  - Variable ANSI (VA)
  - Variable machine (VM)
  - Variable-blocked ANSI (VBA)
  - Variable-blocked machine (VBM)

If the length of the structured field records is less than the logical record length defined for the print data set, a fixed-length record format can be specified.

- Structured fields that change copy groups or page formats cause PSF to eject to a new page. The first print record that follows one of these structured fields is the first record on the new page.
- Structured fields other than those that change copy groups or page formats do not affect the placement of line data records, nor can they affect the font or the orientation used for printing the line data records. These characteristics of line data records are defined in the page definition. Line data records mixed with structured fields print as defined in the page definition, regardless of whether structured fields are present. For example, if line data record two prints on line 2 of the page when the data set contains no structured fields, the record still prints on line 2 of the page even if a structured field to write MO:DCA-P data or to include a page overlay is written before it.

A structured field can affect the printing of line data records if it contains text control sequences that change the spacing between characters and between words. Spacing between characters and between words in the line data record is changed because these characteristics are not controlled in the page definition.

- Line data records can affect the placement, the font, and the orientation of MO:DCA-P data defined with structured fields. If the font ID, the orientation, or the placement is not defined in the structured field, or if the placement is defined as "relative", the values that are specified in the page definition for the current line data record is used for the MO:DCA-P data.

If ANSI control characters are defined, the current line data record is the line data record that immediately precedes the structured field record. If machine-code controls are defined for the print data set, the current line data record is the line data record following the structured field record.

- Line data records can also affect the placement of page segments, page overlays, and images. The structured fields that are used to include page segments or page overlays can specify absolute position relative to the page origin, or placement at the current print line position. Images included directly in a line data set are always placed relative to the current print line. When you include these objects in line data, you can control their position by controlling the placement of the current line data record, which might involve writing a blank line data record at the position that is desired, or by using carriage control characters to place the current line data record at the correct position.

To include a structured field in a print data set, you can either edit the print data set or include instructions in the application program to put the record in the print data set at the correct location, as in the segment from a sample application program shown in [Figure 33 on page 76](#).

[Figure 33 on page 76](#) shows a segment of a sample application program that is writing an Invoke Data Map (IDM) structured field to change the page format to be DATAMAP2, which defines 90° rotated text with an underscored font. The IDM structured field is written between the fifth and sixth line data records to call a new page format. The sixth line data record becomes the first print record on the new page to

be formatted with the DATAMAP2 page format. Notice that the carriage control byte for the sixth line data record contains the "+" sign, which is the ANSI code to space zero lines. If the carriage control byte is a blank (ANSI code to skip one line before printing), the sixth record does not print on line 1 of the new page, but on line 2.

The coding in Figure 33 on page 76 shows the layout and the description of the IDM record. The structured field begins with the X'5A' carriage control byte. A record descriptor word (RDW) field is coded in front of the structured field because this is an assembly language program that is writing variable-length records. No RDW is coded if fixed-length records are written or if the program is written in a high-level language such as COBOL, which does not require user-coded RDW fields.

For information about coding carriage control bytes or variable-length records, see the publications for your programming language. For descriptions of the AFP structured fields, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

```

.
.
.
      OPEN (PRINT,(OUTPUT)) OPEN OUTPUT DATA SET
*****
*   PUT FIRST 5 LINES OF DATA                               *
*****
PUT PRINT,LINE1
.
.
PUT PRINT,LINE5
*****
*   PUT A Structured Field TO CHANGE THE PAGE FORMAT SO     *
*   THAT THE TEXT IS ROTATED 90 AND PRINTED USING AN UNDER- *
*   SCORED FONT.  A NEW PAGE IS STARTED.                   *
*****
PUT PRINT,IDM
*****
*   PUT REMAINING LINES OF DATA                             *
*****
PUT PRINT,LINE6
.
.
PUT PRINT,LINE10
CLOSE PRINT                                CLOSE OUT DATA SET
.
.
*****
*   DECLARES FOR FIRST 5 LINES OF DATA                       *
*****
LINE1  DS 0CL19                                FIRST LINE
RDW1   DC X'00130000'                          RECORD DESCRIPTOR WORD
DATA1  DC CL15'+THIS IS LINE 1'                CARRIAGE CONTROL AND DATA
.
.
LINE5  DS 0CL19                                FIFTH LINE
RDW5   DC X'00130000'                          RECORD DESCRIPTOR WORD
DATA5  DC CL15'+THIS IS LINE 5'                CARRIAGE CONTROL AND DATA
*****
*   DECLARES FOR THE STRUCTURED FIELD                       *
*****
IDM     DS 0XL21                                INVOKE DATAMAP STRUCTURED FIELD
RDW     DC X'00150000'                          RECORD DESCRIPTOR WORD
CC      DC X'5A'                                5A CONTROL CHARACTER
SFLEN   DC X'0010'                              LENGTH OF STRUCTURED FIELD
SFICODE DC X'D3ABCA'                            HEX CODE FOR INVOKE DATAMAP
SFIFLAGS DC B'00000000'                        FLAGS
SFISEQ  DC X'0001'                              SEQUENCE NUMBER
PARMS   DC C'DATAMAP2                          DATAMAP NAME
*****
*   DECLARES FOR SECOND 5 LINES OF DATA                     *
*****
LINE6  DS 0CL19                                SIXTH LINE
RDW6   DC X'00130000'                          RECORD DESCRIPTOR WORD
DATA6  DC CL15'+THIS IS LINE 6'                CARRIAGE CONTROL AND DATA
.
.
PRINT  DCB DSORG=PS,DDNAME=PRINT,MACRF=PM,BLKSIZE=25,      X
        LRECL=21,RECFM=VBA
END

```

Figure 33. Sample application program

## XML data

This information is intended to help you understand how PSF processes XML data. For a description of XML, see *Extensible Markup Language (XML) 1.0 Specification* at W3C ([www.w3.org](http://www.w3.org)).

The XML Descriptor structured field (XMD) contains information such as data position, text orientation, font selection, field selection, and conditional processing identification. It is used to format XML data,

which consists of information delimited by start and end tags. To process XML data, PSF builds a *Qualified Tag* by concatenating XML start tags. PSF then compares these Qualified Tags to Qualified Tags in the Data Map. PSF builds Qualified Tags for each XMD in the Data Map by concatenating the separate XML Name triplets specified on each XMD. A separate XML Name triplet is specified on an XMD for each XML Start tag that must be traversed to process the content of an XML element.

If an XMD with a matching Qualified Tag is found, the content of the XML element is formatted with that XMD. If an XMD with a matching Qualified Tag is not found, processing resumes with the next start tag.

**Note:** As PSF parses the XML, it must buffer the traversed XML start tags to have a "current" Qualified Tag. Each time an end tag is found, the last matching start tag is removed. Consider this XML hierarchy:

```
<person>
  <name>
    <first>John</first>
    <last>Doe</last>
  </name>
</person>
```

The Qualified Tag for the element `<first>` is `{person name first}`. To process this "current" Qualified Tag, PSF looks for an XMD in the Data Map that has the same Qualified Tag. If found, that XMD is used to present the XML element content John on the page. PSF then repeats the process by building a Qualified Tag for the next start tag, `<last>`, which becomes the new "current" Qualified Tag. The Qualified Tag for `last` is `{person name last}`. Notice that the tag for element `<first>` was removed since its end was received before the start tag for element `<last>`.

The processing that is described in this information is similar to the way the 10-byte record identifier is used for record format line data. After PSF determines which XMD is needed to process the XML element content, the processing is nearly identical to record format processing. For more information about formatting XML data, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

## Element content

When PSF parses the element content:

- Leading and trailing space characters are removed and sequences of space characters are replaced by a single space character. If a sequence of space characters is part of a CDATA section, the sequence is left intact.
- Any data that follows a nested element is ignored. For example:

```
<begin>Here is some data.
  <nested>This element is nested.
  </nested>
  This follows a nested element.
</begin>
```

The data "This follows a nested element." is ignored by PSF.

## External entities

PSF does not parse external entities.

## Supported encoding schemes and conversions

The encoding schemes that are supported by PSF for the XML data are:

- EBCDIC (single-byte only)
- ASCII (single-byte only)
- UTF-8
- UTF-16

XML data encoded in UTF-16 can use either *little endian order* or *big endian order*. PSF checks the first 2 bytes of the data for the Byte Order Mark. If the Byte Order Mark indicates little endian order, PSF converts the data to big endian order. PSF does not support fonts that support the little endian order.

The encoding schemes that are supported for font code pages in a page definition to print XML data are:

- EBCDIC (single-byte only)
- ASCII (single-byte only)
- Unicode Presentation (UTF-16 without surrogate support)

PSF checks the encoding of the XML data against the font encoding to see whether the XML data needs to be converted to another encoding. PSF converts the XML data from:

- ASCII or UTF-8 to UTF-16 if the font encoding is Unicode Presentation. PSF assumes that the ASCII or UTF-8 data converts to UTF-16 data without surrogates.
- UTF-16 to UTF-8 if the font encoding is ASCII. PSF assumes that the UTF-16 data converts to the equivalent of single-byte ASCII when the data is converted to UTF-8. ASCII is a proper subset of UTF-8.

Because EBCDIC data requires EBCDIC fonts, PSF does not convert it.

## JCL parameters

These JCL parameters are ignored when you are printing XML data:

- OPTCD=J
- TRC
- UCS
- CHARS

In addition, the carriage control character specification on the RECFM subparameter of the DCB parameter is also ignored. Carriage control characters are not valid in XML data.

For an explanation of the JCL parameters, see [Chapter 6, “Using JCL for Advanced Function Presentation,”](#) on page 81.

## MO:DCA-P data

---

MO:DCA-P data sets contain only structured fields, whose type and sequence must meet the specifications of the AFP architecture as defined in *Mixed Object Document Content Architecture Reference*.

PSF supports MO:DCA Presentation Interchange Set (IS) data streams, including:

### **MO:DCA AFP/Archive (AFP/A)**

MO:DCA AFP/A is an AFP document architecture interchange set that is used for long-term preservation and retrieval. This subset ensures page independence and eliminates images without clearly specified resolution, device default fonts, and external resources.

### **MO:DCA IS/3**

MO:DCA IS/3 is the first interchange set to achieve industry consensus through a rigorous open standards process. It improves existing functions and introduces new functions, such as Begin Print File (BPF) and End Print File (EPF) structured fields, and multiple image TIFF object support.

### **MO:DCA AFP/A, IS/3**

MO:DCA AFP/A, IS/3 is an AFP document architecture interchange set that complies with the rules and restrictions of both the AFP/Archive and IS/3 interchange sets.

### **MO:DCA Graphic Arts Function Set (GA)**

MO:DCA GA is an extension of MO:DCA IS/3 that adds PDF presentation object support.

You can write your own MO:DCA-P data, or you can use a product that creates the data for you. The following IBM licensed programs produce MO:DCA-P data for output on page printers. When you use any



of these licensed programs to create a document for printing on an AFP printer, the document is created as a data set consisting of structured fields.

### **Document Composition Facility (DCF)**

A general-purpose text-processing program that supports full-page composition of documents, including graphics and images. DCF includes a text formatter, SCRIPT/VS, that processes documents marked up with its own control words and documents marked up with Generalized Markup Language (GML) tags. GML tags, which are shorthand text markup, format the elements of the document.

For information about formatting with DCF, see *Document Composition Facility: SCRIPT/VS Text Programmer's Guide*.

### **Publishing Systems BookMaster® (BookMaster)**

An implementation of GML that uses the SCRIPT/VS formatter to control the formatting of documents for printing or viewing.

For information about formatting with BookMaster, see *Publishing Systems BookMaster User's Guide*.

### **AFP Printer Driver for Windows**

An AFP printer driver that lets any Windows application generate MO:DCA-P output for printing on any printer that is defined to PSF on the host, or on the LAN to a Windows print server.

To download the AFP Printer Driver for Windows, see [Ricoh Production Print Software](http://dl.riohsoftware.com/downloads/aa9a248e-101c-4aa6-b109-1cf7403f3b4f) (dl.riohsoftware.com/downloads/aa9a248e-101c-4aa6-b109-1cf7403f3b4f).

In AFP terms, a compound document is a collection of data objects that makes up the document's content, and the resources and formatting specifications that dictate the processing functions to be done on that content. A MO:DCA-P document can contain a mixture of presentation text, image, graphics, and bar code data objects. This IBM product can assist application programmers in producing compound documents from the MO:DCA-P data stream:

### **AFP Toolbox**

A program that provides access to sophisticated AFP functions through a callable C, C++, or COBOL interface. For more information, see *AFP Toolbox User's Guide*.

## **Formatted data**

MO:DCA-P data that is created by DCF, BookMaster, AFP Printer Driver, or AFP Toolbox contains in its structured fields all the information necessary to describe how the data is formatted on a page. You do not need a page definition to print MO:DCA-P data; if you specify a page definition, PSF ignores it.

A form definition is required to define such functions as page positioning, duplex printing, and overlays, among others. You can use both DCF and AFP Toolbox to change copy groups in the form definition, enabling functions such as duplex printing selectively in a print data set. BookMaster does not provide a tag for changing copy groups.

Unless you specifically code a FORMDEF parameter in the JCL for your job, PSF uses a default form definition. The default might contain controls, such as duplex printing, bin (paper source) selection, and page presentation, that are different from those controls you want for your job. For information about the FORMDEF parameter, see [“Specifying a form definition” on page 112](#). For information about the contents of a form definition, see [“Form definitions” on page 52](#).

## **Font selection**

The page output of DCF, BookMaster, AFP Printer Driver, and AFP Toolbox contains the names of the fonts to be used for the print data set.

## **JCL parameters**

When you print MO:DCA-P data, the RECFM subparameter of the DCB parameter must specify that the data set contains carriage control characters; for example, RECFM=VBA or RECFM=VBM. Because the only carriage control character that is used in MO:DCA-P data is the X'5A' for structured fields, you can specify either ANSI (VBA) or machine (VBM) controls.

These JCL parameters are ignored when you print MO:DCA-P data:

- FCB
- OPTCD=J
- PAGEDEF
- TRC
- UCS

For an explanation of the JCL parameters, see [Chapter 6, “Using JCL for Advanced Function Presentation,”](#) on page 81.

## AFP Conversion and Indexing Facility (ACIF)

---

AFP Conversion and Indexing Facility (ACIF) is a PSF batch application development utility that you can use for several purposes:

- To convert line data, XML data, or mixed data into MO:DCA-P data, which is an architected, device-independent data stream used for interchanging documents between different operating systems.

ACIF formats line data and XML data according to instructions in a page definition and converts it into an AFP data stream file. ACIF accepts any data stream that PSF can process, including line-printer data, XML data, the AFP data stream, and MO:DCA-P. The output can then be sent to any AFP-supported system for printing or to a workstation for viewing.

- To index a document to improve viewing, archiving, or retrieving individual pages or groups of pages from large documents; or to create a separate *index object file* from the indexing tags.
- To retrieve and package AFP resources that are needed for printing or viewing a document and place them in a separate file; to enable viewing and printing of the exact document, possibly years after its creation.

ACIF identifies and processes microfilm setup resources for creating AFP output to a microfilm device. For information about using ACIF, see [AFP Conversion and Indexing Facility User's Guide](#).

---

## Chapter 6. Using JCL for Advanced Function Presentation

This information describes the JCL statements and parameters you use to print a job with PSF, and gives abbreviated examples. For complete task examples, see [Chapter 7, “Printing tasks and examples,” on page 111](#).

Each page printer is started with an assigned set of initialization parameters that identify resource libraries and establish the initial printing environment.<sup>14</sup> You can specify JCL parameters in OUTPUT and DD statements to do tasks such as:

- Assign OUTPUT statements to a DD statement.
- Select the printer.
- Indicate when checkpoints are taken.
- Specify how error messages and data checks are to be handled during printing.
- Specify resources to be used.
- Specify output characteristics.
- Specify font information.
- Specify a form name for channel attached printers (except the 3820 printer).
- Specify a user resource library.
- Specify the resolution at which the output was formatted.
- Request that PSF notify you or another user when your print job is finished.
- Select a forms flash for the IBM 3800 printer.
- Override print-labeling defaults (for authorized users only). For more information about print labeling, see [PSF for z/OS: Security Guide](#).
- Indicate whether to generate an AFP Statistics (AFPSTATS) report.
- Submit jobs to functions provided by Infoprint Server, including the Print Interface subsystem.

For complete details about parameters and syntax, see the JCL reference publication for your operating system.

**Note:** If your installation has a license for the Infoprint Server feature of z/OS, you can specify AFP parameters to PSF on the `lp` command. See [z/OS Infoprint Server User's Guide](#) for a description of the `lp` command.

---

### Determining printer defaults

When JES starts a printer, JCL in the PSF startup procedure establishes libraries. The initial parameters for the printer are set up from Infoprint Server Printer Inventory or the startup procedure. These parameters become the default values for the system. For an explanation of the JCL in the startup procedure, the Printer Inventory, and a general description of how PSF selects resources, see [PSF for z/OS: Customization](#).

---

### Page printer defaults form

Before you submit a file for printing, you need to know the defaults for each printer in your installation. At the minimum, you must know the printer selection parameters for the printer you want to use. See your system support group for this information.

---

<sup>14</sup> PSF-supported printers are commonly, but not always, controlled by OUTPUT and data definition (DD) statements. For more information, see the JCL publications.

For a handy reference, you can copy the form in Appendix D, “Page-printer defaults form,” on page 221 and fill it in with the defaults for your installation. Complete a form for each printer you use. You can then select a printer with the defaults you need by referring to the information you have on the forms. You can also determine the parameters that you must specify for your file. As an example, a completed form is shown in Figure 34 on page 82.

#### PAGE-PRINTER DEFAULTS FORM

PAGE-PRINTER Type is: InfoPrint 60

Printer selection parameters:CLASS= B

DEST= REMOTE

FLASH= \_\_\_\_\_

1. Media or form usually loaded in printer:  
 For continuous-forms printer, size of form usually loaded: \_\_\_\_\_  
 For continuous-forms printer with a burster-trimmer-stacker (BTS), output usually burst? YES | NO  
 For cut-sheet printer, type of media usually in each source:  
Bin 1 8 1/2 x 11 (white paper) Bin 2 8 1/2 x 11 (blue paper)  
 Bin \_\_\_\_\_ Bin \_\_\_\_\_
  2. DATAACK = BLOCK | UNBLOCK | BLKCHAR | BLKPOS
  3. CKPTPAGE = 0 or CKPTSEC = \_\_\_\_\_
  4. PIMSG = NO | YES Message count = 16
  5. FORMDEF= F1A10110  
 Copy Group= F2A10110  
 Page Position= 0.165", 0.165"  
 Paper source for cut-sheet printers: BIN1 | BIN2 | BIN \_\_\_\_  
 Duplex: NO | NORMAL | TUMBLE | RNORMAL | RTUMBLE  
 For 3800 printer:Flash activated? YES | NO
  6. PAGEDEF = P1A06462  
 Form size = 8 1/2 x 11  
 No. of print lines = 64  
 Printable area = 8.17" x 10.67"  
 Lines per inch = 6  
 Print direction:ACROSS | DOWN | BACK | UP
  7. Default fonts are (listed in TRC sequence):  
 0 = 60D8 1 = \_\_\_\_\_ 2 = \_\_\_\_\_ 3 = \_\_\_\_\_  
 If PAGEDEF is not specified in JCL:  
 Fonts CAN | CANNOT be changed using CHARS \*
  8. Fonts supported by printer: Resident | Host | Both  
 Double-byte fonts supported? YES | NO  
 If yes, default PRMODE: SOSI1 | SOSI2 | SOSI3 | SOSI4  
 PSF Exit 7 has raster fonts automatically mapped to outline fonts for this printer: YES | NO
  9. Printer attached to DPF or PSF direct? YES | NO
  10. Other characteristics of this printer: \_\_\_\_\_
  11. COMSETUP= H1SETUPD (microfilm device only)
  12. Where are the messages sent: Printed with output | Printed on printer | On output class
  13. COLORMAP = M1RESET
- \* Fonts can be changed unless a JES default forms control buffer (FCB) (page definition) defined for this printer specifies fonts.

Figure 34. Page-printer defaults form example

## Creating JCL for direct-printing mode

When a job requires direct-printing mode, the PRINTDEV JCL statement is included in the JCL for the job. No PSF startup procedure is used. Some parameters can be specified in the PRINTDEV JCL statement rather than in an OUTPUT or DD statement. The PRINTDEV statement is described in [PSF for z/OS: Customization](#).

Some of the parameters that are described in this information do not apply if the printer is in direct-printing mode. If a parameter does not apply, its description includes a statement to that effect.

**Note:** Some printers cannot be used in direct-printing mode. To see whether your printer supports direct-printing mode, see the documentation provided with the printer.

## Creating JCL for microfilm devices

PSF supports sending AFP data to microfilm<sup>15</sup> devices. You can use the same data set for printing jobs on paper or microfilm by specifying a parameter in your JCL. For information about the JCL parameters, see “Specifying JCL parameters for microfilm jobs” on page 148.

One physical AFP file can have multiple output destinations. JES does not purge the spool file until all outputs are satisfied. Because of this, separate OUTPUT statements can be designated, for instance, one for paper and one for microfilm, for the same print job on the JES spool. Your system programmer can provide the system setup. For more information about using AFP on microfilm devices, see *PSF for z/OS: Customization*.

**Note:** You cannot send AFP jobs to a microfilm device attached through the distributed print function (DPF) of PSF. A microfilm device cannot be driven in direct-printing mode.

## Assigning OUTPUT statements to a DD statement

To assign one or more OUTPUT statements to the DD statement, use the OUTPUT parameter in your DD statement. The OUTPUT parameter causes the parameters on the OUTPUT statement to apply to that DD statement. The names of the OUTPUT statements are referred to by the OUTPUT parameter in the DD statement.

All OUTPUT statements that are assigned to a DD statement must precede the DD statement. The system processes a separate output data set for each OUTPUT statement listed in the OUTPUT parameter of a DD statement. The following example shows the OUTPUT statements that are assigned to the DD1 DD statement. In this example, the system produces separate output data sets for each of the OUTPUT statements: a data set for OUTPUT1, and a data set for OUTPUT2.

```
//OUTPUT1 OUTPUT DEST=name  
//OUTPUT2 OUTPUT DEST=name,FORMDEF=fdefname,PAGEDEF=pdefname  
//DD1 DD SYSOUT=A,OUTPUT=(*.OUTPUT1,*.OUTPUT2)
```

Some parameters can be specified only in a DD statement, and some can be specified only in an OUTPUT statement. Others can be specified in either a DD statement or an OUTPUT statement. If you specify the same parameter in both statements, PSF uses the DD statement parameter rather than the OUTPUT statement parameter. For example, if you use this JCL, the printer destination that is selected is PRT0, not PRT1:

```
//OUTPUT1 OUTPUT DEST=PRT1  
//DD1 DD SYSOUT=A,DEST=PRT0,OUTPUT=(*.OUTPUT1)
```

In the examples in this information, DD statements use specifically assigned OUTPUT statements. However, an OUTPUT statement can be assigned to a DD statement by default by specifying DEFAULT=Y on the OUTPUT statement. For more information about this assignment or about the DD and OUTPUT statements, see the JCL reference publication for your operating system.

## Specifying AFP parameters in the JCL

AFP parameters for PSF printer tasks can be specified in JCL on the OUTPUT statement, in a DD statement, or both. Some parameters that you specify on the OUTPUT JCL statement have equivalent parameters that you can specify in the DD statement. For some tasks (such as selecting a printer class), the parameter you specify in the DD statement is not the same as the parameter you would specify on the OUTPUT statement. For other tasks (such as selecting a printer destination name), the same parameter can be specified in either statement.

### Notes:

1. If a parameter for the same task is specified in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement.

<sup>15</sup> Microfilm can mean either microfiche or 16 mm film.

2. If you do not specify a JCL parameter, you can use defaults that are specified in the Printer Inventory, PSF startup procedure, or JES initialization statements. For details, see *PSF for z/OS: Customization*.

Table 4 on page 84 shows the AFP parameters that you can specify in the JCL and lists whether you can specify a parameter on the OUTPUT JCL statement, in the DD statement, or both.

Table 4. AFP parameters in JCL			
AFP Parameters	OUTPUT	DD	See
AFPSTATS=YES   NO	✓		<a href="#">“AFPSTATS” on page 85</a>
AFPPARMS='dsname[(membername)]'	✓		<a href="#">“Specifying the AFPPARMS control statement on the OUTPUT statement” on page 144</a>
BURST=YES   NO	✓	✓	<a href="#">“BURST” on page 86</a>
CHARS=(fontname1[,fontname2][,fontname3][,fontname4])	✓	✓	<a href="#">“CHARS” on page 86</a>
CKPTPAGE=pages	✓		<a href="#">“CKPTPAGE” on page 87</a>
CKPTSEC=seconds	✓		<a href="#">“CKPTSEC” on page 87</a>
CLASS=name	✓		<a href="#">“CLASS” on page 87</a>
COLORMAP=membername	✓		<a href="#">“COLORMAP” on page 88</a>
COMSETUP=membername	✓		<a href="#">“COMSETUP” on page 88</a>
CONTROL=PROGRAM   SINGLE   DOUBLE   TRIPLE	✓		<a href="#">“CONTROL” on page 89</a>
COPIES=(nnn,(groupvalue,groupvalue...))	✓	✓	<a href="#">“COPIES” on page 89</a>
COPYCNT=(nnnnnnnnnn)	✓	✓	<a href="#">“COPYCNT” on page 90</a>
DATACK=BLOCK   UNBLOCK   BLKCHAR   BLKPOS	✓		<a href="#">“DATACK” on page 90</a>
DEST=[node.]name   '[node.]IP:ipaddr'	✓	✓	<a href="#">“DEST” on page 91</a>
DPAGELBL=YES   NO	✓		<a href="#">“DPAGELBL” on page 92</a>
DUPLEX=NO   NORMAL   TUMBLE	✓		<a href="#">“DUPLEX” on page 93</a>
FCB=pdefname	✓	✓	<a href="#">“FCB” on page 93</a>
FLASH=(flashname,[count])	✓	✓	<a href="#">“FLASH” on page 93</a>
FORMDEF=fdefname	✓		<a href="#">“FORMDEF” on page 94</a>
FORMLEN=xx.yyyIN   xx.yyyCM	✓		<a href="#">“FORMLEN” on page 95</a>
FORMS=formname	✓		<a href="#">“FORMS” on page 95</a>
INTRAY=nnn	✓		<a href="#">“INTRAY” on page 96</a>
LINECT=nnn	✓		<a href="#">“LINECT” on page 96</a>
NOTIFY=(node.userid1[,node.userid2][,node.userid3][,node.userid4])	✓		<a href="#">“NOTIFY” on page 97</a>
OFFSETXB=nnnn[.mmm]unit	✓		<a href="#">“OFFSETXB” on page 97</a>
OFFSETXF=nnnn[.mmm]unit	✓		<a href="#">“OFFSETXF” on page 98</a>
OFFSETYB=nnnn[.mmm]unit	✓		<a href="#">“OFFSETYB” on page 98</a>
OFFSETYF=nnnn[.mmm]unit	✓		<a href="#">“OFFSETYF” on page 99</a>
OUTBIN=(1-65535)	✓		<a href="#">“OUTBIN” on page 100</a>
OVERLAYB=ovlyname	✓		<a href="#">“OVERLAYB” on page 100</a>
OVERLAYF=ovlyname	✓		<a href="#">“OVERLAYF” on page 100</a>
PAGEDEF=pdefname	✓		<a href="#">“PAGEDEF” on page 100</a>
PIMSG=YES   NO   (YES,nnn)   (NO,nnn)   (,nnn)	✓		<a href="#">“PIMSG” on page 101</a>
PRMODE=SOSI1   SOSI2   SOSI3   SOSI4   aaaaaaaa	✓		<a href="#">“PRMODE” on page 102</a>

Table 4. AFP parameters in JCL (continued)			
AFP Parameters	OUTPUT	DD	See
<b>PRTEROR=HOLD   QUIT   DEFAULT</b>	✓		<a href="#">“PRTEROR” on page 102</a>
<b>PRTQUEUE='printqueue'</b>	✓		<a href="#">“PRTQUEUE” on page 103</a>
<b>RESFMT=P240   P300</b>	✓		<a href="#">“RESFMT” on page 103</a>
<b>SEGMENT=page-count</b>		✓	<a href="#">“SEGMENT” on page 104</a>
<b>SUBSYS=(subsystem_name, printer_definition_name, job_attributes)</b>		✓	<a href="#">“SUBSYS” on page 104</a>
<b>SYSAREA=YES   NO</b>	✓		<a href="#">“SYSAREA” on page 105</a>
<b>SYSOUT=(class,,formname)</b>		✓	<a href="#">“SYSOUT” on page 105</a>
<b>TRC=YES   NO</b>	✓		<a href="#">“TRC” on page 106</a>
<b>UCS=fontname</b>	✓	✓	<a href="#">“UCS” on page 106</a>
<b>USERLIB=('libname1','libname2',...,'libname8')</b>	✓		<a href="#">“USERLIB” on page 107</a>
<b>USERPATH=('libpath1','libpath2',...,'libpath8')</b>	✓		<a href="#">“USERPATH” on page 107</a>

The user JCL parameters in [Figure 35 on page 85](#) add information to the separator pages printed with an output data set and can help in distributing your printed output. You specify these parameters on the OUTPUT JCL statement.

```
ADDRESS=address
BUILDING=building
DEPT=dept
NAME=name
ROOM=room
TITLE=title
```

Figure 35. Additional JCL parameters for distributing output

For a description of the parameters for distributing output, see [“Additional parameters to help in distributing output” on page 108](#).

## AFPSTATS

Specifies whether you want to generate an AFPSTATS report. An AFPSTATS report gives you detailed information about the print file, such as where resources were found and what significant events happened. The report is generated as a softcopy file but can be formatted and printed by using a page definition provided by IBM. This option is ignored if the AFPSTATS repository is not set up by the system programmer or if the AFPSTATS bit (XTP7ASAP) in Exit 7 is set to OFF by the installation exit.

Specify the AFPSTATS parameter in an OUTPUT statement, as follows:

### AFPSTATS=YES | NO

The values are:

#### Y or YES

Specifies that an AFPSTATS report is generated.

#### N or NO

Specifies that an AFPSTATS report is not generated. NO is the default.

**Example:** In this example, an AFPSTATS report is requested:

```
//OUT1 OUTPUT AFPSTATS=YES
//PRINT1 DD SYSOUT=A,OUTPUT=*.OUT1
```

For more information about the AFPSTATS report, see [Chapter 10, “Obtaining AFP statistics,” on page 167](#).

## BURST

Specifies whether you want continuous-forms paper separated into single sheets with a post-processing device that bursts, trims, or stacks, or sent to the continuous-forms stacker. If your data set is printed on a cut-sheet printer, PSF ignores the BURST parameter.

Specify the BURST parameter either in a DD statement or in an OUTPUT statement. If you specify the parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement. If you do not specify the BURST parameter, PSF uses the JES installation default.

### **BURST=YES | NO**

The values are:

#### **Y or YES**

Specifies that the printed output is separated into separate sheets in the burster-trimmer-stacker (BTS).

#### **N or NO**

Specifies that the output is to be sent to the continuous-forms stacker (CFS) in continuous fanfold form.

For channel-attached printers, the printer operator receives a message that the output is burst. For SNA-attached and TCP/IP-attached printers, the **Issue setup messages** parameter in the Printer Inventory or the SETUP parameter on the PRINTDEV statement can be used to notify the printer operator that the output is burst. For more information about these parameters, see [PSF for z/OS: Customization](#).

See [“Bursting and stacking continuous-forms paper” on page 139](#) for examples of using the parameter.

## CHARS

Specifies the member name of the FOCA coded font that you want to use to print a data set. Coded fonts that can be used with the CHARS parameter are supplied with the IBM AFP Font Collection and the z/OS Font Collection. For details about the available fonts and the naming conventions, see [IBM AFP Fonts: Font Summary for AFP Font Collection or z/OS Font Collection](#).

**Note:** You cannot specify TrueType and OpenType fonts with this parameter. Instead, use [“USERPATH” on page 107](#). See [“TrueType and OpenType fonts” on page 23](#) for information about referencing TrueType and OpenType fonts.

Specify the CHARS parameter either in a DD statement or in an OUTPUT statement, as follows:

**CHARS=(fontname1[,fontname2][,fontname3][,fontname4])**

The value is:

#### **fontname**

Specifies the 1–4 alphanumeric character name of a FOCA coded font (in a font library). When you use CHARS to specify the member name, do not include the two-character prefix of the coded-font name (X0 through XG).

When you specify fonts with CHARS, remember:

- If PSF uses a default page definition that names a font, and that page definition is specified as the JES default FCB for your printer, you cannot use the CHARS parameter to override fonts that are specified in that page definition. See [“Specifying and selecting fonts” on page 126](#) for the order in which PSF selects fonts.
- If you specify more than one font with the JCL CHARS parameter, you must use the TRC parameter to tell PSF which font to use for each line of data.
- The fonts that you specify must reside in a library that is assigned to the printer you are using or in a user library that is specified with the JCL USERLIB parameter, or else they must be inline with the print data set.
- Raster versions of the outline fonts are used unless the system programmer specifies the **Map to outline fonts** parameter in the Printer Inventory, the MAP2OLN parameter in the PRINTDEV



statement, or XTP7MTOF in PSF Exit 7, and the font name conforms to the CHARS naming convention. Generally, specify raster fonts in the CHARS parameter.

- You can use a CHARS parameter in jobs printed on 3800 line printers, without changing your JCL.

See [“Specifying and selecting fonts” on page 126](#) for examples of using the CHARS parameter.

## CKPTPAGE

Specifies how many pages you want between data set checkpoints.

Specify the CKPTPAGE parameter in an OUTPUT statement, as follows:

**CKPTPAGE=pages**

The value is:

**pages**

1–32767

If you do not specify CKPTPAGE or if you specify both CKPTPAGE and CKPTSEC, PSF uses the JES installation default. See the JCL reference for your operating system.

See [“Specifying printer checkpoints” on page 153](#) for examples of using the CKPTPAGE parameter and additional considerations.

**Note:** When you send output to a microfilm device, see [Appendix E, “Microfilm device considerations,” on page 223](#).

## CKPTSEC

Specifies how many seconds you want between data set checkpoints.

Specify the CKPTSEC parameter in an OUTPUT statement, as follows:

**CKPTSEC=seconds**

The value is:

**seconds**

1–32767

If you do not specify CKPTSEC or if you specify both CKPTPAGE and CKPTSEC, PSF uses the JES installation default. See the JCL reference for your operating system.

See [“Specifying printer checkpoints” on page 153](#) for examples of using the CKPTSEC parameter and additional considerations.

**Note:** When you send output to a microfilm device, see [Appendix E, “Microfilm device considerations,” on page 223](#).

## CLASS

Specifies the output print class for printing the data set.

To specify an output class, either use a SYSOUT parameter on the DD statement, or use a CLASS parameter on an OUTPUT statement assigned to that DD statement. You must specify a class in one statement or the other. If you specify a parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement.

**CLASS=name**

The value is:

**name**

Specifies one alphanumeric character for the output print class name.

If a single printer is defined for an output class, CLASS is all that you need to specify. However, if an output class contains a group of printers, select a printer from the group by specifying a destination

parameter (see “[DEST](#)” on page 91). If you do not, the system selects the first available printer in the class.

#### Examples:

1. To select a printer when only one printer is defined for an output class, specify the SYSOUT parameter on the DD statement:

```
//DD1 DD SYSOUT=A
```

2. The SYSOUT parameter on the DD statement overrides the OUTPUT JCL CLASS parameter. To use the OUTPUT JCL CLASS parameter, you must code a null class on the DD statement, such as SYSOUT=(,).

```
//OUTPUT3 OUTPUT CLASS=C,FORMDEF=fdefname  
//DD3 DD SYSOUT=(,),OUTPUT=(*.OUTPUT3)
```

## COLORMAP

Specifies the member name of the object container for the color mapping table resource. PSF uses the COLORMAP parameter only when it sends output to a printer that supports color mapping table resources.

Specify the COLORMAP parameter in an OUTPUT statement, as follows:

#### **COLORMAP=membername**

The value is:

##### ***membername***

Specifies one to eight alphanumeric or national characters for the color mapping table resource name. The first character cannot be numeric. The full member name must be specified; PSF does not add a prefix. IBM recommends a prefix of M1 for color mapping table resources.

If a job is sent to a printer that supports color mapping, and COLORMAP is not specified in the JCL, PSF uses the color mapping table specified in the Printer Inventory or PSF PRINTDEV as a system default. If no system default is specified, PSF uses the value M1RESET. A color mapping table that is named M1RESET is provided internally with PSF. This table resets all mapping to null. If a color mapping table named M1RESET is available in a user library or PSF object container library, it is used instead of the internal M1RESET table.

You can create your own color mapping table by using the Color Mapping Tool that is included with PSF (see “[Creating color mapping tables](#)” on page 156) or you can use an existing resource created by your system programmer.

The color mapping table that is specified on the COLORMAP parameter must be stored in a PSF system object container library, in a user library specified by the USERLIB parameter, or coded inline as part of the print data set. See “[Specifying color mapping tables](#)” on page 149 for examples of using the COLORMAP parameter, including what you need to do to use color mapping table resources inline or from a user library.

## COMSETUP

Specifies the member name of the object container for the microfilm setup resource. PSF uses the COMSETUP parameter only when it sends output to a microfilm device (see [Appendix E, “Microfilm device considerations,”](#) on page 223).

Specify the COMSETUP parameter in an OUTPUT statement, as follows:

#### **COMSETUP=membername**

The value is:

##### ***membername***

Specifies one to eight alphanumeric or national characters for the microfilm setup resource name. The first character cannot be numeric. The full member name must be specified; PSF does not add a prefix. IBM recommends a prefix of H1 for microfilm setup resources.

The microfilm setup resource must be stored in a PSF system library, in a PSF user library, or coded inline as part of the print data set. You can create your own microfilm setup resource (see the publications that are provided with your microfilm device) or use a resource already available on your system. See [“Specifying JCL parameters for microfilm jobs” on page 148](#) for examples of using the COMSETUP parameter, including what you need to do to use microfilm setup resources inline or from a user library.

## CONTROL

Specifies that each logical record in traditional line data starts with a carriage control (CC) character or that the output is printed with single, double, or triple line spacing.

Specify the CONTROL parameter in an OUTPUT statement, as follows:

**CONTROL=PROGRAM | SINGLE | DOUBLE | TRIPLE**

The values are:

### PROGRAM

Specifies that the DCB=RECFM subparameter on the DD statement defines the type of CC character each logical record in the data set begins with. CC characters are either American National Standards Institute (ANSI) or machine code. PROGRAM is the default. See [“Specifying carriage control and table reference characters in line data” on page 132](#).

### SINGLE

Specifies that the output is printed with single spacing.

### DOUBLE

Specifies that the output is printed with double spacing.

### TRIPLE

Specifies that the output is printed with triple spacing.

**Example:** This example uses an OUTPUT statement and a DD statement to specify that the output data set is printed with a machine code CC character as the first character of each logical record:

```
//OUTPUT1 OUTPUT CONTROL=PROGRAM
//DD2 DD SYSOUT=A,OUTPUT=* .OUTPUT1,DCB=RECFM=VBM
```

## COPIES

Specifies the number of copies you want to print, up to 255 copies.

Specify the COPIES parameter in a DD statement or in an OUTPUT statement. If you specify the parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement. If you do not specify the COPIES parameter (or you specify COPIES=0), and the COPYCNT parameter is not specified, the default is one copy printed.

**COPIES=(nnn,(groupvalue,groupvalue...))**

The values are:

### *nnn*

Specifies the one- to three-digit decimal number 1 - 255 that indicates the total number of collated copies you want to print. The data set is sent to the printer *nnn* times, and each copy is printed in page-number sequence. This parameter applies only to the user's data set. Only one transmission is processed for separator pages and for the PSF message data set.

See the COPYCNT parameter if you want to print more than 255 copies. If you specify a number value on both the COPIES and COPYCNT parameters, the value that is specified on COPYCNT is used. If a group value is specified on the COPIES parameter, PSF ignores the number value that is specified on the COPIES and COPYCNT parameters.

### *groupvalue*

Specifies the one- to three-digit decimal number 1 - 255 that indicates the number of copies of each page of the data set to be printed consecutively before the next page is printed. You can specify up to eight copy groups (group values), and the data set is sent to the printer one time for

each group. The total number of copies equals the sum of the group values. No single group value can be more than 255.

If you specify one or more group values, PSF ignores the number value that is specified on the COPIES and COPYCNT parameters.

If a FORMDEF parameter is specified for the print data set, the number of group values that are specified determines the number of times the user's print data set is transmitted. Only one transmission is processed for separator pages and for the PSF message data set. In this case, group values that follow the first group value are ignored. However, the number of copies of each page is determined by information in the form definition. The effects of specifying FORMDEF and COPIES are described in [“Using FORMDEF with COPIES or FLASH parameters in JCL” on page 114](#).

For more information about printing multiple copies of the job header or the job trailer pages, see [PSF for z/OS: Customization](#).

See [“Printing more than one copy” on page 137](#) for examples of using the COPIES parameter.

**Note:** When you specify copies to a microfilm device, see [Appendix E, “Microfilm device considerations,” on page 223](#).

## COPYCNT

Specifies the number of copies you want to print, up to 2 GB (2,147,483,647) copies.

Specify the COPYCNT parameter in a DD statement or in an OUTPUT statement. If you specify the parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement. If you do not specify the COPYCNT parameter (or you specify COPYCNT=0), and the COPIES parameter is not specified, the default is one copy printed.

### **COPYCNT=(nnnnnnnnnn)**

The value is:

**nnnnnnnnnn**

Specifies the one- to ten-digit decimal number 1–2147483647 that indicates the total number of collated copies you want to print. The data set is sent to the printer *nnnnnnnnnn* times, and each copy is printed in page-number sequence. This parameter applies only to the user's data set. Only one transmission is processed for separator pages and for the PSF message data set.

If you specify a number value on both the COPIES and COPYCNT parameters, the value that is specified on the COPYCNT parameter is used. If a group value is specified on the COPIES parameter, PSF ignores the number value that is specified on the COPIES and COPYCNT parameters.

## DATAACK

Specifies whether you want the printer to block print-positioning and incorrect-character errors. A print-positioning error is an attempt to print outside the valid printable area. An incorrect-character error is an attempt to use a code point that is not assigned to a character.

**Note:** When you specify the DATAACK parameter to a microfilm device, see [Appendix E, “Microfilm device considerations,” on page 223](#).

Specify the DATAACK parameter in an OUTPUT statement, as follows:

### **DATAACK=BLOCK | UNBLOCK | BLKCHAR | BLKPOS**

The values are:

#### **BLOCK**

Specifies that the printer is not to report data-check errors. The printer does not return error messages to PSF, even if data is lost. No data check error messages are created. BLOCK is the default.

## UNBLOCK

Specifies that the printer is to report all data-check errors. If a print-positioning error occurs, the exception is highlighted on the printed page to help you find the place at which the printer attempted to print outside the valid printable area.

## BLKCHAR

Specifies that the printer is not to report incorrect-character errors. The printer reports print-positioning errors as usual, with exception highlighting to show the locations of the errors.

## BLKPOS

Specifies that the printer is not to report print-positioning errors. The printer reports incorrect-character errors as usual.

**Example:** This example uses an OUTPUT statement to specify that data-check errors are not to be blocked:

```
//OUTPUT1 OUTPUT DATAACK=UNBLOCK  
//DD1 DD SYSOUT=A,OUTPUT=(*.OUTPUT1)
```

When you specify a DATAACK parameter, remember this:

- Not all printers perform exception highlighting, and the types of highlighting differ among printers. To see whether your printer supports exception highlighting, see the documentation provided with the printer.
- If you receive a message that states that a print-positioning error occurred, you can submit the print job again and specify BLOCK or BLKPOS for the DATAACK parameter to see how much of the job prints without the error. Output that is produced in this manner is sometimes acceptable.
- If you notice that data is missing from your output, and you specified DATAACK BLOCK or BLKPOS, you can submit the print job again, specifying UNBLOCK or BLKCHAR to receive PSF messages about any print-positioning errors that PSF finds. This method does not apply to incorrect-character errors.
- If you do not specify the DATAACK parameter, PSF uses the default from the PSF startup procedure. If you are using the Page-Printer Defaults form, the default is shown on the form. If no default is specified, PSF uses the default value, DATAACK=BLOCK.

Also, see “PIMSG” on page 101, which determines whether PSF prints the error messages that are generated by a data check.

## DEST

Specifies a destination name or TCP/IP address for the print data set when you are using the Download for z/OS and AFP Download Plus features of PSF. For more information about using the DEST parameter with these features, see [PSF for z/OS: Download for z/OS](#) and [PSF for z/OS: AFP Download Plus](#).

Specify the DEST parameter in a DD statement or in an OUTPUT statement, as follows:

**DEST=[node.]name | '[node.]IP:ipaddr'**

The values are:

### node

Specifies a 1–8 alphanumeric character node for a printer attached to a different system. The node is optional unless you are specifying a destination on a remote system.

### name

Specifies a 1–8 alphanumeric character name assigned to a printer on your system. When *node* is specified, indicates a printer attached to a different system.

Not every printer has an assigned destination name. To determine what destination name, if any, you can use for your printer, see [Appendix D, “Page-printer defaults form,” on page 221](#) or consult your system programmer.

**IP:*ipaddr***

Specifies a TCP/IP destination address for a printer, where *ipaddr* is a character string of 1 to 124 printable characters. If *node* is used with the IP address, the entire value is limited to 127 characters. The IP address must be enclosed in single quotation marks.

With the DEST parameter, remember:

- If you specify a DEST parameter both in the DD statement and in the OUTPUT statement, PSF ignores the parameter in the OUTPUT statement.
- When your job uses direct-printing mode, the DEST parameter does not apply.

**Examples:**

1. This example is of a DD statement that specifies DEST:

```
//DD1 DD SYSOUT=A,DEST=PRT38
```

2. In this example, the print data set that is specified on the DD statement selects the destination printer specified in the OUTPUT statement:

```
//OUT1 OUTPUT DEST=PRT19
//DD2 DD SYSOUT=A,OUTPUT=(*.OUT1)
```

3. In this example, JES sends the print data set to a node and printer name on a remote system:

```
//OUT2 OUTPUT DEST=SANJOSE.BLDPRT3
```

4. In this example, JES sends the print data set to a printer at the specified IP address:

```
//OUT3 OUTPUT DEST='IP:192.117.84.53'
```

5. In this example, JES sends the print data set to a node and IP address for a printer on a remote system:

```
//OUT4 OUTPUT DEST='AUSTIN.IP:192.118.96.54'
```

**DPAGELBL**

Specifies whether PSF prints the security identification label on each page of printed output. The identification label represents a security level and categories as defined to Resource Access Control Facility (RACF®).

The identification label that PSF prints is determined by the SECLABEL parameter of the JCL job card. If you do not specify SECLABEL on the job card, the identification label is printed for the security level of the print job. For more information about specifying the SECLABEL parameter, see [PSF for z/OS: Security Guide](#).

You can specify the DPAGELBL parameter in an OUTPUT statement, as follows:

**DPAGELBL=YES | NO**

The values are:

**Y or YES**

Specifies that PSF prints a security identification label on each page of printed output.

**N or NO**

Specifies that PSF does not print a security identification label on each page of printed output.

**Example:** In this example, the security identification label, which is specified on the SECLABEL parameter of the job statement, is printed on each page of output printed on forms named VP20:

```
//JOBA JOB 1,'JOHN DOE',SECLABEL=CONF
.
.
//VPRPT OUTPUT DPAGELBL=YES,FORMS=VP20
```

## DUPLEX

Specifies whether printing is to be done on both sides of each sheet. If DUPLEX is not specified, the value that is specified in the form definition or the internal copy group is used.

You can specify the DUPLEX parameter in an OUTPUT statement, as follows:

### **DUPLEX=NO | NORMAL | TUMBLE**

The values are:

#### **NO**

Specifies the printing is to be done on only the front side of each sheet.

#### **NORMAL**

Specifies that printing is to be done on both sides of the sheet after which the sheets can be bound on the long edge of the paper.

#### **TUMBLE**

Specifies that printing is to be done on both sides of the sheet after which the sheets can be bound on the short edge of the paper.

See [“Specifying duplex printing” on page 117](#) for examples of using the DUPLEX parameter. For more information about duplex printing, see [“Duplex printing” on page 54](#).

## FCB

Specifies the name of the page definition to be used in formatting a print data set. PSF adds the system prefix P1 to the forms control buffer (FCB) name you specify and uses it for the page definition.

You can specify the FCB parameter in a DD statement or in an OUTPUT statement, as follows:

### **FCB=*pdefname***

The value is:

#### ***pdefname***

Specifies 1–4 alphanumeric characters for the page definition name. Do not use a P1 prefix.

When you are using the FCB parameter, remember:

- If you specify the parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement.
- If both an FCB parameter and a PAGEDEF parameter are coded in your JCL, PSF ignores the FCB parameter. See [“PAGEDEF” on page 100](#).
- If you are using Line Mode Migration and an FCB parameter and a FORMLEN parameter are coded in your JCL, PSF uses the FORMLEN value and ignores the FCB parameter. See [“FORMLEN” on page 95](#). For more information about Line Mode Migration, see *PSF for z/OS: Customization*.
- Because the maximum length of a PAGEDEF parameter is 6 characters and the maximum length of an FCB parameter is 4 characters, you cannot use the FCB parameter to specify all page definitions. IBM recommends that you use the PAGEDEF parameter when you do not want any changes in the JCL and you are not converting jobs from line printers.

See [“Specifying a page definition” on page 121](#) for examples of using the FCB parameter to specify page definitions.

## FLASH

Specifies the name of the forms flash to be printed, and the number of copies on which the flash is to be printed. A forms flash is a 3800 printer hardware frame that prints a photographic negative on selected forms.

You can specify the FLASH parameter in a DD statement or in an OUTPUT statement, as follows:

### **FLASH=(*flashname*,*count*)**

The values are:

***flashname***

Specifies the 1–4 alphanumeric character name of the forms flash.

***count***

Specifies the number of copies (1–255) on which the forms flash is to be printed. The default is 255; therefore, if this value is not specified, the forms flash is printed on all copies.

When you specify a forms flash, remember:

- If you specify the FLASH parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement.
- A FLASH parameter that is specified in your JCL statement overrides any FLASH specified in the **Form definition** parameter in the Printer Inventory or the FORMDEF parameter in the PSF PRINTDEV statement. The PRINTDEV statement, which is set up by your system programmer, contains defaults for PSF to use for any parameters that are not specified in the JCL.
- If you specify a forms flash within the active copy-group definition, but do not specify a name with the FLASH parameter, PSF uses the forms-flash frame that is loaded.
- If you specify both the FORMDEF and FLASH parameters in your JCL, but the form definition does not specify FLASH, PSF ignores the FLASH parameter because the FORMDEF parameter overrides it. The forms flash is not printed on any of the copies of your data set. The effects of specifying FORMDEF and FLASH are described in [“Using FORMDEF with COPIES or FLASH parameters in JCL” on page 114](#).
- If you specify the count value in the JCL FLASH parameter, and PSF uses the default form definition because you did not specify the FORMDEF parameter, the modifications that are specified in the default form definition subgroup (number of copies, flash, overlay identifiers, and suppression identifiers) are not used. A form definition subgroup is described in [“Subgroup modifications” on page 61](#).
- The FLASH parameter is valid only for the IBM 3800 printer; PSF ignores FLASH for other printers. However, you can use overlays on any printer. Overlays are described in [“Overlays” on page 30](#).
- If you are using the "Page-Printer Defaults" form, it indicates whether the default form definition activates forms flash (see Appendix D, [“Page-printer defaults form,” on page 221](#)).

See [“Using a forms flash on a 3800 printer” on page 118](#) for examples of using the FLASH parameter to specify a forms flash.

## FORMDEF

Specifies the member name of the form definition you want to use. PSF adds the system prefix F1 to the member name you specify and uses it for the form definition.

A *form definition* specifies how a page of data is placed on a form, the number of copies of a page, any modifications to that group of copies, the paper source, and whether duplex printing is needed. For more information about the contents of a form definition, see [“Form definitions” on page 52](#) and [Appendix A, “Form definitions supplied with PSF,” on page 175](#).

You can specify the FORMDEF parameter in an OUTPUT statement, as follows:

**FORMDEF=fdefname**

The value is:

***fdefname***

Specifies one to six alphanumeric or national characters for the form definition member name. Do not use an F1 prefix.

When you specify a form definition, remember:

- If the FORMDEF parameter is not specified, PSF uses the default form definition specified in the PSF startup procedure or the Printer Inventory. If you are using the "Page-Printer Defaults" form, this default is contained in the form. For a copy of this form, see [Appendix D, “Page-printer defaults form,” on page 221](#).
- When the job uses direct-printing mode, you can specify the FORMDEF parameters in an OUTPUT statement or in a PRINTDEV JCL statement. The PRINTDEV statement is described in [PSF for z/OS: Customization](#).



The form definition must be stored in a PSF system library, in a PSF user library, or coded inline as part of the print data set. You can create your own form definition or use a resource already available on your system. See [“Specifying a form definition” on page 112](#) for examples of using the FORMDEF parameter, including what you need to do to use form definitions inline or from a user library.

## FORMLEN

Specifies the paper length in inches or in centimeters. This parameter is used to change the paper length at the printer (physical paper) without reconfiguring the printer.

You can specify the FORMLEN parameter in an OUTPUT statement, as follows:

**FORMLEN=xx.yyyIN | xx.yyyCM**

The values are:

**xx.yyyIN**

Specifies up to 2 digits before the decimal point and up to 3 digits following the decimal point to indicate paper length in inches.

**xx.yyyCM**

Specifies up to 2 digits before the decimal point and up to 3 digits following the decimal point to indicate paper length in centimeters.

If the FORMLEN parameter is not specified, the printer's default paper length is used. If the printer does not support the Set Media Size command, the FORMLEN parameter is ignored.

The FORMLEN parameter is only specified in the user's OUTPUT JCL statement. If the FORMLEN parameter is specified on an OUTPUT statement in the PSF startup procedure, it is ignored. The FORMLEN parameter value that is specified in the user's OUTPUT JCL statement is used for all pages in the data set including the job header, job trailer, data set header, message data set, and interrupt message page.

The FORMLEN parameter is not used for JESNEWS pages. JES creates a separate data set for JESNEWS that does not have the JCL values for the user data set. JES2 puts the JESNEWS on the job header page and the form length value comes from the FORMDEF for the job header. JES3 puts the JESNEWS on the job trailer page and the form length value comes from the FORMDEF for the job trailer.

If you are using Line Mode Migration and a FORMLEN parameter and an FCB parameter are coded in your JCL, PSF uses the FORMLEN value and ignores the FCB parameter. See [“FCB” on page 93](#). For more information about Line Mode Migration, see [PSF for z/OS: Customization](#).

If you are using Download for z/OS or AFP Download Plus to send jobs specifying FORMLEN, be aware of the following:

- Download for z/OS converts this value to millimeters without specifying a unit when sending it to the receiver.
- AFP Download Plus sends the form length exactly as specified on the OUTPUT JCL statement by default. If this causes an error at the receiver, inform your system programmer to configure AFP Download Plus to match the way Download for z/OS sends this parameter. For more information about the **match-download-format** AFPPARMS parameter, see [PSF for z/OS: AFP Download Plus](#).

**Note:** PSF uses the XOH Set Media Size (SMS) command to set the printer's physical printable area in the length direction by using the FORMLEN value. Although some printers use the SMS values unconditionally, some do not. See your printer hardware documentation to determine how SMS is used by the printer.

**Example:** In this example, a paper length of 9.5 inches is specified in the OUTPUT statement:

```
//OUTPUT1 OUTPUT FORMLEN=9.5IN
//DD1 DD SYSOUT=A,OUTPUT=(*.OUTPUT1)
```

## FORMS

Specifies the name of the form that the print operator is notified to load. For channel-attached printers, the printer operator receives a message to load this form. For SNA-attached and TCP/IP-attached printers, the **Issue setup messages** parameter in the Printer Inventory or the SETUP parameter on the

PRINTDEV statement can be used to notify the printer operator to load this form. For more information about these parameters, see [PSF for z/OS: Customization](#).

This parameter can also be used to notify the operator that specific Selectable Medium Modifications must be enabled on a post processor. Examples of modifications are applying colored plates, cutting, perforating select pages of output, or using MICR fonts. Consult your system-support group for the paper sizes or Selectable Medium Modifications available on your printer and for the form identification that you use in JCL to specify the paper or Selectable Medium Modifications you want.

To specify the name of a form, either use a SYSOUT parameter on the DD statement or use the FORMS parameter on an OUTPUT statement assigned to that DD statement. You must specify a form in one statement or the other. If you specify a parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement.

**FORMS=***formname*

The value is:

***formname***

Specifies the 1–8 alphanumeric character name of the form.

**Example:** In this example, the form is specified in the OUTPUT statement:

```
//OUTPUT1 OUTPUT FORMS=FORM1
//DD1 DD SYSOUT=A,OUTPUT=(*.OUTPUT1)
```

If you are using the Page-Printer Defaults form (see [Figure 34 on page 82](#)), the form that is typically loaded in the printer is shown.

**Note:** For information about printing to alternative paper sources, see [“Specifying bins \(paper source\)” on page 117](#).

## INTRAY

Specifies the number that identifies the tray from which paper is to be selected. To determine the input tray identifiers for your printer, see the documentation provided with the printer.

You can specify the INTRAY parameter in an OUTPUT statement, as follows:

**INTRAY=***nnn*

The value is:

***nnn***

Specifies the one- to three-digit decimal number 1–255 that identifies the input tray.

If INTRAY is not specified, the value that is specified in the form definition is used. However, if no value is specified in the form definition, the printer's default source is used.

**Example:**

```
//OUTPUT1 OUTPUT INTRAY=4
//DDN DD SYSOUT=P,OUTPUT=(*.OUTPUT1)
```

**Note:** For information about Side and Edge Sensitive Paper Handling, see [PSF for z/OS: Customization](#).

## LINECT

Specifies the maximum number of lines PSF prints on each output page when the FCB parameter is used with line-mode conversion. PSF creates a page break and starts a new page when the number is reached. This parameter value is used if it is smaller than the value specified for the FCB line count.

You can specify the LINECT parameter in an OUTPUT statement, as follows:

**LINECT=***nnn*

The value is:

**nnn**

Specifies the one- to three-digit decimal number 0–255 that indicates the maximum number of lines printed on each page. If LINECT=0, PSF stores the document as one page.

**Example:** In this example, PSF creates a page break and starts a new page after every 45 lines:

```
//PRNTDS OUTPUT LINECT=45
//DD1 DD SYSOUT=P,OUTPUT=(*.PRNTDS)
```

You can use the LINECT parameter only if the Line Mode Migration function is enabled in PSF and Exit 7 requests that the LINECT parameter is used to calculate the number of lines on a page (XTP7LCNT flag is set ON). For more information about Line Mode Migration, see [PSF for z/OS: Customization](#).

## NOTIFY

Specifies that you want PSF to notify up to four users when the printer finishes printing your job. The NOTIFY message advises a user that the job completed, successfully or unsuccessfully, and indicates which output is finished, incomplete, or unprintable. If the printer is attached through DPF, the message that PSF issues does not indicate that printing is complete; it means that PSF finished the processing.

PSF sends the notice when an output group is stacked. An output group can contain multiple data sets.

You can specify the NOTIFY parameter in an OUTPUT statement, as follows:

**NOTIFY=(node.userid1[, node.userid2][,node.userid3][,node.userid4])**

The values are:

**node**

Specifies a 1–8 alphanumeric character node for a system where the notification is to be sent. The node is optional if it is the same as the system that runs the print job.

**userid**

Specifies a 1–8 alphanumeric character user ID of the person who is to receive the notification.

Even if you specified the NOTIFY parameter in your print-job JCL, in these circumstances, you do not receive a message that says that your print job is canceled:

- If a printer operator presses the printer CANCEL key while the first data set of a job that contains multiple data sets is printing, the first data set is canceled, rather than the entire group or job. You do not receive a message that says that your print job is canceled, but you receive a message that says that your job is complete, even though the first data set is not complete. If the operator presses the CANCEL key while the last data set of a group is printing, you receive a message that says that your print job is canceled.
- If a printer operator types a JES CANCEL command that specifies your job number, and the job contains multiple data sets, your job is canceled, but you do not receive a message that says that your output is canceled.

See [“Specifying notification when the print job finishes printing” on page 146](#) for examples of using the NOTIFY parameter.

## OFFSETXB

Specifies the offset in the x direction of the logical page origin from the media origin for the back side of each sheet. If OFFSETXB is not specified, the value that is specified in the form definition is used.

You can specify the OFFSETXB parameter in an OUTPUT statement, as follows:

**OFFSETXB=nnnn[.mmm]unit**

The values are:

**nnnn**

Specifies the one- to four-digit number that indicates the offset for the back side of each sheet.

***mmm***

Specifies the one- to three-digit decimal number that indicates the offset for the back side of each sheet.

***unit***

Specifies one of these measurement units:

**IN**

Inches

**CM**

Centimeters

**MM**

Millimeters

**PELS**

Picture elements (1/240 inch)

**POINTS**

Points (1/72 inch)

**Note:** If you specify the unit as PELS or POINTS, you must specify the value as a whole number with no decimal point.

## OFFSETXF

Specifies the offset in the x direction of the logical page origin from the media origin for the front side of each sheet. If OFFSETXF is not specified, the value that is specified in the form definition is used.

You can specify the OFFSETXF parameter in an OUTPUT statement, as follows:

**OFFSETXF=nnnn[.mmm]unit**

The values are:

***nnnn***

Specifies the one- to four-digit number that indicates the offset for the front side of each sheet.

***mmm***

Specifies the one- to three-digit decimal number that indicates the offset for the front side of each sheet.

***unit***

Specifies one of these measurement units:

**IN**

Inches

**CM**

Centimeters

**MM**

Millimeters

**PELS**

Picture elements (1/240 inch)

**POINTS**

Points (1/72 inch)

**Note:** If you specify the unit as PELS or POINTS, you must specify the value as a whole number with no decimal point.

## OFFSETYB

Specifies the offset in the y direction of the logical page origin from the media origin for the back side of each sheet. If OFFSETYB is not specified, the value that is specified in the form definition is used.

You can specify the OFFSETYB parameter in an OUTPUT statement, as follows:

**OFFSETYB=nnnn[.mmm]unit**

The values are:

**nnnn**

Specifies the one- to four-digit number that indicates the offset for the back side of each sheet.

**mmm**

Specifies the one- to three-digit decimal number that indicates the offset for the back side of each sheet.

**unit**

Specifies one of these measurement units:

**IN**

Inches

**CM**

Centimeters

**MM**

Millimeters

**PELS**

Picture elements (1/240 inch)

**POINTS**

Points (1/72 inch)

**Note:** If you specify the unit as PELS or POINTS, you must specify the value as a whole number with no decimal point.

## OFFSETYF

Specifies the offset in the y direction of the logical page origin from the media origin for the front side of each sheet. If OFFSETYF is not specified, the value that is specified in the form definition is used.

You can specify the OFFSETYF parameter in an OUTPUT statement, as follows:

**OFFSETYF=nnnn[.mmm]unit**

The values are:

**nnnn**

Specifies the one- to four-digit number that indicates the offset for the front side of each sheet.

**mmm**

Specifies the one- to three-digit decimal number that indicates the offset for the front side of each sheet.

**unit**

Specifies one of these measurement units:

**IN**

Inches

**CM**

Centimeters

**MM**

Millimeters

**PELS**

Picture elements (1/240 inch)

**POINTS**

Points (1/72 inch)

**Note:** If you specify the unit as PELS or POINTS, you must specify the value as a whole number with no decimal point.

**Example:** This JCL example sets the page origin to 0.5 inches, 1.1 inches on the front side, and 1.5 inches, 1.1 inches on the back side:

```
//OUT1 OUTPUT OFFSETXF=0.5IN,OFFSETYF=1.1IN,  
//          OFFSETXB=1.5IN,OFFSETYB=1.1IN  
//DDN DD SYSOUT=P,OUTPUT=(*.OUT1)
```

## OUTBIN

Specifies the identifier of the output bin into which PSF places a print job. If the output bin is not specified, the value that is specified in the form definition is used. However, if the printer does not support the selection of an output bin, the job is stacked in the default output bin for the printer.

You can specify the OUTBIN parameter in an OUTPUT statement, as follows:

### **OUTBIN=nnnnn**

The value is:

#### **nnnnn**

Specifies the one- to five-digit decimal number 1–65535 that identifies the output bin into which PSF places a print job.

**Example:** In this example, OUTBIN is specified in the OUTPUT statement:

```
//OUTPUT OUTPUT OUTBIN=4  
//DD2 DD SYSOUT=A,OUTPUT=(*.OUTPUT1)
```

## OVERLAYB

Specifies the member name of a medium overlay to be placed on the back medium presentation space of each sheet, in addition to overlays from other sources.

You can specify the OVERLAYB parameter in an OUTPUT statement, as follows:

### **OVERLAYB=ovlyname**

The value is:

#### **ovlyname**

Specifies one to eight alphanumeric or national characters for the member name of a medium overlay. The complete name of the overlay member must be given; PSF does not add an O1 prefix.

See [“Printing a medium overlay” on page 119](#) for examples of using the OVERLAYB parameter.

## OVERLAYF

Specifies the member name of a medium overlay to be placed on the front side of each sheet, in addition to overlays from other sources.

You can specify the OVERLAYF parameter in an OUTPUT statement, as follows:

### **OVERLAYF=ovlyname**

The value is:

#### **ovlyname**

Specifies one to eight alphanumeric or national characters for the member name of a medium overlay. The complete name of the overlay member must be given; PSF does not add an O1 prefix.

See [“Printing a medium overlay” on page 119](#) for examples of using the OVERLAYF parameter.

## PAGEDEF

Specifies the member name of the page definition you want to use. PSF adds the system prefix P1 to the member name you specify and uses it for the page definition.

A *page definition* defines the page format that PSF uses to compose line data into pages. For more information about the contents of a page definition, see [“Page definitions” on page 63](#) and [Appendix B, “Page definitions supplied with PSF,” on page 185](#).

You can specify the PAGEDEF parameter in an OUTPUT statement, as follows:

**PAGEDEF=*pdefname***

The value is:

***pdefname***

Specifies one to six alphanumeric or national characters for the page definition member name. Do not use a P1 prefix.

When you select a page definition, remember this:

- If the PAGEDEF parameter is not specified, but an FCB parameter is coded in the JCL, PSF uses the page definition named in the FCB parameter. See [“FCB” on page 93](#) for information about coding the FCB parameter.
- If a PAGEDEF parameter or an FCB parameter is not coded in your JCL, PSF uses the system default. If you are using the "Page-Printer Defaults" form, this default is shown on the form. For a blank copy of this form, see [Appendix D, “Page-printer defaults form,” on page 221](#).
- PSF does not support the LINECT parameter on the /\*JOBPARM and /\*OUTPUT statements. The maximum number of lines to be printed on a page can be defined in a page definition.
- When your job uses direct-printing mode, you can specify the PAGEDEF parameter in an OUTPUT statement or in a PRINTDEV JCL statement. The PRINTDEV statement is described in [PSF for z/OS: Customization](#).

The page definition must be stored in a PSF system library, in a PSF user library, or coded inline as part of the print data set. You can create your own page definition or use a resource already available on your system. See [“Specifying a page definition” on page 121](#) for examples of using the PAGEDEF parameter, including what you need to do to use page definitions inline or from a user library.

## PIMSG

Specifies whether you want to print PSF messages, and also specifies the maximum number of errors that can occur before printing is stopped. PSF messages have these characteristics:

- A primary message and possibly other, associated environmental messages are generated for an error.
- A group of messages is generated for an error that stops a process. For example, the messages for an I/O error contain:
  - A message identifier and the I/O error description
  - Additional message identifiers with their descriptions of the environment in which the error occurred
- Message groups are separated with a blank line in the printed output.

You can specify the PIMSG parameter in an OUTPUT statement, as follows:

**PIMSG=YES | NO | (YES,*nnn*) | (NO,*nnn*) | (*nnn*)**

The values are:

**Y or YES**

Specifies that all message groups generated in the processing of a data set are to be printed at the end of the data set, preceding any trailer pages. If a data set is left incomplete because of an error, message groups that are generated up to this error are printed, including the message group that describes the error that stops processing of the data set. YES is the default.

**N or NO**

Specifies that no message groups are to be printed unless an error occurs that forces printing to stop prematurely. If that happens, only the message group that describes the error that stops processing is printed.

### ***nnn***

Specifies that PSF is to stop processing the data set after *nnn* message groups are generated. The default value is 16. The final count of printed messages might be larger than *nnn* if errors are reported for pages that are sent to the printer before the message count is reached. A value of 0 for *nnn* allows printing of the data set to continue regardless of the number of message groups that are generated, unless an error that stops processing occurs.

If PIMSG is not specified, the default is either:

- If a default is defined in the PSF startup procedure, PSF uses it. For the "Page-Printer Defaults" form used to record the system defaults, see [Appendix D, "Page-printer defaults form," on page 221](#).
- If a default is not defined in the PSF startup procedure, PSF uses PIMSG=(YES,16).

See ["Specifying whether you want error messages to be printed" on page 139](#) for examples of using the PIMSG parameter.

## **PRMODE**

Specifies the type of data in the print data set and whether PSF must do optional processing of the data.

You can specify the PRMODE parameter in an OUTPUT statement, as follows:

**PRMODE=SOSI1 | SOSI2 | SOSI3 | SOSI4 | *aaaaaaaa***

The values are:

### **SOSI1**

Specifies that each shift-out, shift-in code is to be converted to a blank and a Set Coded Font Local text control.

### **SOSI2**

Specifies that each shift-out, shift-in code is to be converted to a Set Coded Font Local text control.

### **SOSI3**

Specifies that the shift-in code is to be converted to a Set Coded Font Local text control and two blanks. A shift-out code is to be converted to a Set Coded Font Local text control.

### **SOSI4**

Specifies that each shift-out, shift-in code is to be skipped and not counted when offsets are calculated for the print data set. SOSI4 is used when double-byte character set (DBCS) text is converted from ASCII to EBCDIC. When SOSI4 is specified, the page definition offsets are correct after conversion; therefore, the user does not need to account for SOSI characters when FIELD offsets are computed.

### ***aaaaaaaa***

Specifies any alphanumeric string that is defined in the JES initialization parameters as a selection criterion for a printer to process the data set. LINE and PAGE are examples of PRMODE values that JES uses for job routing information.

See ["Specifying shift-out, shift-in \(SOSI\) codes" on page 135](#) for examples of using the PRMODE parameter. For more information about specifying SOSI codes, including the data conversion that PSF makes for SOSI1, SOSI2, SOSI3, and SOSI4, see ["Shift-out, shift-in \(SOSI\) codes" on page 74](#).

## **PRTERORR**

Specifies the disposition of the print data set to be used if an error occurs during printing for which PSF stops processing the data set.

**Note:** Your system programmer must enable the use of PRTERORR in the PRINTDEV statement for PSF. See [PSF for z/OS: Customization](#).

You can specify the PRTERORR parameter in an OUTPUT statement, as follows:

**PRTERORR=HOLD | QUIT | DEFAULT**

The values are:



## HOLD

Specifies that if an error that stops processing occurs during printing, the data set is held on the JES spool until it is released by the system operator.

## QUIT

Specifies that PSF releases the data set to JES, even if an error that stops processing occurs during printing. Then, JES disposes of the data set as though printing completed successfully. For JES2, the data set is handled according to the OUTDISP value associated with the data set. For JES3, the data set is deleted from the spool.

## DEFAULT

Specifies that the standard PSF action is taken if an error that stops processing occurs during printing. As the default, this option is assumed when no attributes are explicitly specified.

The PRTEROR parameter on the OUTPUT statement is not recognized if any of these conditions exist:

- NO (default) is specified on the **Error disposition supported** parameter in the Printer Inventory
- NOTHONOR (default) is specified on the PRTEROR parameter on the PRINTDEV statement
- Data sets are put on hold by PSF installation exits.
- Errors result in a PSF abend.
- Errors in direct-printing mode.
- Mandatory print labeling (MPL) is active.

## PRTQUEUE

Specifies the name of the target print queue on a remote system when you are using the Download for z/OS and AFP Download Plus features of PSF. For more information about using the PRTQUEUE parameter with these features, see [PSF for z/OS: Download for z/OS](#) and [PSF for z/OS: AFP Download Plus](#).

You can specify the PRTQUEUE parameter in an OUTPUT statement, as follows:

**PRTQUEUE='printqueueName'**

The value is:

**printqueueName**

Specifies a 1- to 127-character print queue name.

**Example:** In this example, the PRTQUEUE parameter specifies a target print queue destination with the name 4019:

```
//OUTQUEUE OUTPUT PRTQUEUE='4019'  
//DDN DD SYSOUT=P,OUTPUT=(*.OUTQUEUE)
```

## RESFMT

Specifies the resolution at which the output was formatted. PSF uses this information to choose the correct resolution system library. Your system programmer must define different resolution system libraries, as described in [PSF for z/OS: Customization](#).

You can specify the RESFMT parameter in an OUTPUT statement, as follows:

**RESFMT=P240 | P300**

The values are:

### P240

Specifies that PSF uses resources (fonts, overlays, and page segments) from the 240-pel resource libraries.

### P300

Specifies that PSF uses resources from the 300-pel resource libraries.

**Example:** This example specifies that the output is formatted at a resolution of 240 pels.

```
//OUT2 OUTPUT RESFMT=P240
//DD2 DD SYSOUT=A,OUTPUT=(*OUT2)
```

See “[Printing on printers that support multiple resolutions](#)” on [page 152](#) for more information about specifying format resolution to PSF and the order that PSF looks for the specification.

## SEGMENT

Specifies that part of the output for a job is to be spooled to print while the job is still running, or specifies that different segments of a job are to be printed simultaneously on different printers. This parameter is optional and can be used only with line data, not with MO:DCA-P data.

You can specify the SEGMENT parameter in a DD statement, as follows:

### **SEGMENT=page-count**

The value is:

#### ***page-count***

Specifies the number of pages to be printed in this segment of the output data set. When the page-count is reached, JES prints that number of pages and begins counting the next segment of pages; it continues counting and printing until the remainder of the data set is printed.

**Example:** This example shows how to print a large data set in segments of 100 pages each. When JES writes 100 pages to an output data set, the segment is scheduled for printing, and JES begins the second segment of 100 pages. The process continues until the entire data set is printed.

```
//DD1 DD SYSOUT=A, SEGMENT=100
```

### **Notes:**

1. SEGMENT is supported only on JES2 systems. You might use SEGMENT when you need to print a large job but do not want to monopolize a single printer for a long time.
2. Using the SEGMENT parameter with conditional processing produces unexpected results because conditional processing is not supported across data set boundaries. The SEGMENT parameter causes a data set to be broken into what JES considers separate data sets, which cause conditional processing to stop at the end of the first segment (data set).

## SUBSYS

Replaces SYSOUT as a way of submitting print jobs. SUBSYS provides a means of spooling jobs through batch that gives you access to functions provided by Infoprint Server so you can take advantage of any available Infoprint Server transforms. To use SUBSYS, replace `SYSOUT=(class,,formname)` with `SUBSYS=(subsystem_name, . . .)` on the DD statement.

To use SUBSYS, you must have a license for the Infoprint Server feature of z/OS. For more information about using SUBSYS, see [z/OS Infoprint Server User's Guide](#).

You can specify the SUBSYS parameter in a DD statement, as follows:

### **SUBSYS=(subsystem\_name, printer\_definition\_name, job\_attributes)**

The values are:

#### ***subsystem\_name***

Specifies the subsystem name. The subsystem name is the name that is used in the PSF startup procedure to identify the Printer Inventory. This parameter is required and can be up to 4 characters long.

#### ***printer\_definition\_name***

Specifies the name of the Infoprint Server printer definition. This value is case-sensitive and needs to be entered exactly as specified in the Infoprint Server Printer Inventory. You must surround the name by single quotation marks if it contains lowercase letters. This parameter is optional but recommended. If not specified, the system default printer definition is used.

### ***job\_attributes***

Specifies optional job attributes. This parameter is a quote-delimited string that contains a set of blank-delimited pairs in the format 'job-attribute=value'. If the attribute value contains blanks or special characters, you must enclose the value in double quotation marks. See [z/OS Infoprint Server User's Guide](#) for a list of supported job attributes.

**Example:** This example shows how to submit output to the subsystem named AOP1. The output is printed with printer definition myprinter, which is set up to start the PostScript to AFP transform. The output from the transform includes pages 1-10 only.

```
//OUT1      OUTPUT COPIES=2
//DD1       DD      SUBSYS=(AOP1,'myprinter','filter-options="-p 1-10"'),
//          OUTPUT=(*.OUT1)
```

## **SYSAREA**

Specifies whether an area on each page of printed output is reserved for the security label. The security label represents a security level and categories as defined to RACF. This parameter is used with the DPAGELBL parameter on the OUTPUT statement and the SECLABEL parameter on the job statement.

**Note:** When a system area is reserved for a security label, the printed output is shifted on each page. You cannot print output data in the reserved area.

You can specify the SYSAREA parameter in an OUTPUT statement, as follows:

### **SYSAREA=YES | NO**

The values are:

#### **Y or YES**

Specifies that the area outside the user printable area is reserved for a security label and any attempt to print in the reserved area results in error messages.

#### **N or NO**

Specifies that the printable area is not restricted.

If your printer does not support guaranteed print labeling, this parameter must be set to NO. For more information about guaranteed print labeling, security labels, and the SECLABEL parameter, see [PSF for z/OS: Security Guide](#).

**Example:** In this example, the security label, which is specified on the SECLABEL parameter of the job statement, is printed in the system area on each page of output printed on forms named CSEC:

```
//JOBA      JOB      1, 'JOHN DOE', SECLABEL=CONF
//PRESRPT   OUTPUT   DPAGELBL=YES, SYSAREA=YES, FORMS=CSEC
//DD1 DD     SYSOUT=A, OUTPUT=(*.PRESRPT)
```

## **SYSOUT**

Specifies the output class and optionally, the form name for the print data set. The output class and the form name can also be specified in the CLASS and FORMS parameters of the OUTPUT statement. For information about using these parameters, see [“CLASS” on page 87](#) and [“FORMS” on page 95](#).

You can submit jobs to the Infoprint Server subsystem for processing and spooling by using the JCL SUBSYS parameter (you must have a license for the Infoprint Server feature of z/OS). For more information about the Print Interface subsystem, see [z/OS Infoprint Server User's Guide](#).

You can specify the SYSOUT parameter in a DD statement, as follows:

### **SYSOUT=(class,,formname)**

The values are:

#### **class**

Specifies one alphanumeric character for the output print class name.

***formname***

Specifies the 1–8 alphanumeric character name of the form.

You must specify a class value on either the CLASS parameter in the OUTPUT statement or the SYSOUT parameter in the DD statement. If the class or the form name is coded both in the OUTPUT statement and in the SYSOUT parameter, PSF uses the value coded in the DD statement.

**Examples:**

1. This example specifies an output class of A. The form name is specified in the OUTPUT statement:

```
//OUT1 OUTPUT FORM=FORM1
//DD1 DD SYSOUT=A,OUTPUT=(*.OUT1)
```

2. In the next example, both class and form are specified with the SYSOUT parameter:

```
//DD2 DD SYSOUT=(A, ,FORM2)
```

3. To allow class to be specified in an OUTPUT statement, code the SYSOUT parameter as in this example:

```
//OUT3 OUTPUT CLASS=A
//DD3 DD SYSOUT=(, ),OUTPUT=(*.OUT3)
```

## TRC

Specifies whether the print data set contains table reference characters (TRCs).

In traditional line data, you can use different fonts on different lines of a file by specifying TRCs at the beginning of each line after the carriage control characters, if any are present. This parameter cannot be used with record format line data.

To specify a TRC, either use DCB=OPTCD=J on the DD statement, or use the TRC parameter on an OUTPUT statement assigned to that DD statement. You must specify a TRC in one statement or the other. If you specify a parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement.

**TRC=YES | NO**

The values are:

**Y or YES**

Specifies that the print data set contains TRCs.

**N or NO**

Specifies that the print data set does not contain TRCs. NO is the default.

See [“Using table reference characters to select fonts” on page 133](#) for examples of using the TRC parameter and guidelines for using TRCs.

## UCS

Specifies the member name of the FOCA coded font that you want to use to print a data set. When a CHARS parameter is not specified in the output JCL, you can specify the universal character set (UCS) parameter to identify one font.

You can specify the UCS parameter in a DD statement or in an OUTPUT statement, as follows:

**UCS=*fontname***

The value is:

***fontname***

Specifies the 1–4 alphanumeric character name of a FOCA coded font (in a font library). When you use UCS to specify the member name, do not include the 2-character prefix of the coded-font name (X0 through XG).

**Example:** This example shows the UCS parameter in a DD statement:

```
//DD1 DD SYSOUT=A,UCS=60DB
```

When you specify the UCS parameter, remember:

- If you specify the parameter in both a DD statement and an OUTPUT statement, PSF uses the parameter in the DD statement.
- If you specify a page definition on the OUTPUT statement that specifies fonts for your data set, the UCS parameter is ignored. For more information, see [“Fonts” on page 67](#).
- The UCS parameter is never used in deferred-printing mode under JES3.

## USERLIB

Specifies the name, up to 44 characters, of 1–8 cataloged MVS™ data sets (user libraries) containing AFP resources for processing the print data set.

PSF dynamically allocates the data sets and searches for resources in them in the order specified on the USERLIB statement. If PSF finds no resources, PSF searches the system libraries defined in the startup procedure. The libraries that you specify can contain any of these AFP resources: FOCA fonts, page segments, overlays, page definitions, form definitions, or object container resources. If RACF is installed on your system, RACF checks the authority of the user ID requesting access to a user library.

You can specify the USERLIB parameter in an OUTPUT statement, as follows:

**USERLIB=('libname1','libname2',... 'libname8')**

The value is:

### **libname**

Specifies the name of resource user libraries, such as  
USERLIB= ( 'USERA.RESOURCE' , 'USERA.FONT' ).

When you specify the USERLIB parameter, remember:

- For user data sets, PSF ensures that a USERLIB resource is used if USERLIB is specified, even if a resource (not from a USERLIB) with the same name is already loaded in the printer or in virtual storage.
- An inline resource overrides a resource of the same name contained in a USERLIB parameter.
- If you are printing on a printer that is driven by a distributed print function product or are using a printer that supports resident fonts, you can use a DPF-resident resource or a printer-resident font instead of your USERLIB resource. For more information, see [“Using printer-resident fonts” on page 24](#).
- As soon as PSF finishes processing a data set, PSF deletes all the USERLIB resources from the printer or from memory. PSF deallocates USERLIB data sets at data set end to make them unavailable to another user.

Also, remember these constraints to dynamic allocation:

- Any libraries that are specified with the USERLIB parameter must be accessible on all systems on which output can be printed.
- The USERLIB parameter is not supported for direct-printing mode.

See [“Printing with resources from a user library” on page 142](#) for examples of using the USERLIB parameter and for other guidelines.

## USERPATH

Specifies the name, up to 255 characters, of one to eight UNIX file resource path libraries that contain extended code pages or any resource installed with a resource access table (RAT), such as TrueType and OpenType fonts and data object resources.

Extended coded pages can be installed in path libraries or PDS or PDSE libraries (see [“Using extended code pages” on page 141](#)). RAT-installed resources must be installed in the path libraries that are specified with the USERPATH parameter (see [“Using TrueType and OpenType fonts” on page 140](#) for

information about installing fonts or [“Specifying object container libraries in UNIX files” on page 150](#) for information about installing data objects).

PSF uses z/OS UNIX System Services to access the resources in the user path libraries in the order specified on the USERPATH parameter. If PSF does not find any resources in the user path libraries, it searches the system path libraries; for example, libraries defined on the FONTPATH parameter or the OBJCPATH parameter in the PSF startup procedure.

You can specify the USERPATH parameter in an OUTPUT statement, as follows:

**USERPATH=('libpath1','libpath2',...*libpath8*)**

The value is:

***libpath***

Specifies the name of UNIX file resource path libraries, such as USERPATH=(' /jdoe/fonts/truetype', '/jdoe/fonts/truetype/myfonts/').

**Notes:**

1. Any path libraries that are specified with the USERPATH parameter must contain UNIX files that are accessible with z/OS UNIX System Services and PSF. UNIX files are those that are in a Hierarchical File System (HFS) or the z/OS File System (zFS).
2. Make sure that user access permissions are set for any path libraries specified with the USERPATH parameter. Otherwise, PSF cannot obtain the resources from the UNIX files. See [“Using TrueType and OpenType fonts” on page 140](#) for information about setting permissions.
3. The USERPATH parameter is not supported for direct-printing mode.

See [“Printing with resources from a user library” on page 142](#) for examples of using the USERPATH parameter and for other guidelines.

## Additional parameters to help in distributing output

These user JCL parameters add information to the separator pages printed with an output data set. A sample separator page that uses these parameters is shown in [Figure 36 on page 109](#).



2. This example prints "57 Fair Lane, Omaha, NE 12121" on the separator pages of each data set that references OUTDS3. The first line that is reserved for the addressee's name is blank on the separator page. The ZIP code does not require apostrophes because it contains only characters that are valid without apostrophes:

```
//OUTDS3 OUTPUT ADDRESS=(,'57 Fair Lane','Omaha, NE',12121)
```

The output is printed as follows:

```
57 Fair Lane  
Omaha, NE  
12121
```

### **BUILDING=building**

Specifies the name of a building, containing 1–60 characters, on the separator pages of the output data set. This parameter can help in distributing output and is optional.

**Example:** This example prints "920" on the line reserved for BUILDING on the separator pages of any output data set that references OUTDS4:

```
//OUTDS4 OUTPUT BUILDING='920'
```

### **DEPT=department**

Specifies a department name, containing 1–60 characters, that is associated with the output data set. This parameter can help in distributing output and is optional.

**Example:** This example prints "PAYROLL" on the line reserved for DEPT on the separator pages of any output data set that references OUTDS5:

```
//OUTDS5 OUTPUT DEPT='PAYROLL'
```

### **NAME=name**

Specifies a name, containing 1–60 characters, that is associated with the output data set. This parameter can help in distributing output and is optional.

**Example:** This example prints "R. ROPER" on the line reserved for NAME on the separator pages of any output data set that references OUTDS6:

```
//OUTDS6 OUTPUT NAME='R. ROPER'
```

**Note:** When you are sending output to a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

### **ROOM=room**

Specifies the name, containing 1–60 characters, of the room to be associated with the output data set, which can help in distributing output. This parameter is optional.

**Example:** This example prints "CONFERENCE ROOM" on the line reserved for ROOM on the separator pages of any output data set that references OUTDS7:

```
//OUTDS7 OUTPUT ROOM='CONFERENCE ROOM'
```

**Note:** When you are sending output to a microfilm device, see [Appendix E, “Microfilm device considerations,”](#) on page 223.

### **TITLE=title**

Specifies a description, containing 1–60 characters, of the output data set. This parameter can help in distributing output and is optional.

**Example:** This example prints "ANNUAL REPORT" on the line reserved for TITLE on the separator pages of any output data set that references OUTDS8:

```
//OUTDS8 OUTPUT TITLE='ANNUAL REPORT'
```



---

## Chapter 7. Printing tasks and examples

This information shows detailed examples of tasks that require PSF to print data sets on AFP printers. If you need detailed reference information about using the JCL commands, see [Chapter 6, “Using JCL for Advanced Function Presentation,” on page 81](#).

These printing tasks are described:

- [“Printing on an AFP printer” on page 111](#)
- [“Specifying a form definition” on page 112](#)
- [“Specifying duplex printing” on page 117](#)
- [“Specifying bins \(paper source\)” on page 117](#)
- [“Using a forms flash on a 3800 printer” on page 118](#)
- [“Printing with overlays” on page 118](#)
- [“Printing line data or XML data with page definition options” on page 121](#)
- [“Using multiple copy groups or page formats” on page 128](#)
- [“Printing page segments” on page 130](#)
- [“Printing MO:DCA-P data” on page 131](#)
- [“Specifying carriage control and table reference characters in line data” on page 132](#)
- [“Merging data lines into a single print line” on page 135](#)
- [“Specifying shift-out, shift-in \(SOSI\) codes” on page 135](#)
- [“Printing more than one copy” on page 137](#)
- [“Bursting and stacking continuous-forms paper” on page 139](#)
- [“Specifying whether you want error messages to be printed” on page 139](#)
- [“Using TrueType and OpenType fonts” on page 140](#)
- [“Processing Unicode Complex Text” on page 141](#)
- [“Using extended code pages” on page 141](#)
- [“Printing with resources from a user library” on page 142](#)
- [“Printing with inline resources” on page 143](#)
- [“Specifying notification when the print job finishes printing” on page 146](#)
- [“Inhibiting recovery of a print job” on page 146](#)
- [“Specifying duplex-page offset” on page 147](#)
- [“Transmitting a data set to an IBM i System” on page 147](#)
- [“Specifying JCL parameters for microfilm jobs” on page 148](#)
- [“Specifying color mapping tables” on page 149](#)
- [“Specifying object container libraries in UNIX files” on page 150](#)
- [“Finishing your output” on page 151](#)
- [“Printing on printers that support multiple resolutions” on page 152](#)
- [“Specifying printer checkpoints” on page 153](#)

---

### Printing on an AFP printer

To print on an AFP printer, you send your z/OS print job to the AFP printer that your system programmer defined to PSF. For most jobs, you can use the same JCL that you would use for non-AFP printers, changing only the MVS routing information to direct your job to the correct printer. If you do not require

special AFP functions, you do not need to specify any AFP options for your print job; you can use the PSF defaults defined for your printer.

The examples in this information show how to select a printer by specifying the JCL class and destination parameters. You specify an output class either with a SYSOUT parameter in the DD statement for the data set, or with a CLASS parameter on an OUTPUT statement assigned to that DD statement (see [“CLASS” on page 87](#)). You must specify a class in one statement or the other.

In the first example, a single printer is defined for an output class; therefore, the CLASS parameter is all that you need to specify. However, if an output class contains a group of printers, you also need to include a destination (DEST) parameter to select a specific printer from the group, as in the second example. You can specify DEST either in a DD statement or in an OUTPUT statement. If you do not specify the DEST parameter, and the output class contains more than one printer, the system selects the first available printer in the group.

#### Examples:

1. This example shows how you use only a SYSOUT parameter in the DD statement to select a printer with print class defined as A.

```
//AFPUSERA JOB ...  
//STEP1 EXEC PGM=USERA  
//DD1 DD SYSOUT=A
```

2. This example shows how you select a specific 3820 printer from a group of printers, all of which are defined as CLASS B. You want the 3820 printer with destination name REMOTE1. For this example, the DEST option is coded on the OUTPUT statement.

```
//AFPUSERB JOB ...  
//STEP1 EXEC PGM=USERB  
//OUT2 OUTPUT DEST=REMOTE1  
//DD2 DD SYSOUT=B,OUT=(*.OUT2)
```

In both examples, the job prints according to the AFP defaults that PSF specified for the printer. For jobs that require special AFP options, see the examples in [“Specifying a form definition” on page 112](#).

## Specifying a form definition

A form definition is a resource that defines numerous parameters, such as:

- The placement of a page of data on a form
- The number of copies of a page
- Any modifications to that copy group
- Whether you want a copy group stacked offset from the preceding group
- Whether you want any data fields suppressed (not printed)

You can create your own form definition or use a form definition already available on your system. To create a form definition, use an AFP utility product such as IBM Page Printer Formatting Aid (PPFA). For instructions on how to create a form definition that specifies the options that you require, see the publications for your utility product.

Using the forms that are completed with your installation's defaults, read the descriptions of the form definitions available to you. Also, see Appendix A, “Form definitions supplied with PSF,” on page 175, for descriptions of form definitions provided with PSF. Based on this information, you can decide which form definition best meets your present printing requirements.

If your job requires special form-definition options, use the JCL for the print job to specify a form definition that contains those options. The name of the form definition, without its two-character prefix of “F1”, is coded in the FORMDEF parameter of the OUTPUT statement (see [“FORMDEF” on page 94](#)).

**Examples:** PSF provides standard form definitions, which are listed in Appendix A, “Form definitions supplied with PSF,” on page 175. The form definition that is provided for the 3800 printer offsets the page down 0.5 inch to avoid the unprintable area of that printer. Form definitions for other AFP printers

offset the page 0.167 inch across and down from the medium origin and provide different combinations of duplex and paper source options. These form definitions are referenced in these examples:

1. This example specifies a PSF-supplied form definition that is named F1A10111 to specify duplex printing on a cut-sheet printer. Notice that the prefix "F1" is not coded in the JCL.

```
//AFPUSERA JOB ...  
//STEP1 EXEC PGM=USERA  
//OUT1 OUTPUT FORMDEF=A10111  
//DD1 SYSOUT=A,OUT=(*.OUT1)
```

2. This example specifies a user-created form definition named F1MYFDEF and a page definition named P1MYPDEF. For more information about specifying page definitions, see [“Specifying a page definition” on page 121](#).

```
//AFPUSERB JOB ...  
//STEP1 EXEC PGM=USERB  
//OUT1 OUTPUT FORMDEF=MYFDEF,PAGEDEF=MYPDEF  
//DD1 SYSOUT=A,OUT=(*.OUT1)
```

The FORMDEF parameter can specify the resource name of the inline form definition or can specify the keyword DUMMY. If the name in the FORMDEF parameter does not match the name of an inline form definition, PSF uses the resource from the resource library that matches the name in the JCL. If the job does not specify the FORMDEF parameter, PSF uses the first inline form definition in the print data set.

3. In this example, form definition F1MYFDEF is coded inline in the print data set that is generated by program USERC. For information about coding inline resources, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

```
//AFPUSERC JOB ...  
//STEP1 EXEC PGM=USERC  
//OUT1 OUTPUT FORMDEF=DUMMY  
//DD1 SYSOUT=A,OUT=(*.OUT1)
```

The form definition that is specified on the FORMDEF parameter must be stored in one of these places:

- In a PSF system library
- In a user library specified by the USERLIB parameter
- Inline in the print data set (except in XML data)

## Using form definitions from a user library

You can instruct PSF to select a form definition from your user library rather than from a system library assigned to PSF. To use a form definition from a user library:

1. Reference the user library that contains the form definition in your JCL. For details, see [“USERLIB” on page 107](#).
2. Specify the name of the form definition in the JCL FORMDEF parameter.

To use a form definition from a PSF user library, see [“Printing with resources from a user library” on page 142](#).

## Using inline form definitions

To use an inline form definition:

1. Include the inline form definition in the print data set.
2. If you specify the FORMDEF parameter in your JCL, ensure that the name of the inline form definition matches the form definition name that is specified in your JCL, or else specify FORMDEF=DUMMY in the JCL.
3. If a form definition resource is included inline with the data, ensure that the data set is identified as containing carriage control characters. If the length of the records in the form definition is less than or equal to the logical-record length defined for the data set, you can specify fixed-length records

for the record format. If the length of the records in the form definition is greater than the logical-record length defined for the data set, you must specify variable-length records as variable-blocked with ANSI carriage control characters (VBA) or as variable-blocked with machine carriage control characters (VBM) for the record format.

**Notes:**

1. If you specify FORMDEF=DUMMY in your JCL, and you do not include an inline form definition, PSF uses the default form definition for your printer.
2. If you specify multiple inline form definitions in the print data set and you specify FORMDEF=DUMMY on the OUTPUT statement, PSF uses the last inline form definition in the print data set.
3. If you do not specify the FORMDEF parameter in your JCL, PSF selects the first inline form definition in the print data set.
4. You cannot use inline resources in XML data.

You can include more than one inline form definition in a print data set, and you can change the form definition name in the JCL for different printing jobs to test different form definitions. If the name of an inline form definition does not match the FORMDEF name that is specified in the JCL, PSF uses the form definition from the resource library that matches the name in the JCL. For more information about using inline form definitions, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

## Using FORMDEF with COPIES or FLASH parameters in JCL

If you specify a form definition by coding the FORMDEF parameter in the OUTPUT JCL statement, the number of copies and whether a forms flash is used is determined by the specifications in the subgroups of a copy group. If the JCL COPIES parameter is also specified, it is used only to determine how many times the data set is to be transmitted. If the JCL FLASH parameter is specified, PSF ignores it.

### Specifying COPIES with the FORMDEF parameter

The following examples show the effects on the number of copies that are printed when the JCL COPIES parameter is coded and the number of copies is specified in the form definition selected by FORMDEF. [“COPIES” on page 89](#) shows how to code the JCL COPIES parameter.

1. In this example, assume that you have a two-page print job that uses a form definition that includes a copy group that contains two subgroups. The first subgroup calls for two copies with a particular set of modifications, and the second subgroup calls for one copy with another set of modifications. In the JCL, you would specify COPIES=3. The data set is transmitted to the printer three times, and you get three collated copies of the entire job. The printed output would be as shown in [Figure 37 on page 115](#). Nine copies of page one are printed, and nine copies of page two are printed.

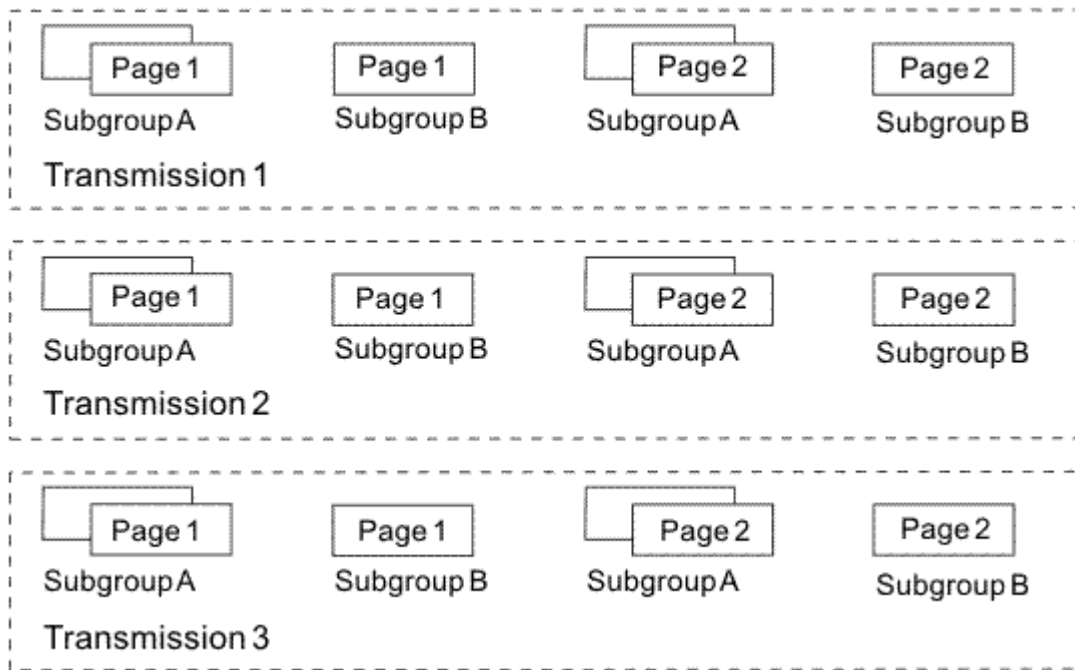


Figure 37. Output from three transmissions of a two-page data set

2. In this example, assume that you use the same form definition as in the first example: the first subgroup specifies two copies, and the second subgroup specifies one copy. In the JCL, specify `COPIES=(, (3,1,5,2))`.

The two-page data set is sent to the printer four times because four groupvalues (3,1,5,2) are specified (see “groupvalue” on page 89). However, those group values have no effect on the number of copies that are printed because that number is determined by the form definition that is specified in the JCL `FORMDEF` parameter; the number of copies that are printed would be the same if you used an entirely different set of group values. When the data set is printed, 12 copies of page one are printed, and 12 copies of page two are printed, as [Figure 38 on page 116](#) shows.

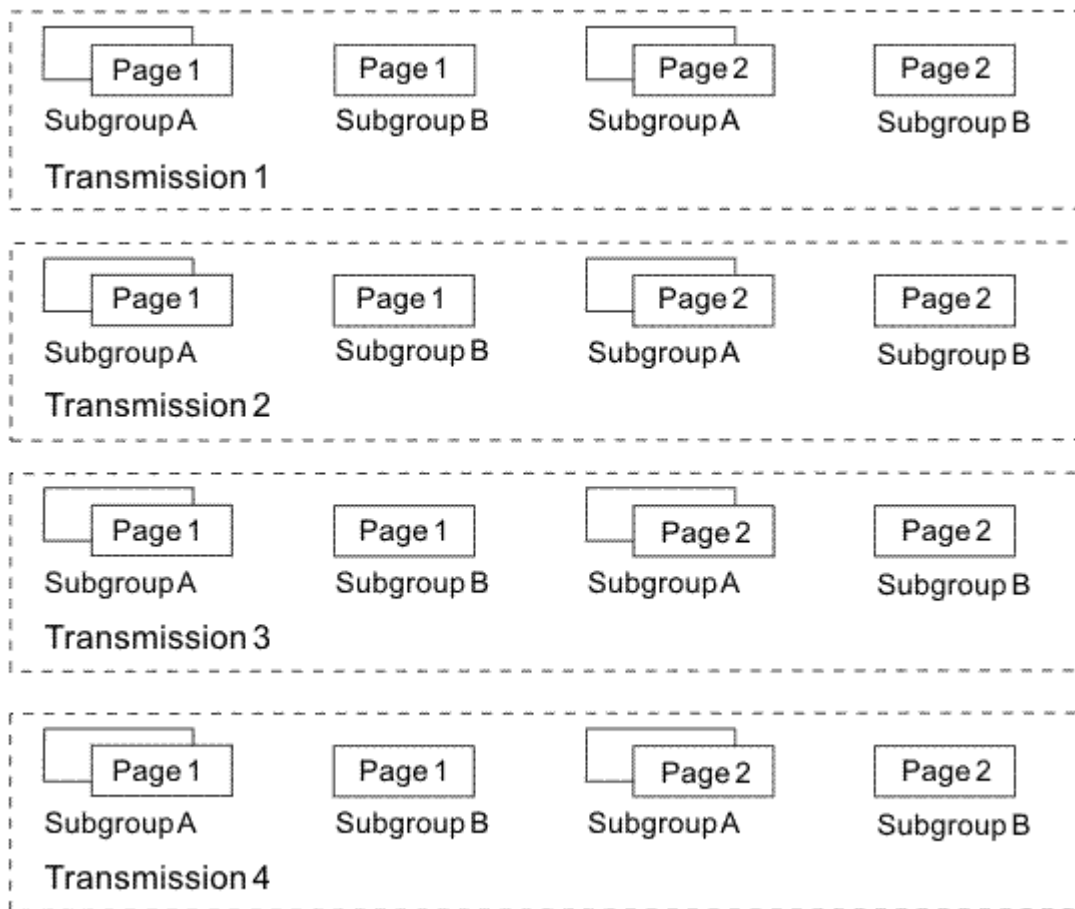


Figure 38. Output from four transmissions of a two-page data set

## Specifying COPIES without the FORMDEF parameter

If PSF uses the default form definition because you did not code the FORMDEF parameter, you can use the JCL COPIES parameter to specify the number of copies to be printed. The system interprets the COPIES parameter variables, *nnn* and *groupvalue*, as described in [“groupvalue” on page 89](#).

If you specify group values in the COPIES parameter, PSF uses only these modifications in the default form definition's subgroup:

- Maximum horizontal adjustment
- Offset stacking
- Edge marking

PSF ignores other modifications. If you specify group values in your JCL, the values override the copy group in the default form definition. A default form definition that is modified by the group value variable in the COPIES parameter is classified as a modified-default form definition. If you do not specify the COPIES parameter or the COPYCNT parameter, and no copy group values are specified in the active copy group in the form definition, PSF prints a single copy.

## Specifying FLASH without the FORMDEF parameter

If PSF uses the default form definition because you did not code the FORMDEF parameter, you can use the JCL FLASH parameter to specify whether the forms flash unit on your 3800 printer is to be used for your print job. The default form definition that is modified by the FLASH parameter is classified as a modified-default form definition. For more information about the FLASH parameter, see [“Using a forms flash on a 3800 printer” on page 118](#).

PSF uses only these modifications in the default form definition subgroup:

- Maximum horizontal adjustment
- Offset stacking
- Edge marking

PSF ignores other modifications. If you omit the FLASH parameter, PSF uses the default form definition without modification.

## Specifying duplex printing

If your AFP printer is capable of printing in duplex mode (printing on both sides of the paper), use the DUPLEX parameter in the OUTPUT statement (see [“DUPLEX” on page 93](#)) or the form definition (see [“Specifying a form definition” on page 112](#)) to control duplexing for your print job.

Printing in duplex saves paper and the space required for storing blank forms and printed documents. You can compound the benefits of multiple-up or N\_UP printing by printing more than one page of application data on one side of a sheet of paper. For more information, see [“Specifying multiple-up printing” on page 124](#).

### Examples:

1. In this example, DUPLEX is defined in the OUTPUT statement:

```
//OUT1 OUTPUT DUPLEX=NORMAL
//DDN DD SYSOUT=P,OUTPUT=(*.OUT1)
```

2. The job in this example uses PSF-supplied form definition F1A10112 to print tumble duplex on paper selected from the primary paper source:

```
//DUPLEX JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT FORMDEF=A10112
//DD1 DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

Form definition F1A10112 is described in [Appendix A, “Form definitions supplied with PSF,” on page 175](#).

If your output requires different page margins on the front and back of the form for binding, you can create a form definition that specifies different page offsets for the front and back of the page (see [“Duplex-page offsets” on page 56](#)). For more information about duplex printing, see [“Duplex printing” on page 54](#).

## Specifying bins (paper source)

Some AFP printers have multiple bins (paper sources) from which you can select the paper for printing your job. To control selecting the paper source, use the form definition. For more information, see [“Specifying a form definition” on page 112](#).

**Example:** This job uses form definition F1A10120, supplied with PSF, to select paper from the alternative paper source.

```
//ALTBIN JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT FORMDEF=A10120
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

Form definition F1A10120 is described in [“Form definitions for printers other than the 3800, PCL4, and PPDS printers” on page 176](#).

**Note:** For information about Side and Edge Sensitive Paper Handling, see [PSF for z/OS: Customization](#).

## Changing the paper source in a document

In the example in [“Specifying bins \(paper source\)”](#) on page 117, all the pages of the document are printed on paper from the same source. You might prefer to print your document on white paper from the main paper source and use sheets of colored paper from the alternative source as internal dividers.

To do this, use a form definition that contains multiple copy groups and tell PSF when to use each copy group. For an example of using a form definition that contains multiple copy groups, see [“Using multiple copy groups or page formats”](#) on page 128.

## Using a forms flash on a 3800 printer

A forms flash is a 3800 hardware frame that prints a photographic negative on selected forms. You can use the FLASH parameter (see [“FLASH”](#) on page 93) to specify whether the 3800 prints a forms flash. The number of copies to be flashed might be different than the number of copies requested. For example, if the number of copies to be flashed is smaller than the total number of copies that are requested, the overlay is printed only on the number of copies specified by the FLASH parameter, beginning with the first copy. The following examples show how to use a forms flash on a 3800 printer.

### Examples:

1. In this example, the FLASH parameter is specified in the DD statement and requests that a forms flash named LOGO is printed on 15 copies. Because the total copies requested is only 10, the LOGO forms flash is printed on each copy.

```
//DD1 DD SYSOUT=A,FLASH=(LOGO,15),COPIES=10
```

2. In this example, 15 copies are requested, but the FLASH parameter specifies only 10 copies in the OUTPUT statement; therefore, the forms flash is only printed on the first 10 copies.

```
//OUT1 OUTPUT FLASH=(LOGO,10)
//DD1 DD SYSOUT=A,COPIES=15,OUTPUT=*.OUT1
```

3. In this example, the JCL specifies form definition F1UFLASH, a user-created form definition that specifies that a forms flash is used. The FLASH parameter specifies that the forms flash named FL001 is to be used on the first two transmissions of the data set.

```
//OUT2 OUTPUT FLASH=(FL001,2),FORMDEF=UFLASH
//DD2 DD SYSOUT=A,COPIES=3,OUTPUT=*.OUT2
```

If the JCL specifies a form definition that does not specify a forms flash, such as the IBM-supplied form definition F1A10110, PSF ignores the FLASH parameter.

## Printing with overlays

Electronic overlays can be called by the form definition. In addition, with PSF, you can call an overlay by using the OVERLAY subcommand in PPFA. You can name and position overlays on the PRINTLINE, LAYOUT, or XLAYOUT command, so that at print time, PSF can access them and merge them with the variable data. Doing this eliminates the need to code Include Page Overlay (IPO) structured field records in the print application.

A *medium overlay* is an overlay called by a form definition; a *page overlay* is an overlay called by a page definition or an IPO structured field. A medium overlay or page overlay can be identified as a *preprinted form overlay* in a form definition. For more information, see [“Overlays”](#) on page 30.

You can use a medium overlay to do jobs such as:

- Print the same overlay on every page of a data set.
- Print different overlays on different copies of the same page.
- Print different overlays on front and back of a duplexed data set.
- Print an overlay on either the back or the front of a duplexed form.



- Print an overlay on the back of a duplexed form, without printing application data on the back. For more information, see [“Constant forms” on page 57](#).
- Print different overlays on different pages of a document by using medium overlays defined in multiple copy groups. For an example of using multiple copy groups, see [“Changing formatting within a document” on page 128](#).
- Print as a preprinted form overlay to simulate preprinted forms or colored paper.

For an example of using a medium overlay, see [“Printing a medium overlay” on page 119](#).

You can use a page overlay to do jobs such as:

- Print different overlays on different pages of the same job.
- Print the same overlay at different positions on different pages of the data set.
- In multiple-up applications, use different arrangements of overlays on the "subpages" of a physical form.
- Print as a preprinted form overlay to simulate preprinted forms or colored paper.

For an example of using a page overlay, see [“Printing a page overlay” on page 119](#).

## Printing a medium overlay

A medium overlay can be called on the JCL OUTPUT statement or in the form definition. To print a medium overlay:

1. Create an overlay resource. You can create an overlay by using an AFP utility such as IBM Overlay Generation Language (OGL). For more information, see *Overlay Generation Language/370 User's Guide and Reference*.
2. Use the overlay by one of these methods:
  - Specify the overlay name on the OVERLAYF parameter, the OVERLAYB parameter, or both parameters on the JCL OUTPUT STATEMENT.
  - Use a form definition that names that overlay for printing. You can build the form definition by using an AFP utility such as IBM Page Printer Formatting Aid (PPFA). For more information about building form definitions, see *Page Printer Formatting Aid: User's Guide*.
3. To identify the medium overlay as a preprinted form overlay, create a form definition with a medium preprinted form overlay local ID keyword (X'D2') on the MMC structured field. You can build the form definition with an AFP utility that supports preprinted form overlays.

**Example:** This example specifies that medium overlay O1FRONT is printed on the front side of each sheet and medium overlay O1BACK is printed on the back side of each sheet:

```
//AFPUER JOB ...
//STEP1 EXEC PGM=MYAPPL
//OUT1 OUTPUT OVERLAYF=O1FRONT,OVERLAYB=O1BACK
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

## Printing a page overlay

A page overlay is called either by adding the Include Page Overlay (IPO) structured field in the print data or by referencing the overlay in the page definition used for line data or XML data. To print page overlays with a line data or XML data application:

1. Create an overlay resource. You can use, for example, an AFP utility such as IBM Overlay Generation Language (OGL). For more information, see *Overlay Generation Language/370 User's Guide and Reference*.
2. Use the overlay, by either of these methods:

- Use a page definition that names the overlay for printing. Then, specify the name of the page definition in your JCL. You can create the page definition by use of an AFP utility such as IBM Page Printer Formatting Aid (PPFA). For more information, see [Page Printer Formatting Aid: User's Guide](#).
  - Code an IPO structured field in the print job, except in XML data.
3. To identify the page overlay as a preprinted form overlay, create a form definition with a Resource Object Include triplet (X'6C') on the PMC structured field that is specified with a PFO object type ID. You can build the form definition with an AFP utility that supports preprinted form overlays.

**Example:** This example, which is applicable to traditional line data only, shows you how to print an overlay named O1PAGE1 on page one of the output and an overlay named O1PAGE3 on page three of the output. This example includes the print records as part of the print job and uses the system utility IEBGENER to send them to the printer.

```
//PAGOVLY JOB ...
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//OUT1 OUTPUT PAGEDEF=P1USEROV
//SYSIN DD DUMMY
//SYSUT2 DD SYSOUT=A,DCB=(RECFM=FBA,BLKSIZE=80),OUTPUT=*.OUT1
//SYSUT1 DD *
1This is print record 1 of page 1.
!.....O1PAGE1.....
This is print record 2 of page 1.
This is print record 3 of page 1.
1This is print record 1 of page 2.
This is print record 2 of page 2.
This is print record 3 of page 2.
1This is print record 1 of page 3.
!.....O1PAGE3.....
This is print record 2 of page 3.
This is print record 3 of page 3.
1This is print record 1 of page 4.
This is print record 2 of page 4.
This is print record 3 of page 4.
```

The IPO structured field contains unprintable hexadecimal coding, represented by periods in the example. For information about coding the IPO structured field, see *Mixed Object Document Content Architecture Reference*; see also “AFP structured fields included in line data” on page 74.

MO:DCA data also requires an IPO structured field to call a page overlay. The coding of the IPO structured field is identical with the coding for line data. The name of the page overlay to be used on a page must be specified in the Map Page Overlay (MPO) structured field of the Active Environment Group for that page. If the MO:DCA application is created by a text-formatting product, that product might automatically create both the MPO and IPO records when the page overlay is requested. If you are writing the Active Environment Group records, see *Mixed Object Document Content Architecture Reference* for the content and structure of the MPO structured field.

## Positioning a page overlay

If you use an IPO structured field to include an overlay, you specify in the structured field the position at which the overlay is to print. If you use the page definition OVERLAY subcommand to include an overlay, you specify the position as a parameter on that command. This position includes the offset coded within the overlay resource. For ease of positioning, page overlays are created with an internal overlay offset of 0,0.

Take care when positioning page overlays in a page that has a rotated print direction, such as DOWN (90°), BACK (180°), or UP (270°). PSF positions the overlay relative to the logical page origin, which does not change when the print direction (that is, the text orientation) is changed. In addition, PSF positions the physical, upper-left corner of the overlay, which means that in a page with a DOWN print direction, the IPO position is measured from what appears to be the lower-left corner of the page to the lower-left corner of the overlay, as shown in [Figure 39 on page 121](#).

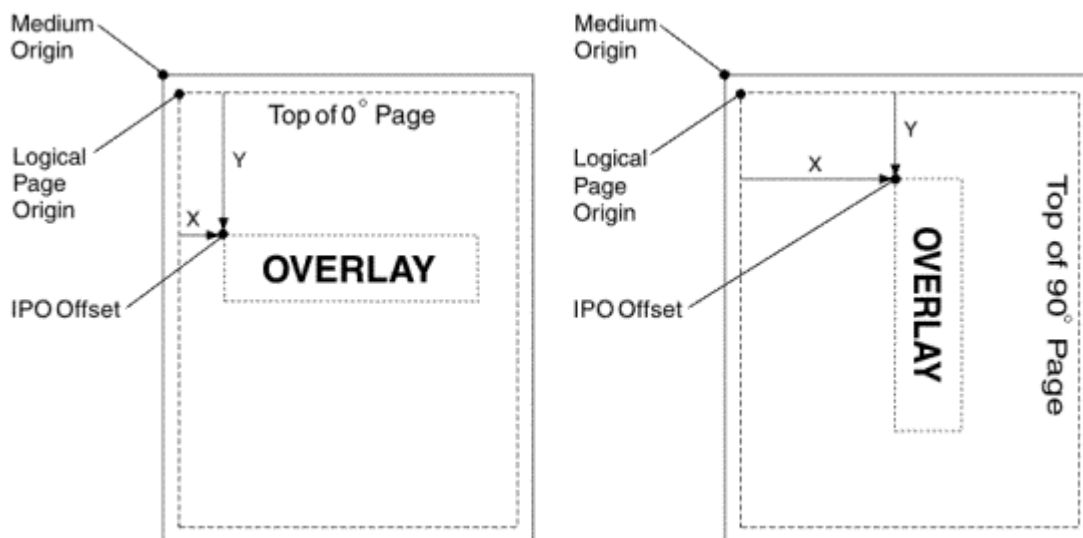


Figure 39. Positioning a page overlay

For more information about coding page overlays, see *Overlay Generation Language/370 User's Guide and Reference*.

## Printing line data or XML data with page definition options

Line data and XML data are printed according to instructions in the page definition used for printing the data set. A page definition that is coded in the PAGEDEF or the FCB parameter must be contained in one of the libraries available to the target printer. For more information, see [“Line data” on page 71](#) or [“XML data” on page 76](#).

To print line data, you can use the default page definition that is defined for your printer, or you can specify your own page definition. A default page definition can also be used to print XML data. However, since the page definition is tightly coupled to the XML data to be printed, it is more likely that you specify your own page definition.

## Specifying a page definition

If your job requires special page definition options, specify a page definition that contains those options in the JCL for the print job. To do this, code the page definition name, without its two-character prefix of P1, in the PAGEDEF parameter of the OUTPUT statement (see [“PAGEDEF” on page 100](#)). For compatibility with non-AFP printers, you can code the page definition name in the FCB parameter of the DD or OUTPUT statement (see [“FCB” on page 93](#)). The name that is coded in the FCB parameter cannot be more than 4 characters long.

The page definition must be stored in a private user library, a PSF user library (if supported for your system), or inline as part of the print data set (except in XML data). You can create your own page definition or use a page definition already available on your system. To use a page definition from a PSF user library, see [“Printing with resources from a user library” on page 142](#).

PSF provides standard page definitions for traditional line data, which are listed in Appendix B, [“Page definitions supplied with PSF,” on page 185](#). Page definitions are provided to fit standard paper sizes for AFP printers. These page definitions provide different combinations of line spacing and print direction, and some multiple-up definitions. The page definitions are referred to in the following examples.

### Examples:

1. This example specifies a user-created page definition named P1MYPDEF and a form definition named F1MYFDEF in the OUTPUT statement. Do not code the prefixes P1 and F1 in your JCL.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
```

```
//OUT2 OUTPUT PAGEDEF=MYPDEF,FORMDEF=MYFDEF  
//DD2 DD SYSOUT=A,OUT=(*.OUT2)
```

To create a page definition, use an AFP utility product such as IBM Page Printer Formatting Aid (PPFA). For information about how to use PPFA to create page definitions, see [Page Printer Formatting Aid: User's Guide](#). For information about form definitions, see [“Specifying a form definition” on page 112](#).

2. In this example, a page definition that is named P1STD1, supplied with PSF for compatibility with non-AFP printers, is specified with the FCB parameter in the DD statement. Because no form definition is selected for this job, the PSF default form definition for the printer is used.

```
//AFPUSERB JOB ...  
//STEP1 EXEC PGM=USERB  
//DD2 DD SYSOUT=A,FCB=STD1
```

3. In this example, an FCB is specified in the OUTPUT statement:

```
//AFPUSERD JOB ...  
//STEP1 EXEC PGM=USERD  
//OUTPUT1 OUTPUT FCB=STD1  
//DD1 DD SYSOUT=A,OUTPUT=(*.OUTPUT1)
```

4. In this example, page definition P1MYPDEF is coded inline in the print data set that is generated by the program USERC. For information about coding inline resources, see *Mixed Object Document Content Architecture Reference*.

```
//AFPUSERC JOB ...  
//STEP1 EXEC PGM=USERC  
//OUT1 OUTPUT PAGEDEF=DUMMY  
//DD1 DD SYSOUT=A,OUT=(*.OUT1)
```

The PAGEDEF parameter must specify either the resource name of the inline page definition or the keyword DUMMY. If the name in the PAGEDEF parameter does not match the name of an inline page definition, PSF uses the resource from the resource library that matches the name in the JCL. If the PAGEDEF parameter is not specified in the JCL, PSF selects the first inline page definition in the print data set.

You can store the page definition in any of these places:

- In a system library assigned to PSF for your printer
- In a user library referenced in your JCL
- Inline in the print data set, except in XML data

For more information about the function of a page definition in printing, see [“Page definitions” on page 63](#).

## Using page definitions from a user library

You can instruct PSF to select a page definition from your user library rather than from a system library assigned to PSF. To use a page definition from a user library:

1. Include in your JCL a reference to the user library that contains the page definition. For details, see [“USERLIB” on page 107](#).
2. Specify the name of the page definition in the JCL PAGEDEF parameter or the FCB parameter of your JCL.

## Using inline page definitions

To use an inline page definition:

1. Include the inline page definition in the print data set.
2. If you specify the PAGEDEF parameter or the FCB parameter in your JCL, ensure that the name of the inline page definition matches the name of the page definition name that is specified in your JCL, or else specify PAGEDEF=DUMMY in the JCL.

3. If a page definition resource is included inline with the data, ensure to identify the data set as containing carriage control characters. If the length of the records in the page definition is less than or equal to the logical-record length defined for the data set, you can specify fixed-length records for the record format. If the length of the records in the page definition is greater than the logical-record length defined for the data set, you must specify variable-length records variable-blocked with ANSI carriage control characters (VBA) or variable-blocked with machine carriage control characters (VBM) for the record format.

**Notes:**

1. If you specify PAGEDEF=DUMMY in your JCL, and you do not include an inline page definition, PSF uses the default page definition for your printer.
2. If you specify multiple inline page definitions in the print data set and you specify PAGEDEF=DUMMY on the OUTPUT statement, PSF uses the last inline page definition in the print data set.
3. If you do not specify the PAGEDEF or FCB parameter in your JCL, PSF selects the first inline page definition in the print data set, unless a JES default page definition exists.
4. You cannot use inline resources in XML data.

You can include more than one inline page definition in a print data set, and you can change the page definition name in the JCL on different printing jobs to test different page definitions. However, if the name of an inline page definition does not match the PAGEDEF name that is specified in the JCL, PSF uses the page definition from the resource library that matches the name in the JCL. For more information about using inline page definitions, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

## Using page definitions converted from FCBs

PSF supports line data applications that are designed for line printers so that those jobs can be processed on a page printer with no need to change the application program or its JCL. If you do not specify a PAGEDEF parameter, but you do specify a forms control buffer (FCB) parameter, PSF uses the FCB name as the page definition name.

If you are using line printers, you are probably using some of the FCB modules provided by IBM. Those FCBs are converted to page definitions and are provided in the system page definition library. If you used other FCB modules for your line printer, check with your system support group to see whether the FCBs are converted to page definitions before you attempt to use them.

## Specifying print direction

For line data or XML data, the page definition can specify a print direction of ACROSS, DOWN, BACK, or UP. For a description of AFP print directions, see [“Print direction” on page 64](#). In the following examples, PSF-supplied page definitions are used to control the print direction.

**Examples:**

1. This example specifies printing in the DOWN direction on a 3800 printer loaded with forms that measure 12 inches wide by 8.5 inches high. The resulting output prints in the portrait format; that is, the page is turned so that its top is the short (8.5-inch) edge of the form.

```
//AFPUSERA JOB ...  
//STEP1 EXEC PGM=USERA  
//OUT1 OUTPUT PAGEDEF=06061  
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)  
/*
```

2. This example specifies printing in the ACROSS direction on a cut-sheet printer. The resulting output is in portrait format because the top of the page is the shorter, 8.5-inch side of the form. For IBM cut-sheet printers such as the 3825 printer, the top of an ACROSS page is always the short side of the form.

```
//AFPUSERB JOB ...  
//STEP1 EXEC PGM=USERB  
//OUT1 OUTPUT PAGEDEF=A06462
```

```
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

3. This example specifies printing in the ACROSS direction on a 3835 printer. Because the 3835 printer has the Page Presentation Compatibility feature, the output is in portrait format unless the form definition that is used affects page compatibility. For more information, see [“Page-presentation compatibility”](#) on page 59.

```
//AFPUSERC JOB ...
//STEP1 EXEC PGM=USERC
//OUT1 OUTPUT PAGEDEF=A06462
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

## Specifying lines per inch spacing

For line data and XML data print jobs, the page definition controls the spacing of the print lines on the page. The following examples use page definitions that are supplied with PSF to specify different lines per inch (lpi) spacing for traditional line data.

### Examples:

1. This example specifies printing at 8 lpi on a 3800 printer. Page definition P106080 prints 60 lines in the ACROSS direction on forms 12 inches wide and 8.5 inches high. The job also specifies the GT12 font, recommended for this page definition.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT PAGEDEF=06080
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1),CHARS=GT12
/*
```

2. This example prints at 8.5 lpi on a non-3800 printer. Page definition P1V06683 prints 66 lines in the DOWN print direction on letter-sized paper. The job also specifies the 60D8 font, recommended for this page definition as suitable for printing at 8.5-lpi spacing.

```
//AFPUSERB JOB ...
//STEP1 EXEC PGM=USERB
//OUT1 OUTPUT PAGEDEF=V06683
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1),CHARS=60D8
/*
```

## Specifying multiple-up printing

For traditional line data print jobs, the page definition can specify the arrangement of print lines to enable multiple pages of application data to fit on a single printed page. This arrangement is called *multiple-up* printing, which is not the same as N\_UP printing. For more information about N\_UP printing, see [“N\\_UP printing: printing multiple pages on a sheet”](#) on page 125.

Multiple-up printing reduces the number of pages that are required to print a file and can also increase the throughput (that is, the number of pages of application data per minute) of your printer, freeing your printer for other work. In addition to saving paper, multiple-up printing saves space that is required to store blank forms and printed output. To compound these benefits, see [“Specifying duplex printing”](#) on page 117.

The following examples use page definitions that are supplied with PSF to specify multiple-up printing. When you are printing multiple-up, be careful to select a font small enough to allow the data to fit on the page.

### Examples:

1. This example specifies printing two side-by-side application pages of 66 lines each on a 3800 printer. Page definition P1M13280 is designed for forms 12 inches wide and 8.5 inches high. No font is specified in the JCL for the print job because the page definition itself specifies that font GT24 is to be used for printing.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT PAGEDEF=M13280
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

2. This example specifies how to use page definition P1W120C2 to print two over/under application pages of 60 lines each on a non-3800 printer. Page definition P1W120C2 is designed for letter-size cut-sheet paper or continuous-forms paper 12 inches wide by 8.5 inches long.

Program listings, dumps, and similar application output that are typically printed at 229 impressions per minute (ipm) on a 3900 printer (1 up) are still printed at or near the same ipm, but are printed at two pages, rather than one page, per impression. By printing multiple-up (with two pages on each side of a sheet), you can effectively print as many as 458 pages of application data per minute.

```
//AFPUSER1 JOB ...
//STEP1 EXEC PGM=USER1
//OUT1 OUTPUT PAGEDEF=W120C2,FORMDEF=C10110
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1), CHARS=GT20
```

## N\_UP printing: printing multiple pages on a sheet

With PSF, you can print up to four pages on a sheet printed in simplex mode, or up to eight pages on a sheet printed in duplex mode, by creating a new form definition that contains the PPFA N\_UP subcommand. To create an application such as the one described here, you need a printer that supports N\_UP printing, such as an InfoPrint printer. You can use the PPFA program to create the required form definition.

As an example, you might want to print pages one and two of a letter side by side on a single sheet of paper, and then print pages three and four on a second sheet, thus printing a four-page letter on two sheets of paper, as shown in [Figure 40 on page 125](#). You might also want to specify different overlays or page segments for each page. For example, page one of your letter might contain a company logo, and page four a signature. To avoid printing pages too small to read, you might want to print on a 3935 or a 3900-0W1 printer, either of which can print data up to 17 inches wide.

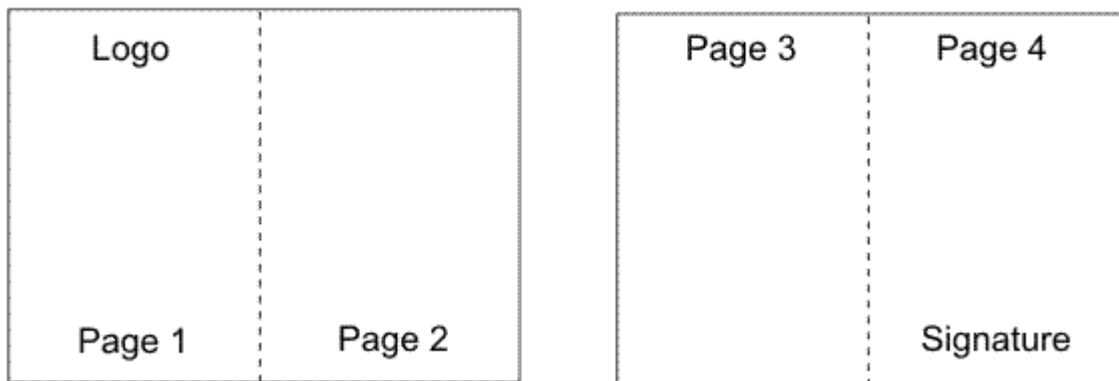


Figure 40. Printing four pages on two sheets

To create a job similar to the one diagrammed in [Figure 40 on page 125](#), begin by using PPFA to create a form definition that specifies:

- The N\_UP subcommand
- The ACROSS printing direction
- The PORTRAIT page presentation
- A font of the correct size to allow your text to fit in the page area

The *Page Printer Formatting Aid: User's Guide* describes the types of N\_UP printing you can perform, shows the command syntax, and provides numerous examples for basic N\_UP printing and the



commands and syntax for enhanced N\_UP printing. For the N\_UP 2 form definitions that are supplied with PSF, see Table 16 on page 178.

**Example:** In this example, SYSOUT A prints on the 3900-0W1 printer. No font is specified in the JCL because your page definition names a font.

```
//AFPUSER JOB ...
//STEP1 EXEC PGM=USERA
//OUTPUT1 OUTPUT PAGEDEF=nuppdf,FORMDEF=nuppdf
//PRINT DD SYSOUT=A,OUTPUT=(*.OUTPUT1)
/*
```

## Suppressing print data

For line data and XML data, you can tell PSF not to print certain fields in the print records you send to the printer. You do this by using either selective field formatting or print suppression.

You can format selective fields by using a page definition to format only the fields in the print record that you want to print. Fields that you omit from the format descriptions in the page definition are not sent to the printer by PSF.

To suppress print data, use a page definition to format the fields and label the fields as eligible for suppression. Then, use a form definition to specify which of the labeled fields are not to be printed. The suppressed fields are not sent to the printer. Use this method of suppression when you are printing multiple copies of a page and you want the fields printed on some copies but suppressed on other copies.

To create page definitions and form definitions that format fields and suppress data, use an AFP program such as IBM Page Printer Formatting Aid (PPFA). For more information, see [Page Printer Formatting Aid: User's Guide](#).

### Examples:

1. This example uses a user-created page definition to format only the first 80 bytes of a 120-byte record.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT PAGEDEF=FORMAT
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

2. In this example, a user-created page definition formats all the fields to be printed and labels a Salary field for suppression. The user-created form definition prints two copies of each page, suppressing the Salary field on the first copy.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT PAGEDEF=SUPSAL,FORMDEF=SUPSAL
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

## Specifying and selecting fonts

To use fonts with PSF, you make two choices: you *specify* fonts and *select* fonts. You specify fonts to print the entire print data set. You select fonts to print individual lines or fields of data.

For MO:DCA-P data, you specify and select the fonts within the data stream. For data that PSF must compose into pages, the fonts are specified either in a page definition or in JCL with the CHARS parameter, but not in both. For a single data set, you cannot mix fonts that are specified in a page definition with fonts that are specified in JCL. Select fonts with table reference characters (TRCs), with AFP control records, or in a page definition.

For printing traditional line data, you can specify FOCA fonts in your JCL or all AFP fonts, including TrueType and OpenType fonts, in the page definition. If you do not specify fonts, PSF uses the default font for the printer. In record format line data and XML data, you can specify fonts only in the page definition.



If fonts are needed by a record format or XML page definition and none are specified, PSF issues an error message.

If you want to print the entire data set in a single direction, you can specify fonts only in JCL. PSF uses the fonts that have 0° character rotation for the specified direction. When a data set requires fonts with more than one print direction or character rotation, you must specify the fonts in the page definition. To verify whether fonts can be specified in JCL for the default page definition, see the "Page-Printer Defaults" form. Not all printers can print in all four directions. For information about the print directions your printer supports, see the documentation that is provided with the printer.

This hierarchy shows the order in which PSF selects fonts:

1. Fonts that are specified in a page definition or FCB that is specified with the PAGEDEF or FCB parameter on the OUTPUT statement. For more information, see [“Fonts” on page 67](#).
2. CHARS parameter that is specified on a DD statement.
3. CHARS parameter that is specified on an OUTPUT statement.
4. UCS<sup>16</sup> parameter that is specified on a DD statement.
5. UCS parameter that is specified on an OUTPUT statement.
6. JES installation default fonts that are defined for your printer. For more information, contact your system programmer. If you are using the "Page-Printer Defaults" form, the default fonts are identified.
7. Fonts that are specified in a default page definition that is specified with the PAGEDEF parameter in the PRINTDEV statement or the **Page definition** parameter in the Printer Inventory.

Select a font that is of the correct size for the amount of data and the lines per inch spacing of your print job. Recommended FOCA fonts are listed with the page definition descriptions in [Appendix B, “Page definitions supplied with PSF,” on page 185](#). Also see [“Page definition line-spacing values and fonts” on page 192](#) for tables that list the line-spacing values for some of the page definitions that are supplied with PSF.

The following examples apply only to fonts used in traditional line data applications. In MO:DCA-P documents or overlays, fonts are defined in structured fields within the document or overlay. See the reference publication for the AFP utility you used to create the overlay or document, or see *Mixed Object Document Content Architecture Reference*.

For more information about fonts, see [“Fonts” on page 19](#); for information about referencing page definitions, see [“Specifying a page definition” on page 121](#).

#### Examples:

1. In this example, the JCL CHARS parameter is specified in the DD statement:

```
//DD1 DD SYSOUT=A,CHARS=(60D8,50FB,GT12)
```

2. In this example, the JCL CHARS parameter is specified in the OUTPUT statement:

```
//OUT1 OUTPUT CHARS=(60D8,50FB,GT12)
//DD1 DD SYSOUT=A,OUTPUT=(*.OUT1)
```

3. In this example, the JCL CHARS parameter is used to select font 60D8. Because no font is specified in page definition P1V06683, PSF uses the font that is specified in the CHARS parameter.

```
//AFPUSERB JOB ...
//STEP1 EXEC PGM=USERB
//OUT1 OUTPUT PAGEDEF=V06683
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1),CHARS=60D8
/*
```

4. This example uses a page definition that contains a font specification. The specified font can be a FOCA font or a TrueType and OpenType font. You do not need to code a CHARS parameter in your JCL. If you do code the CHARS parameter in your JCL, PSF ignores it and uses the font that is specified in the page definition.

---

<sup>16</sup> The UCS parameter is never used in deferred-printing mode under JES3.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT PAGEDEF=M13280
//PRINT DD SYSOUT=A,OUTPUT=(*,OUT1)
/*
```

5. This example uses the default page definition for the printer and specifies font 60D8 in the JCL.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//PRINT DD SYSOUT=A,CHARS=60D8
/*
```

Whether the 60D8 font is used depends on how the default page definition is defined for your printer. Your system programmer might provide this information on the form that is shown in [Appendix D, “Page-printer defaults form,”](#) on page 221.

For the examples in this information, assume that the printer is defined so that specifications of fonts in the CHARS parameter in the JCL override specifications of fonts in the default page definition.

6. In this example, multiple fonts are listed in the JCL CHARS parameter, which means that you can select fonts by using table reference characters (TRC) in the print data records. The presence of TRCs in the print data is indicated in the TRC parameter of the OUTPUT statement. If TRCs are not specified in the print data, all the data prints in the first font in the list, 60D8.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT TRC=YES
//PRINT DD SYSOUT=A,CHARS=(60D8,50FB)
/*
```

Fonts for TRC selection can also be specified in the page definition. For more information, see [“Using table reference characters to select fonts”](#) on page 133.

## Changing formatting within a document

In the examples in [“Specifying and selecting fonts”](#) on page 126, the same formatting specifications are used for all the pages of the line data set. But suppose that you want to change some of the formatting in the middle of your data set. For example, suppose that you want some pages printed at six lines per inch and other, summary pages printed at 8 lines per inch.

To change page definition options, such as print direction, line spacing, or record formatting in a print job, you need to use a page definition that contains multiple page formats. You also must tell PSF when to use which page format. For an example that shows how to do this, see [“Using multiple copy groups or page formats”](#) on page 128.

## Using multiple copy groups or page formats

If your print job requires you to change copy group options, such as medium overlays or paper source, for different pages in the data set, do these steps:

1. Create a form definition that contains multiple copy groups, with the options you want for different pages coded in different copy groups.
2. Identify that form definition in the JCL of your print job by the method described in [“Specifying a form definition”](#) on page 112.
3. In your print data, include an Invoke Medium Map (IMM) structured field in front of any page on which you want to change the copy groups.
4. Alternatively, you might be able to use conditional processing in a page definition to trigger the use of a new copy group determined by the content of data fields in the application. This conditional processing eliminates the need to code IMM structured fields in the data.

**Note:** Instead of defining a copy group in a form definition, you can define a copy group in the print data set. See [“Internal copy groups”](#) on page 51 for more information.

Similarly, if your job requires different page definition options for different pages in the data set, such as lines per inch spacing or print direction, do these steps:

1. Create a page definition that contains multiple page formats, with the options you want for different pages coded in different page formats.
2. Identify that page definition in the JCL of your print job, as described in [“Specifying a page definition” on page 121](#).
3. In your print data, include an Invoke Data Map (IDM) structured field in front of any page on which you want to change the page formats.
4. Alternatively, you might be able to use conditional processing in a page definition to use a new page formats base on data fields in the application. This conditional processing eliminates the need to code IDM structured fields in the data.

You can include the IMM and IDM structured fields yourself, or you can call them by using the conditional processing function of the page definition. For more information, see [“Conditional processing” on page 67](#) and [“AFP structured fields included in line data” on page 74](#).

**Example:** In the following example, pages 1, 2, and 4 are printed at a line spacing of six lines per inch (lpi) on paper from the main paper source. Page three is printed at a spacing of 8 lpi on blue paper from the alternative paper source.

Form definition F1MUCG is created containing two copy groups:

Page format F2MUMAIN prints on paper from the main paper source.

Page format F2MUALTB prints on paper from the alternative paper source.

Page definition P1MUFMT is created containing two page formats:

Page format P2MU6LPI prints pages at 6 lpi.

Page format P2MU8LPI prints pages at 8 lpi.

This example shows the IMM and IDM structured fields that are coded in the traditional line data print data set. The IMM and IDM structured fields contain unprintable hexadecimal fields, which are represented as periods in the coding example. The format of these records is described in *Mixed Object Document Content Architecture Reference*.

For this example, the print records are included as part of the print job, and the system IEBGENER utility is used to send them to the printer. Because the data set contains structured fields, the record format must be defined with carriage control (RECFM=FBA).

```
//AFPIDM JOB ...
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//OUT1 OUTPUT FORMDEF=MUCG,PAGEDEF=MUFMT
//SYSIN DD DUMMY
//SYSUT2 DD SYSOUT=A,DCB=(RECFM=FBA,BLKSIZE=80),OUTPUT=*.OUT1
//SYSUT1 DD *
!.....F2MUMAIN
!.....P2MU6LPI
1This is print record 1 of page 1.
This is print record 2 of page 1.
This is print record 3 of page 1.
1This is print record 1 of page 2.
This is print record 2 of page 2.
This is print record 3 of page 2.
!.....F2MUALTB
!.....P2MU8LPI
1This is print record 1 of page 3.
This is print record 2 of page 3.
This is print record 3 of page 3.
!.....F2MUMAIN
!.....P2MU6LPI
1This is print record 1 of page 4.
This is print record 2 of page 4.
This is print record 3 of page 4.
```

If the IMM and IDM structured fields are omitted before page one of the data set, PSF begins printing with the first copy group in the form definition and with the first page format in the page definition.

If conditional processing is used for this application, IMM and IDM structured fields are not required in the print data. For more information about using conditional processing, see [“Conditional processing”](#) on page 67.

## Printing page segments

You can include page segments to be printed as part of an overlay resource, or you can include them as part of the data by using an Include Page Segment (IPS) structured field or an Include Object (IOB) structured field. The IPS and IOB structured fields that name and position the page segment are included as a record in the print data set.

When you are using IPS structured fields in line data sets, you can also map the page segment in the page definition to retain it in the printer while your data set is printing. Mapping the page segment can speed performance if the page segment is used multiple times in the same print job.

### Examples:

1. In this example, page segment S1LOGO is specified in an overlay. The overlay is referenced in the form definition F1USERA.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT FORMDEF=USERA
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

For information about form definitions and overlays, see [“Specifying a form definition”](#) on page 112 and [“Printing a medium overlay”](#) on page 119. For information about specifying page segments in an overlay, see the publications for the AFP utility product that is used to create your overlay.

2. In this example, page segments are called by using IPS records in a traditional line data set. Page segment S1LOGO is printed on all four pages of the document. Page segment S1MAP is printed only on pages two and four. The user listed S1LOGO in the page definition but did not list S1MAP in the page definition.

This example shows how a hard page segment is coded by naming it in the page definition Segment List:

```
setunits 10 cpi 6 lpi linesp 6 lpi ;
pagedef nnxx0 width 8.3 in height 10.8 in replace no ;
font fnorm cr10 ;
font fbold cb10 ;

pageformat p2nnxxx0 direction across ;
trcref 0 font fnorm ;
trcref 1 font fbold ;
segment S1LOGO ;

printline channel 1 repeat 1 position 5 5 font fnorm;
/* name */ field start 6 length 20 position current current ;
/* acct */ field start 1 length 5 position 20 current font fbold;
/* acct */ field start 1 length 5 position 70 -4 direction down ;

printline repeat 3 position 0 next font fnorm ;

printline channel 2 repeat 55 position 0 next ;
```

For this example, the print records are included as part of the print job, and the system IEBGENER utility is used to send them to the printer. Because structured field records are included in the data set, the record format must be defined with carriage control (RECFM=FBA).

```
//AFPIPS JOB ...
/*ROUTE PRINT DL3820B
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//OUT1 OUTPUT PAGEDEF=USERB
//SYSIN DD DUMMY
//SYSUT2 DD SYSOUT=A,DCB=(RECFM=FBA,BLKSIZE=80),OUTPUT=*.OUT1
//SYSUT1 DD *
1This is print record 1 of page 1.
```

```

!.....S1LOGO.....
This is print record 2 of page 1.
This is print record 3 of page 1.
1This is print record 1 of page 2.
!.....S1LOGO.....
!.....S1MAP.....
This is print record 2 of page 2.
This is print record 3 of page 2.
1This is print record 1 of page 3.
!.....S1LOGO.....
This is print record 2 of page 3.
This is print record 3 of page 3.
1This is print record 1 of page 4.
!.....S1LOGO.....
!.....S1MAP.....
This is print record 2 of page 4.
This is print record 3 of page 4.

```

Because S1LOGO is listed in the page definition, it is loaded in the printer at the beginning of page one and used for all four pages. S1MAP, which is not listed in the page definition, is loaded in the printer at page 2, and loaded again for page 4.

The IPS structured field contains unprintable hexadecimal coding, which is represented by periods in the example. For information about coding the IPS structured field, see *Mixed Object Document Content Architecture Reference* and see “AFP structured fields included in line data” on page 74.

## Printing MO:DCA-P data

MO:DCA-P data is printed according to instructions in the MO:DCA-P structured fields in the data set. To change the formatting of a page document, you change the structured fields. For a description of the structured fields that are used in page applications, see *Mixed Object Document Content Architecture Reference* or the publication for the product that is used in creating your MO:DCA-P data. PSF supports MO:DCA Presentation Interchange Set (IS) data streams, including:

### MO:DCA AFP/Archive (AFP/A)

MO:DCA AFP/A is an AFP document architecture interchange set that is used for long-term preservation and retrieval. This subset ensures page independence and eliminates images without clearly specified resolution, device default fonts, and external resources.

### MO:DCA IS/3

MO:DCA IS/3 is the first interchange set to achieve industry consensus through a rigorous open standards process. It improves existing functions and introduces new functions, such as Begin Print File (BPF) and End Print File (EPF) structured fields, and multiple image TIFF object support.

### MO:DCA AFP/A, IS/3

MO:DCA AFP/A, IS/3 is an AFP document architecture interchange set that complies with the rules and restrictions of both the AFP/Archive and IS/3 interchange sets.

### MO:DCA Graphic Arts Function Set (GA)

MO:DCA GA is an extension of MO:DCA IS/3 that adds PDF presentation object support.

**Example:** This example shows the code for submitting the output of a page application to an AFP printer.

```

//AFPPAGE JOB ...
//STEP1 EXEC PGM=PAGEA
//OUT1 OUTPUT FORMDEF=A10111
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1),
//      DCB=(RECFM=VBA,LRECL=1993,BLKSIZE=32760)
/*

```

No page definition or fonts are specified in the JCL because the MO:DCA-P data contains its own internal formatting commands and font definitions. However, a form definition is required. If you do not specify a form definition in the JCL, PSF uses the default form definition for the printer. For more information, see “Specifying a form definition” on page 112.

MO:DCA-P data must be defined with carriage controls, which are specified in the RECFM parameter. MO:DCA-P data is typically written as variable-length blocked records, with RECFM coded as either VBA or VBM. Record size and block size might vary.

## Specifying carriage control and table reference characters in line data

Traditional line data can contain CC characters, TRCs, both, or neither. You can use either of two types of CC characters: American National Standards Institute (ANSI) or machine code. To indicate that the records contain CC characters, and to indicate which type is being used, specify A or M in the DCB=RECFM subparameter.

To indicate that TRCs are contained in the records, specify TRC=YES in the OUTPUT JCL statement (see “TRC” on page 106) or DCB=OPTCD=J in the DD statement.

### Using carriage control characters in line data records

CC characters are used in traditional line data to control writing, spacing, and skipping operations as the data is being formatted. The FCB is used to map CC characters to the physical actions that line printers use to format print data vertically. For page printers, the page definition replaces the function of the FCB. If you are using a page definition with a page printer to replace an FCB, the page definition must specify the same number of lines per inch and define the same actions for CC characters as the FCB used with the line printer.

You can use the Page Printer Formatting Aid (PPFA) product to create page definitions that specify the same skipping and spacing instructions as the FCBs used with existing traditional line data applications.

Table 5 on page 132 lists the hexadecimal CC characters that you can use. The decimal representations of the ANSI codes are in parentheses.

Table 5. Carriage control characters			
Action	Machine Code Control Characters Action After Printing	Machine Code Control Characters Action Only (Immediate)	ANSI Control Characters Action Before Printing
Print (no space)	01	-	4E (+)
Space 1 line	09	0B	40 ( )
Space 2 lines	11	13	F0 (0)
Space 3 lines	19	1B	60 (-)
Skip to Channel 1	89	8B	F1 (1)
Skip to Channel 2	91	93	F2 (2)
Skip to Channel 3	99	9B	F3 (3)
Skip to Channel 4	A1	A3	F4 (4)
Skip to Channel 5	A9	AB	F5 (5)
Skip to Channel 6	B1	B3	F6 (6)
Skip to Channel 7	B9	BB	F7 (7)
Skip to Channel 8	C1	C3	F8 (8)
Skip to Channel 9	C9	CB	F9 (9)
Skip to Channel 10	D1	D3	C1 (A)
Skip to Channel 11	D9	DB	C2 (B)
Skip to Channel 12	E1	E3	C3 (C)

**Note:** PSF ignores these hexadecimal machine CC characters: 02–07, 0A, 12, 23, 43, 63, 6B, 73, 7B, EB, F3, and FB. PSF prints data lines that contain these characters in single-spacing mode.

If your application creates line data records containing CC characters that skip to a channel, the page definition that is used to print the application output must contain instructions for processing those channel codes. If the skipping or spacing actions you specified in a page definition move the print position

past the last line of the current page, PSF starts a new page at the first print position indicated by the page format. However, PSF does not carry skipping or spacing over to the new page.

An alternative to using CC characters or channel codes in line data records is to define data placement entirely within the page definition by using line count or field formatting.

## Using table reference characters to select fonts

You can use TRCs in traditional line data records to select the font to be used in printing full lines or text in a line. To select fonts for full lines, code a TRC in each output data line. When traditional line data contains both CC characters and TRCs, the CC character precedes the TRC, as in [Figure 31 on page 72](#). The TRC (0, 1, 2, or 3) selects the font corresponding to the order in which you specified the font names with the JCL CHARS parameter.

You can also code TRCs that correspond to font names specified in a font list defined for the page definition. When coding TRCs, your COPYGROUP must reference FOCA fonts only or TrueType and OpenType fonts only. You cannot have a mixture of both types of fonts. Although you can specify 128 fonts in a page definition, the printer can allow fewer than 128 fonts per page.

## Examples of selecting fonts with TRCs

1. This example shows how to use TRCs to specify fonts in line data. The example uses column 1 for the CC character and column 2 for the TRC (0, 1, 2, or 3).

```
12This line should print in a bold font.
1This is a normal print line.
```

2. This example creates printing resources by using the PPFA PAGEDEF TRCREF command. Values different from the default values are specified in the DIRECTION and ROTATION subcommands of the TRCREF command.

Page definition source code example:

```
SETUNITS LINESP 8 LPI;

FORMDEF rhdr (form definition name)
  REPLACE yes
  OFFSET .5 in .5 in
  DUPLEX NORMAL;

PAGEDEF rhdr (page definition name)

  WIDTH 8.5 IN
  HEIGHT 10.0 IN
  LINEONE 1.0 IN 1.0 IN
  DIRECTION ACROSS
  REPLACE YES;

  FONT normal 70D0 ROTATION 0;
  FONT heading 60D8 ;

  PAGEFORMAT rhdr;

  TRCREF 1 FONT normal;
  TRCREF 2 FONT heading DIRECTION DOWN ROTATION 270 ;

  PRINTLINE CHANNEL 1
    POSITION 8.0 IN 1.0 IN
    DIRECTION DOWN;

  PRINTLINE
    POSITION MARGIN 2.0 IN
    DIRECTION ACROSS
    REPEAT 60;
```

Input for page definition code example:

```
=====
12Chapter 1 should be printed in font 60D8.
1This is a normal print line and should be printed in font 70D0.
1
1Some normal text for illustration, chapter 1.
```

```

12Chapter 2 should be printed in font 60D8.
1This is a normal print line and should be printed in font 70D0.
1
1Some normal text for illustration, chapter 2.
=====

```

3. This example uses the TRC parameter in the OUTPUT statement to specify that the traditional line data set contains TRCs:

```

//OUT2 OUTPUT CHARS=(60D8,60D0),TRC=YES
//DD2 DD SYSOUT=A,OUTPUT=*.OUT2

```

4. This example uses the DCB subparameter of the DD statement to specify that the data set contains TRCs:

```

//OUT2 OUTPUT CHARS=(60D8, 60D0)
//DD2 DD SYSOUT=A,OUTPUT=*.OUT2,DCB=OPTCD=J

```

The order in which the fonts are specified in the CHARS parameter establishes which number is assigned to each associated TRC. For example, the reference characters for the fonts in this example are 0 for the first font name listed and 1 for second font name.

## Rules for coding table reference characters

PSF uses the TRC from the output line to select a font. When you are coding a TRC in an output data line, remember these rules:

- For TRCs corresponding to font names specified in the CHARS parameter:
  - The valid TRCs are 0, 1, 2, and 3 and are called compatibility TRCs because they are compatible with the 3800 Model 1 printer. PSF ignores the leftmost 4 bits of the TRCs. Thus, X'F0' and X'00' are both valid representations for zero.
  - A TRC that refers to a font with a number higher than the number loaded by use of the CHARS parameter defaults to 0. For example, if two fonts are specified with CHARS, a TRC of 2 (referring to a third font) defaults to 0 and selects the first font specified. A number that is not valid, such as 4, also defaults to 0.
- For TRCs corresponding to font names specified in a page definition:
  - Valid TRCs are 0–127, inclusive. If four or fewer fonts are specified, they are treated as compatibility TRCs, and the leftmost 4 bits of the TRC are ignored. In this case, X'F0' and X'00' are both valid representations for 0. If more than four fonts are specified, PSF treats them the same as non-compatibility TRCs and reads all 8 bits. In this case, X'00' is 0, but X'F0' is decimal 240.
  - A TRC that refers to a font with a number higher than the number of fonts that are specified in the page definition defaults to the first font in the page definition.
  - A TRC of 0 selects the first font defined in the page definition.
  - A TRC higher than 127 selects the first font defined in the page definition.
- If both TRC=NO and DCB=OPTCD=J are specified, PSF ignores the TRC=NO parameter and expects the line data to contain multiple fonts.
- If TRC=YES or DCB=OPTCD=J is specified, but the data set contains no TRCs, the first character of each line (or the second character if carriage control characters are used) is interpreted as the font identifier. Therefore, the font that is used to print each line of the data set might not be the one you expect.
- If you do not specify TRC=YES or DCB=OPTCD=J in the JCL, but your line data contains a TRC as the first character of each line (or the second character if carriage control characters are used), the TRC is not used as a font identifier, but is printed as a text character.



## Merging data lines into a single print line

PSF can merge multiple input data records to print a single print line. Thus you can print composite characters with line data. You can also use this function to create printed lines in which different fields are printed in different fonts.

If you type a print-with-no-space control character at the beginning of an input data line, that line is superimposed on the next. The print-with-no-space control character indicates that no lines are skipped before or after this line is printed. You can superimpose as many lines as you like by typing that control character at the beginning of each of those lines, except for the last line. If you are coding with machine-code print control characters, the print-with-no-space control character is X'01'. If you are coding with ANSI control characters, the print-with-no-space control character is X'4E' or the "+" sign.

**Note:** If a data set containing merged lines is created for printing on a line printer, a page printer might not produce the same results.

When you use PSF to merge lines that use fonts with different pitches or typographic fonts, be aware that lines are merged pel by pel, not character by character. When the job is printed, 10 characters in the first line align with 10 characters in the second line only if the characters in both lines have the same pel width. Otherwise, overprinting can occur, as in [Figure 41 on page 135](#), where the data in the third record is superimposed on the data from the second record.

I AM THE FIRST LINE TO BE PRINTED

I AM THE SECOND LINE TO BE PRINTED  
I AM THE SECOND LINE MERGED WITH ANOTHER RECORD

*Figure 41. Output containing merged lines printed with a typographic font*

## Example of merging data lines

The application in [Figure 42 on page 135](#) shows three input data records, of which two are to be merged into a single print line.

```
//MERGEIT JOB
//STEP1 EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=A
//SYSIN DD DUMMY
//SYSUT2 DD SYSOUT=S,DCB=(RECFM=FBA,LRECL=80,BLKSIZE=2000)
//SYSUT1 DD *
1THIS IS THE FIRST LINE ON THE PAGE
  THIS IS THE SECOND LINE
+
/*
//
```

*Figure 42. IEBGENER example of merging two print lines*

The resulting printed output lines are:

```
THIS IS THE FIRST LINE ON THE PAGE
THIS IS THE SECOND LINE - THIS FRAGMENT IS MERGED WITH THE SECOND LINE
```

In the example in [Figure 42 on page 135](#), to achieve the merged output line, blanks are entered in the third input data record to place "- THIS FRAGMENT IS MERGED WITH THE SECOND LINE" in the correct position.

By using different fonts in lines to be merged, you can highlight or subordinate data in a print line, or you can print different columns of data across a page in different styles or sizes. Or, alternatively, you can select fonts in a page definition and then specify their use for specific fields in a record.

## Specifying shift-out, shift-in (SOSI) codes

To change from one type of character code to another in a data set, PSF provides special SOSI processing. PSF uses the process mode values SOSI1, SOSI2, SOSI3, and SOSI4 to print data sets containing both single-byte and double-byte fonts. You can specify the SOSI codes on one of these:

- PRMODE parameter of the OUTPUT JCL statement (see “PRMODE” on page 102)
- **Default process mode** parameter in the Printer Inventory
- XTP7PRMD in Exit 7
- PRMODE parameter on the PRINTDEV statement

See *PSF for z/OS: Customization* for information about specifying PRMODE on the PRINTDEV statement, specifying a default PRMODE in Exit 7, or specifying the **Default process mode** parameter in the Printer Inventory.

You can specify a particular single-byte and double-byte font pair for a line or field, or you can use the same single-byte and double-byte font pair for the entire page. If you specify a specific single-byte and double-byte font pair for a line or a field, you must use the font list in a page definition to specify the single-byte fonts and double-byte fonts you want to use for the page. Then, you can use more than one single-byte font and more than one double-byte font per page.

If you use the same single-byte and double-byte font pair for the entire page, you must specify the single-byte font first and the double-byte font second. Using the JCL CHARS parameter or the font list in a page definition, specify a single-byte font as font 0 and a double-byte font as font 1. If you do not place font 0 and font 1 in this sequence, PSF generates an error message and stops the printing process. You can also use the SOSIFONTS subcommand on the PPFA PAGEDEF or PAGEFORMAT command to ensure that a single-byte font is mapped first and a double-byte font is mapped second. See *Page Printer Formatting Aid: User's Guide*.

The following example indicates that PSF uses a particular data-scanning mode when it is printing line data. Both single-byte fonts and double-byte fonts are to be used for printing. The first font that is specified is 60D8, a single-byte font; the second is G24F, a double-byte font. The single-byte coded font is named X060D8, and the double-byte font file is named X0G24F.

**Example:** This example uses the PRMODE parameter in the OUTPUT statement to specify that the SOSI1 process mode is to be set up for a data set that is printed on a specific 3820 printer:

```
//OUT1 OUTPUT CHARS=(60D8,G24F),PRMODE=SOSI1
//DD1 DD SYSOUT=B,DESTINATION=(REMOTE1),OUTPUT=(*.OUT1)
```

#### Notes:

1. The SOSI process must be started when the printer is started. If you are using the Page-Printer Defaults form (see [Figure 34 on page 82](#)), it shows whether the process is active or not.
2. For the process to work correctly, the first font that is specified in the CHARS parameter (or in a page definition font list) must be the single-byte font, and the second font must be the double-byte font.
3. When you use the same single-byte and double-byte font pair for the entire page, IBM recommends that you do not mix the use of shift-out, shift-in processing with the use of table reference characters in line data. This recommendation is because of the rules that are used in the scanning process and because the fonts used for the shift-in and shift-out codes are always font 0 and font 1.
4. When you use the same single-byte and double-byte font pair for the entire page, IBM does not recommend mixing the use of shift-out, shift-in processing with the use of font lists in page definitions, for the reasons given in Note 1. If you follow this, you must know exactly what the data and the font list contain.
5. If your print job consists of multiple steps that alternate among PRMODE=SOSI1, PRMODE=SOSI2, PRMODE=SOSI3, and PRMODE=SOSI4, JES2 and JES3 reorder the steps to group all those with the same PRMODE value. The resulting output is different for JES2 and JES3 systems.
6. IBM recommends that shift-in codes and shift-out codes alternate in a specific record.
7. When the PRMODE parameter is specified in the OUTPUT JCL, it overrides the default process mode specified in the Printer Inventory, Exit 7, or the PRINTDEV statement.
8. When a default PRMODE is specified for both the initialization call and the begin-data-set calls (BDSC) in Exit 7, the BDSC specification overrides the initialization specification of PRMODE.

For more information about SOSI codes, including the data conversion that PSF makes for SOSI1, SOSI2, SOSI3, and SOSI4, see [“Shift-out, shift-in \(SOSI\) codes”](#) on page 74.

## Printing more than one copy

To print more than one copy of your AFP data, you can use these methods:

- You can transmit the data set to the printer any number of times, producing a copy of the entire data set for each transmission. You can specify collated copies either by using the JCL COPIES or COPYCNT parameters or by using multiple OUTPUT statements in the JCL.
- You can print multiple copies of each page in turn. You can specify page copies in the form definition, or you can code them as subgroups on the JCL COPIES parameter.

You can include copies from both methods in a single data set. For more information about these options and the relationships between them, see the descriptions of [“COPIES”](#) on page 89 and [“Using FORMDEF with COPIES or FLASH parameters in JCL”](#) on page 114.

### Examples:

1. In this example, the COPIES parameter is specified in the DD statement. Fourteen copies of the data set are to be printed in page number sequence.

```
//AFPUSERA JOB ...  
//STEP1 EXEC PGM=USERA  
//PRINT DD SYSOUT=A,COPIES=14  
/*
```

The data set is sent to the printer 14 times. If the data set contains three pages, the output is like the example in [Figure 43](#) on page 137.

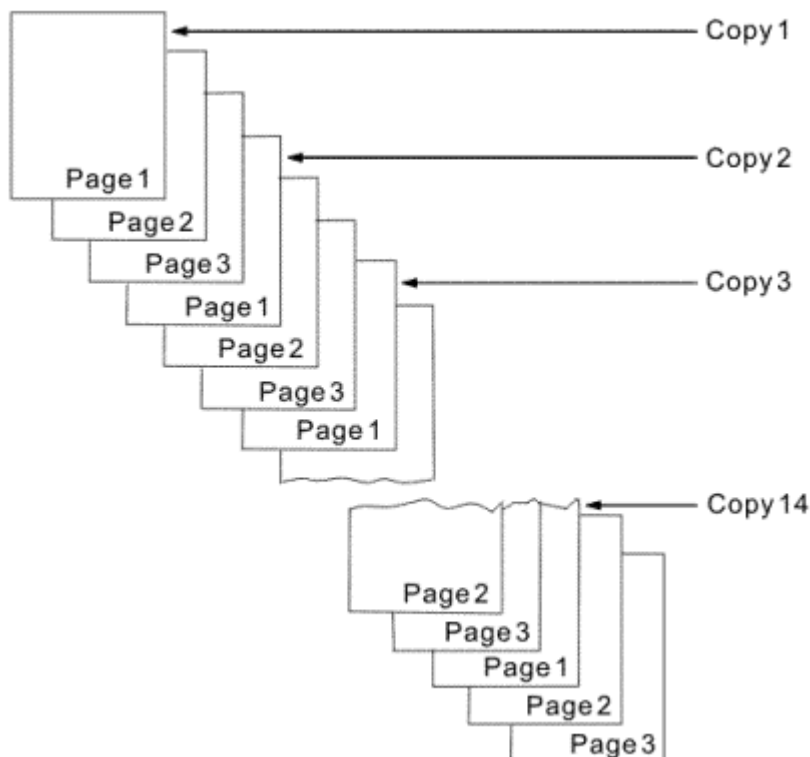


Figure 43. Sample output for COPIES=14

2. This example uses a user-created form definition, named F1UCOPY2, that specifies two copies of each page. These two copies can be defined in the form definition with different overlays, suppressions, or flash options.

```
//AFPUSERB JOB ...  
//STEP1 EXEC PGM=USERB
```

```
//OUT1 OUTPUT FORMDEF=UCOPY2
//DD1 DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

The data set is transmitted one time and contains two copies of each page.

3. This example uses both the COPIES parameter and the user form definition named F1UCOPY2.

```
//AFPUSERC JOB ...
//STEP1 EXEC PGM=USERC
//OUT1 OUTPUT FORMDEF=UCOPY2
//DD1 DD SYSOUT=A,OUTPUT=(*.OUT1),COPIES=3
/*
```

The data set is transmitted three times, as specified in the COPIES parameter. Each of those transmissions contains two copies of each page, as specified in the form definition.

4. In this example, the COPIES parameter is specified in the OUTPUT statement. Three copy groups are to be printed, producing a total of six copies of the data set. The group values are set so that the first group contains one copy of each page, the second contains three copies, and the third contains two copies.

```
//AFPUSERD JOB ...
//STEP1 EXEC PGM=USERD
//OUT2 OUTPUT COPIES=(, (1,3,2))
//DD2 DD SYSOUT=(,),OUTPUT=(*.OUT2)
```

The data set is sent to the printer three times, once for each group value. If the data set contains three pages, the output is like the example in [Figure 44](#) on [page 138](#). Notice that in the second and third copy groups, all the copies of a particular page are printed before the next page is printed.

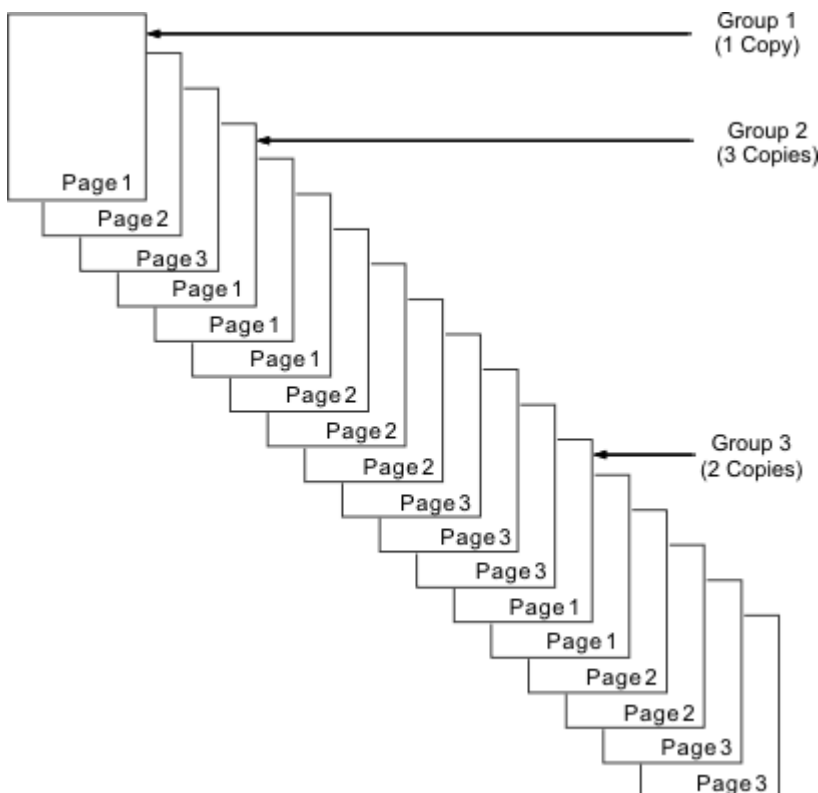


Figure 44. Sample output for COPIES=(, (1,3,2))

5. In this example, you use multiple OUTPUT statements in the JCL to print multiple copies of a data set, each with different AFP formatting.

```
//AFPUSERE JOB ...
//STEP1 EXEC PGM=USERE
//OUT1 OUTPUT PAGEDEF=USER1,FORMDEF=USER1
//OUT2 OUTPUT PAGEDEF=USER2
```

```
//DD1 DD SYSOUT=A,OUTPUT=(*.OUT1,*.OUT2)
/*
```

The job first prints the data set according to the page definition and form definition specified in the OUT1 OUTPUT statement. Then, the job prints the data set again, this time by using the page definition specified in the OUT2 OUTPUT statement. Because this statement does not specify a form definition, the default form definition is used.

## Bursting and stacking continuous-forms paper

Bursting paper means separating the continuous-forms paper into individual sheets. The 3800, 3900, or InfoPrint 4000 printer must be equipped with an optional burster-trimmer-stacker (BTS) device. If the printer does not have a BTS, the forms go to a continuous-forms stacker.

**Note:** If you are using the Page-Printer Defaults form, it indicates whether a specific continuous-forms printer has a BTS. See [Figure 83 on page 221](#).

Use the BURST parameter to specify whether you want the output paper to go to the BTS or to the continuous-forms stacker (see [“BURST” on page 86](#)). If your data set is printed on a cut-sheet-paper printer, PSF ignores the BURST parameter.

### Examples:

1. This example is for printing on a 3900 printer with a BTS installed. The BURST parameter is specified in the DD statement to indicate that you want the output burst into separate sheets:

```
//DD1 DD SYSOUT=A,DEST=PR3900,BURST=YES
```

2. In this example, the BURST parameter is specified in the OUTPUT statement:

```
//OUTPUT1 OUTPUT BURST=YES,FORMDEF=USER1
//DD1 DD SYSOUT=A,OUTPUT=(*.OUTPUT1)
```

## Specifying whether you want error messages to be printed

The PIMSG parameter specifies whether you want error messages to be printed and also specifies the maximum number of errors that can occur before printing is stopped (see [“PIMSG” on page 101](#)). The PIMSG count applies only to errors that would not, in themselves, cause the data set to stop printing. Data set printing is always stopped if an error occurs that stops processing, regardless of the setting of PIMSG.

Values for the PIMSG parameter are assigned in the PSF startup procedure. To override these values, code the PIMSG parameter in the JCL for your print job.

### Examples:

1. The first example specifies that all error messages are to be printed and that printing is to be stopped after 10 errors occur.

```
//OUT1 OUTPUT PIMSG=(YES,10)
//DD1 DD SYSOUT=A,OUTPUT=*.OUT1
```

2. For this example, the system does not print messages unless an error occurs that stops printing, and then prints only the message group caused by that error. The PIMSG count, which is not coded, defaults to 16. If 16 errors that do not stop processing occur, printing is stopped because the PIMSG count was exceeded. The only message group that is printed indicates that PIMSG count was exceeded.

```
//OUT2 OUTPUT PIMSG=NO
//DD2 DD SYSOUT=A,OUTPUT=*.OUT2
```

3. This example sets the PIMSG count to zero and specifies that no messages are to be printed unless an error that stops processing occurs so that the data set continues to print regardless of how many errors that do not stop processing occur. If an error that stops processing occurs, PSF prints the message group caused by that error.

```
//OUT3 OUTPUT PIMSG=(NO,0)
//DD3 DD SYSOUT=A,OUTPUT=*.OUT3
```

PSF issues messages to one or more of these destinations, depending on which is the most appropriate:

**Operator's console**

Messages that the operator or system programmer must address.

**Security administrator's console**

Messages reporting security violations or errors in the processing of security functions.

**Job submitter's console**

Messages reporting the completion of a job or its failure to print.

**In the printed output**

Messages reporting the position from which data is retransmitted during a recovery.

**Following the printed output or elsewhere**

Messages describing data stream errors. Also, a duplicate copy of any message issued within the printed output.

**Note:** Your system programmer might redirect the messages associated with your print job. Consult your programmer to determine where the messages for your job are sent.

## Using TrueType and OpenType fonts

---

TrueType and OpenType fonts are stored in UNIX files (HFS or zFS files). These fonts can be in the printer, inline in the print data set, in user path libraries, or in system font path libraries. TrueType and OpenType fonts are referenced in the print data set by using the Map Data Resource (MDR) structured field. PSF supports TrueType and OpenType fonts only on printers that support Unicode. See your printer documentation to determine whether your printer supports Unicode.

To use TrueType and OpenType fonts with PSF, you or the system administrator must do these steps:

1. Make sure that you have access to one of these products:

- The z/OS Font Collection, a base feature of z/OS, which contains prebuilt RATs for TrueType and OpenType fonts that are installed during the z/OS installation. For more information about the z/OS Font Collection, see [z/OS Font Collection](#).
- A product that supplies TrueType and OpenType fonts, such as WorldType Fonts for AFP Print Servers, which is an optional feature of Infoprint Fonts for Multiplatforms (Program Number 5648-E77). See *IBM Infoprint Fonts: Font Summary* for more information about Infoprint Fonts for Multiplatforms.

You also need an AFP resource installer program, such as Font Installer for AFP Systems (available as a priced feature of Infoprint Fonts for Multiplatforms), to install and build resource access tables (RATs) for the TrueType and OpenType fonts and install the fonts in path libraries.



**Attention:** PSF can successfully process RATs that are created with either the Font Installer for AFP Systems or another AFP resource installer. However, a RAT created with the Font Installer for AFP Systems is not compatible with a RAT created with an AFP resource installer. Though an AFP resource installer can process a RAT created with the Font Installer for AFP Systems and migrate it into the new format, the opposite is not true. When the Font Installer for AFP Systems processes a RAT created with an AFP resource installer, it might lose the RAT information or an error might occur. Therefore, IBM suggests that you do not use the Font Installer for AFP Systems after you create a RAT with another AFP resource installer.

2. Identify the path libraries where the fonts are found:

- Identify the system font path libraries in the PSF startup procedure with the FONTPATH parameter on the PRINTDEV statement. See [PSF for z/OS: Customization](#).
- Identify user path libraries with the USERPATH parameter on the OUTPUT JCL statement. See [“USERPATH” on page 107](#).

3. Use z/OS UNIX System Services to set access permissions to the directories and files in the user path and font path libraries. Read permissions must be set in PSF for each path directory and file it is accessing. If the permissions for each directory or file are not set correctly, PSF cannot access the path library. For more information about z/OS UNIX System Services, see *z/OS UNIX System Services Command Reference*.

For more information about TrueType and OpenType fonts, see *Using OpenType Fonts in an AFP System*.

## Processing Unicode Complex Text

---

PSF supports input data sets that contain complex text, which is Unicode-encoded text that cannot be translated with the traditional one-code-point to one-glyph method; for example, bidirectional Arabic text or combined Hindi characters. Complex text requires:

- Extra processing
- Identification with a PTOCA Unicode Complex Text or Glyph Layout Control (GLC) control sequence
- A layout engine that examines runs of code points and maps these to runs of glyph indexes and their positions
- TrueType and OpenType fonts

Font layout tables contain script-specific information about glyph substitution, glyph positioning, justification, and baseline positioning, all of which are used by the layout engine to translate complex text.

For PSF to correctly process GLC control sequences, the TrueType and OpenType fonts that are used must be placed inline in the print data set. PSF looks in the inline resource group for the font referenced in the Map Data Resources (MDR) structured field. If PSF cannot find the font inline, the complex text is not processed.

## Using extended code pages

---

Extended code pages are code pages that include multiple encodings within a single code page. Traditional code pages include EBCDIC or ASCII encoding only, but extended code pages can contain EBCDIC or ASCII encodings along with the Unicode equivalent value. Each code point can be mapped to one or more Unicode values so a printer can print with TrueType and OpenType fonts. The Code Page Control (CPC) structured field in the code page indicates whether the code page contains Unicode values. If an extended code page contains Unicode values, PSF can send the values to a printer that supports extended code pages. If a printer does not support extended code pages, the Unicode values are removed from the code page before PSF sends it to the printer.

Extended code pages can be stored in partitioned data sets (PDS or PDSE) in user or system font libraries, or in UNIX files (zFS files) in user path or system font path libraries. The recommended file name prefix for extended code pages is T1. When stored in UNIX files, extended code pages must have a .ECP file extension in uppercase format.

PSF uses these methods to access extended code pages:

- PSF uses z/OS UNIX System Services to access extended code pages that are installed in user path libraries specified in the USERPATH parameter on the OUTPUT JCL statement or in system font path libraries specified with the FONTPATH parameter on the PRINTDEV statement of the PSF startup procedure.
- PSF accesses extended code pages that are installed in PDS or PDSE libraries specified in the USERLIB parameter on the OUTPUT JCL statement or with the FONTDD parameter on the PRINTDEV statement of the PSF startup procedure.

To use extended code pages:

1. IBM provides extended code pages as .zip files. Install the extended code pages in UNIX path libraries.



2. To place extended code pages in a PDS or PDSE, run the AFRREBLK program to transfer the code pages from the UNIX files to the partitioned data set. See [Chapter 9, “Using the AFP Reblocking Program,” on page 161](#).

**Note:** Do not use file transfer protocol (FTP) to store an extended code page in a partitioned data set. You use FTP only to transfer an extended code page to a UNIX file.

3. Identify the resource libraries where the extended code pages are found:
  - For code pages in a PDS or PDSE:
    - Identify user libraries with the USERLIB parameter on the OUTPUT JCL statement. See [“USERLIB” on page 107](#).
    - Identify system font libraries in the PSF startup procedure with the FONTDD parameter on the PRINTDEV statement. See [PSF for z/OS: Customization](#).
  - For code pages in UNIX files:
    - Identify user path libraries with the USERPATH parameter on the OUTPUT JCL statement. See [“USERPATH” on page 107](#).
    - Identify the system font path libraries in the PSF startup procedure with the FONTPATH parameter on the PRINTDEV statement.
4. Use z/OS UNIX System Services to set access permissions to the directories and files in the user path and font path libraries. Read permissions must be set in PSF for each path directory and file it is accessing. If the permissions for each directory or file are not set correctly, PSF cannot access the path library. For more information about z/OS UNIX System Services, see [z/OS UNIX System Services Command Reference](#).

## Printing with resources from a user library

---

You can access any PSF resource from user libraries that are not defined to other PSF jobs for that printer. You can specify a maximum of eight user libraries in the USERLIB or USERPATH parameter of the OUTPUT statement (see [“USERLIB” on page 107](#) or [“USERPATH” on page 107](#)).

Because PSF does not retain copies of resources from user libraries between print jobs, user libraries are useful for testing new resources and for protecting secure resources.

PSF does not use resources from user libraries for security page labeling or for processing separator pages or message data sets.

If a resource in a user library is marked PUBLIC and it exists in the printer, PSF uses the resident version of the resource. To use a page definition or a form definition from a user library, you must specify the name of the page definition or form definition in the user JCL.

PSF tells the printer not to capture resources that are loaded from user libraries. If you want them made resident in the printer, then you must directly install the resources into the printer, if your printer is capable, or you must put the resources in a system library. PSF allows only PUBLIC resources from system libraries to be captured.

If you are using fonts from user libraries, the font resolution must be supported by the printer. You can look at the IM-IMAGE AND CODED-FONT RESOLUTION SELF-DEFINING FIELD in the PSF display printer information report to determine what pel resolutions the printer supports. If the printer does not support a font resolution, you must scale the font to fit the printer's supported pel resolution. For information about the display printer information function or information about how to convert fonts, see [PSF for z/OS: Customization](#). Converting fonts is a task that is typically done by a system programmer.

Object container resources and FOCA extended code pages can reside in partitioned data set (PDS or PDSE) libraries and UNIX file path libraries. PSF accesses object containers and extended code pages in user libraries specified with the USERLIB and USERPATH parameters. If both parameters are specified on the OUTPUT statement, PSF uses this order to find the resource:

1. User path libraries specified with USERPATH
2. User libraries specified with USERLIB



For the complete order that PSF uses to search for object container resources, see [“Searching for resources specified by a print job”](#) on page 14.

### Examples:

1. In this example, the USERLIB parameter in the OUTPUT statement tells PSF to search the library specified on the USERLIB statement for the specified data sets. If the data sets are not found, PSF searches the system libraries defined in the startup procedure.

```
//OUT1 OUTPUT USERLIB=('USERA.IMAGES','USER.AFP.RESOURCES')
//DD1 DD SYSOUT=A,OUTPUT=(*.OUT1)
```

2. In this example, the USERPATH parameter in the OUTPUT statement specifies the path libraries for the TrueType and OpenType fonts and extended code pages that PSF needs to print the document. If the fonts are not found, PSF searches the system libraries defined with the FONTPATH parameter in the PSF startup procedure.

```
//OUT1 OUTPUT USERPATH=('/jdoe/fonts/truetype','/jdoe/fonts/truetype/myfonts/')
//DD1 DD SYSOUT=*,OUTPUT=(*.OUT1)
```

3. This example specifies user libraries that are searched for resources before the PSF system libraries are searched. These resources are specified for the print job:

- Form definition F1USERA is in user library USERA.RESOURCE.
- Overlay O1USERA, referenced in F1USERA, is also in user library USERA.RESOURCE.
- Page definition A06462 is in a PSF system library. The page definition identifies the fonts that are needed to print a job.
- FOCA fonts are in PDS user library USERA.FONTSD and extended code pages and TrueType and OpenType fonts are in user path libraries /jdoe/fonts/truetype and /jdoe/fonts/truetype/myfonts/.
- Fonts that are referenced in the overlay are in the system library.

```
//AFPUSERA JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT FORMDEF=USERA,PAGEDEF=A06462,
//      USERLIB=('USERA.RESOURCE','USERA.FONTSD'),
//      USERPATH=('/jdoe/fonts/truetype','/jdoe/fonts/truetype/myfonts/')
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
/*
```

**Note:** You do not define the system libraries in the JCL for your print job. Your system programmer already defined system libraries to PSF for all jobs that use the printer.

## Printing with inline resources

For an inline form definition, a page definition, a color mapping table resource, or a microfilm setup resource, you can do any of these:

- Specify the resource name in a JCL parameter. See [“COLORMAP”](#) on page 88, [“COMSETUP”](#) on page 88, [“FORMDEF”](#) on page 94, or [“PAGEDEF”](#) on page 100.
- Specify the name DUMMY.
- Specify no parameters.

If you specified the inline resource name in the FORMDEF, PAGEDEF, COLORMAP, or COMSETUP parameter, and your data set contains that inline resource, PSF uses the inline resource. If the name in the FORMDEF, PAGEDEF, COLORMAP, or COMSETUP parameter does not match the name of an inline resource, PSF uses the resource from the resource library that matches the name in the JCL.

If you did not specify the FORMDEF, PAGEDEF, COLORMAP, or COMSETUP parameter, and your data set contains inline resources, PSF uses the first inline resource (FORMDEF, PAGEDEF, COLORMAP, or COMSETUP) encountered in the data set.

If you send several resources with the same name inline, PSF uses the last resource received for processing the data set.

If you use an outline font as an inline resource in your data stream, consider including the equivalent raster version of the font, to ensure that your job can be printed, in case the target printer supports only one type of font technology. For more information about font mapping and how PSF selects resources, see *PSF for z/OS: Customization*.

Keep in mind that the maximum length of any record in an inline resource is 32752. The maximum size of each inline resource that PSF can handle is about 1 GB (a gigabyte is 1073741824 bytes).

For more information about using inline resources, see “Specifying a form definition” on page 112, “Specifying a page definition” on page 121, “Specifying JCL parameters for microfilm jobs” on page 148, and “Specifying color mapping tables” on page 149.

**Example:** This example specifies two inline resources for PSF to use. PSF searches the system libraries for the other resources. These resources are specified for the print job:

- Form definition F1INFDEF is sent inline with the data set.
- Overlay O1INLOV, referenced in F1INFDEF, is also sent inline.
- Page definition P1A06462 is in a PSF system library.
- Font 60D8 is in the PSF system font library.

```
//AFPUSERB JOB ...  
//STEP1 EXEC PGM=USERB  
//OUT1 OUTPUT FORMDEF=INFDEF,PAGEDEF=A06462  
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1),CHARS=60D8  
/*
```

PSF uses the inline form definition and overlay and takes the page definition and the font from the system library. You do not need to define the system libraries in the JCL for your print job; your system programmer already defined system libraries to PSF for all the jobs that use the printer.

**Note:** XML data does not support inline resources.

## Specifying that inline resources are stored above the bar

PSF stores inline resources temporarily in virtual storage and uses them only for the data set for which they are sent inline. By default, this virtual storage is from the region area (31-bit storage). To relieve constraints on the region area, the system programmer can specify that PSF always stores inline resources above the bar (64-bit storage).

If your job contains a large number of inline resources, it might end with message APS515I, which indicates insufficient virtual storage in the region area. You can specify that the inline resources for this job are stored above the bar, which decreases region area storage requirements and might allow the job to print.

To do this, specify `above-the-bar-storage=inline-resources` in the AFPPARMS control statement on your OUTPUT JCL statement. For syntax rules, see “AFPPARMS control statement” on page 146.

```
#-----#  
# This is an example of the AFPPARMS control statement that demonstrates #  
# the syntax rules for specifying the control statement #  
#-----#  
above-the-bar-storage=inline-resources # A simple control parameter
```

Figure 45. Example of the AFPPARMS control statement

## Specifying the AFPPARMS control statement on the OUTPUT statement

The AFPPARMS control statement specifies additional parameters that PSF uses to process JES spool data sets. Specify this control statement with the AFPPARMS parameter on the OUTPUT JCL statement

at job submission. The parameters in the AFPPARMS control statement are associated with a current JES spool data set and used by PSF to process the spool data set.

## Syntax

The AFPPARMS parameter has this syntax:

AFPPARMS='dsname[(membername)]'

### **dsname**

Specifies the name of the data set that contains PSF parameters. Data set names can contain 1–44 alphanumeric (A–Z, 0–9) and national (@#\$) characters.

### **membername**

Specifies the name of a member within the data set that contains PSF parameters. Member names can contain 1–8 alphanumeric and national characters. The member name is optional, but if the member name is not specified the AFPPARMS data set must be a sequential file.

## Example

```
//OUT1 OUTPUT PAGEDEF=MYDEF,  
//      AFPPARMS='MY.PDS.PARMS(MEMBER)',  
//      USERLIB='MY.RESOURCE.DATASET'
```

## AFPPARMS data set allocation

Table 6 on page 145 shows how to allocate the AFPPARMS data set that is specified with the AFPPARMS DD name in the startup procedure or the AFPPARMS parameter on the OUTPUT JCL statement.

Table 6. Allocation of AFPPARMS data set attributes			
Attribute	Value	Type	Description
<b>DCB=DSORG=</b>	PO or PS	Required	Data set organization
<b>DSNTYPE=</b>	LIBRARY	Required	Data set defined as PDSE
<b>DCB=RECFM=</b>	xx	Required	Any value except U
<b>DCB=LRECL=</b>	nnnn	Required	Maximum bytes in the record (see Note)
<b>DISP=</b>	SHR	Required	Data set can be shared
<b>SPACE=</b>	(CYL,(nn,1,10))	Required	Direct access storage device (DASD) cylinders that are required to process data

**Note:** Any record size is valid; however, use a small record size, such as DCB=LRECL=120, to conserve DASD space.

## AFPPARMS parameter selection order

PSF selects AFPPARMS parameters in this order:

1. AFPPARMS control parameter on the OUTPUT JCL statement
2. Printer Inventory
3. FSA member name in the AFPPARMS control statement that is specified in the PSF startup procedure
4. The default member name, PSFDEF

## AFPPARMS control statement

The syntax rules for the AFPPARMS control statement are:

- A parameter is a *keyword=value* pair.
- Spaces are allowed after the keyword and before the value (on either side of the equals sign).
- Only one parameter is allowed on a line.
- A parameter can span multiple lines. For example, the keyword can be on the first line, the equals sign on the second line, and the value on the third line.
- Parameter keywords and values can be specified in upper case, lower case, or mixed case.
- Comments are delimited by the # character. PSF ignores everything on a line after this character.

## Specifying notification when the print job finishes printing

If you want PSF to send a notification message when the printer completes your job, specify the NOTIFY option in the OUTPUT statement (see “NOTIFY” on page 97).

PSF issues a print-complete message with these variables:

- JOBNAME
- STEP
- ID
- SYSTEM

If the NOTIFY message is sent to a z/OS node, the message is saved until the user logs on.

### Examples:

1. This example shows how to specify the NOTIFY parameter in an OUTPUT statement:

```
//OUTPUT1 OUTPUT NOTIFY=(DEST01.USERID1)
//DD1 DD SYSOUT=N,OUTPUT=(*.OUTPUT1)
```

2. This example specifies that a print-complete message is to be sent to four users at the same node:

```
//GO.OUTPUT1 OUTPUT NOTIFY=(DEST01.USERID1,DEST01.USERID2,
// DEST01.USERID3,DEST01.USERID4)
//GO.SYSOUT1 DD SYSOUT=N,DCB=(RECFM=VBM,BLKSIZE=8192),
// OUTPUT=*.OUTPUT1
```

3. This example specifies that a print-complete message is to be sent to 16 users at the originating node (the system that runs the print job):

```
//GO.OUTPUT1 OUTPUT NOTIFY=(USERID1,USERID2,USERID3,USERID4)
//GO.OUTPUT2 OUTPUT NOTIFY=(USERID5,USERID6,USERID7,USERID8)
//GO.OUTPUT3 OUTPUT NOTIFY=(USERID9,USERID10,USERID11,USERID12)
//GO.OUTPUT4 OUTPUT NOTIFY=(USERID13,USERID14,USERID15,USERID16)
//GO.SYSUT1 DD SYSOUT=N,DCB=(RECFM=VBM,BLKSIZE=8192),
// OUTPUT=(*.OUTPUT1,*.OUTPUT2,*.OUTPUT3,*.OUTPUT4)
```

To determine whether your installation supports the print-notify function, contact your system programmer.

## Inhibiting recovery of a print job

The normal PSF error recovery actions include resetting the printer to the point in the data path at which an exception is reported, and resending the data from this point. Any pages that are reprinted are in the portion of the printer's paper path where they might be damaged or imperfect, and they must be discarded.

The **Inhibit recovery** parameter in the Printer Inventory or the PSF Exit 7 routine can request that PSF inhibit recovery for a job. This means that PSF is not to resend any portion of the job that is already

printed. If an error does occur that requires recovery for pages that are printed, such as for a paper jam, processing of the job is stopped. This can be used to prevent the reprinting of sensitive data, such as checks. If you need this function, consult your system programmer.

## Specifying duplex-page offset

---

### Examples:

1. This example shows how to use the duplex-page offset function. The data is specified first, followed by a form definition created by using Page Printer Formatting Aid (PPFA).

```
THIS IS PRINT LINE 1
THIS IS PRINT LINE 2
.
.
THIS IS PRINT LINE 70
```

2. This example shows the form definition, which specifies that both the back and the front of the duplexed page are to be offset, possibly to accommodate hole-punching or binding.

```
FORMDEF D011
  REPLACE YES
  DUPLEX NORMAL ;
COPYGROUP CPNAME
  OFFSET .94 in 1 in .81 in 1 in ;
```

## Transmitting a data set to an IBM i System

---

You can transmit a data set to an IBM i system and print it on an IPDS printer.

The steps for setting up the IBM i system to receive and print files are:

1. Correctly define AFP printers to the IBM i system.
2. Configure the printers according to instructions in *Printer Device Programming* or IBM i configuring printing in IBM Documentation.
3. Make resources available by using these commands:
  - CRTFORMDF (form definition)
  - CRTOVL (overlays)
  - CRTPAGSEG (page segments)
  - CRTPAGDEF (page definition)
  - CRTFNTRSC (fonts)
4. Use the CRTDEVPRT or CHGDEVPRT command to specify the correct default form definition and font and to specify the printer as an AFP printer.
5. Start the printer writer to print the job.

For information about IBM i commands, see *AS/400 CL Reference* or IBM i control language (CL) in IBM Documentation.

**Example:** In this example, assume that OS400SYS is the destination name of the IBM i system. Replace USER.DATASET, \*.MYOUT, and class J with your own values.

```
//MYJOB JOB (A,BC), 'TEST',MSGCLASS=H
//INSTR PROC
//SPOOL EXEC PGM=IEBGGENER
//MYOUT OUTPUT DEST=OS400SYS.USERID,
// COPIES=1
//SYSPRINT DD SYSOUT=*
//SYSUT2 DD SYSOUT=J,OUTPUT=*.MYOUT
//SYSUT1 DD DSN=USER.DATASET,DISP=SHR
//SYSIN DD DUMMY
```

```
//      PEND
//SAMPLE EXEC PROC=INSTR
```

## Specifying JCL parameters for microfilm jobs

When you send output to a microfilm device, you can identify the microfilm setup resource to distinguish the special microfilm options. If you do not, the system uses the default COMSETUP parameter that your system programmer specified in the Printer Inventory or the PRINTDEV statement. The full name of the microfilm setup resource is coded in the COMSETUP parameter of the OUTPUT statement (see [“COMSETUP” on page 88](#)).

### Examples:

1. This JCL example shows the COMSETUP parameter that is specified in the OUTPUT statement for data sent to a microfilm device:

```
//OUTPUT1 OUTPUT DEST=REMOTE1,FORMDEF=mifdef,PAGEDEF=mipdef,
//      COMSETUP=H1MICRO
//DD1    DD  SYSOUT=q,OUTPUT=(*.OUTPUT1)
```

2. This example of JCL shows two OUTPUT statements that are specified for data sent to a microfilm device and to a printer:

```
//OUTPUT1 OUTPUT CLASS=m,FORMDEF=fdef1,PAGEDEF=A06462,
//      COMSETUP=H1MICRO
//OUTPUT2 OUTPUT CLASS=A,FORMDEF=fdef1,PAGEDEF=A06462,
//DD1    DD  SYSOUT=q,OUTPUT=(*.OUTPUT1,*.OUTPUT2)
```

3. This example of JCL shows two different FORMDEF parameters that are specified for data sent to a microfilm device and to a printer. This example specifies three printed copies and one microfilm copy:

```
//OUTPUT1 OUTPUT CLASS=m,FORMDEF=fdef1,PAGEDEF=A06462,
//      COMSETUP=H1MICRO
//OUTPUT2 OUTPUT CLASS=A,FORMDEF=A10111,PAGEDEF=A06462,
//      COPIES=3
//DD1    DD  SYSOUT=q,OUTPUT=(*.OUTPUT1,*.OUTPUT2)
```

Check with your system programmer to find out which CLASS or DEST to use for sending AFP output to the microfilm device.

The microfilm setup resource must be in a PSF system library, in a PSF user library, or inline as part of the print data set. You can create your own microfilm setup resource (see the publications that are provided with your microfilm device) or use a resource already available on your system.

See [Appendix E, “Microfilm device considerations,” on page 223](#) for more information about sending output to a microfilm device.

## Using microfilm setup resources from a user library

You can instruct PSF to select a microfilm setup resource from your user library rather than from a system library assigned to PSF. To use a microfilm setup resource from a user library:

1. Reference the user library that contains the microfilm setup resource in your JCL. For details, see [“USERLIB” on page 107](#).
2. Specify the name of the microfilm setup resource in the JCL COMSETUP parameter.

## Using inline microfilm setup resources

To use a microfilm setup resource inline:

1. Include an inline microfilm setup resource in the print data set, except in XML data.
2. If you specify the COMSETUP parameter in your JCL, the name of the inline microfilm setup resource must match the COMSETUP name that is specified in your JCL, or else you must specify COMSETUP=DUMMY in the JCL.

3. If you specify multiple inline microfilm setup resources in the print data set and you specify COMSETUP=DUMMY on the OUTPUT JCL statement, PSF uses the last inline microfilm setup resource in the print data set.
4. If you do not specify the COMSETUP parameter in your JCL, PSF selects the first inline COMSETUP in the print data set.
5. If a microfilm setup resource is included inline with the data, the data set must be identified as containing carriage control characters. If the length of the records in the microfilm setup resource is less than or equal to the logical-record length defined for the data set, you can specify fixed-length records for the record format. If the length of the records in the microfilm setup resource is greater than the logical-record length defined for the data set, you must specify variable-length records variable-blocked with ANSI carriage control characters (VBA) or variable-blocked with machine carriage control characters (VBM) for the record format.

You can include more than one inline microfilm setup resource in a print data set, and you can change the microfilm setup resource name in the JCL on different printing jobs to test different microfilm setup resources. However, if the name of an inline microfilm setup resource does not match the COMSETUP name that is specified in the JCL, PSF uses the microfilm setup resource from the resource library that matches the name in the JCL.

## Specifying color mapping tables

When you send output to a printer that supports the color mapping table resource, you can specify a color mapping table to map color translations to the printer. The full name of the color mapping table is coded in the COLORMAP parameter of the OUTPUT JCL statement (see [“COLORMAP” on page 88](#)).

If you do not code a COLORMAP parameter on your OUTPUT JCL statement, PSF uses the default parameter that your system programmer specified in the Printer Inventory or the PRINTDEV JCL statement. If no system default is specified and a color mapping table resource is not coded inline in the print data set, PSF uses the value M1RESET, which is a reset color mapping table resource that is provided internally with PSF. For information about the reset color mapping table resource, see *Mixed Object Document Content Architecture Reference*.

**Example:** This example of JCL shows the COLORMAP parameter that is specified in the OUTPUT JCL statement for data to be printed on a printer that supports color mapping tables. In this example, the COLORMAP member name, M1COLOR, is in a PSF system object container library:

```
//OUTPUT1 OUTPUT DEST=REMOTE1,FORMDEF=A10111,PAGEDEF=A06462,
//          COLORMAP=M1COLOR
//DD1      DD SYSOUT=q,OUTPUT=(*.OUTPUT1)
```

Check with your system programmer to find out which DEST to use for sending print data sets to a printer that supports color mapping tables.

You can create your own color mapping table by using the Color Mapping Tool that is included with PSF (see [“Creating color mapping tables” on page 156](#)) or you can use an existing resource created by your system programmer. The color mapping table resource that is specified on the COLORMAP parameter must be stored in one of these places:

- In a PSF system object container library
- In a user library specified by the USERLIB parameter
- Inline in the print data set (except in XML data)

## Using Color Mapping Table Resources from a User Library

You can instruct PSF to select a color mapping table resource from a user library rather than from a system library assigned to PSF. To use a color mapping table resource from a user library:

1. Reference the user library that contains the color mapping table resource in the OUTPUT JCL statement. For details, see [“USERLIB” on page 107](#).
2. Specify the name of the color mapping table resource in the JCL COLORMAP parameter.



## Using Inline Color Mapping Table Resources

To use a color mapping table resource inline:

1. Include an inline color mapping table resource in the print data set.
2. If you specify the COLORMAP parameter in your JCL, the name of the inline color mapping table resource must match the COLORMAP name that is specified in the JCL, or else you must specify COLORMAP=DUMMY in the JCL.
3. If you specify multiple inline color mapping table resources in the print data set and you specify COLORMAP=DUMMY on the OUTPUT JCL statement, PSF uses the last inline color mapping table resource in the print data set.
4. If you do not specify the COLORMAP parameter in your JCL, PSF selects the first inline COLORMAP from the print data set.
5. If a color mapping table resource is included inline with the data, the data set must be identified as containing carriage control characters. If the length of the records in the color mapping table resource is less than or equal to the logical-record length defined for the data set, you can specify fixed-length records for the record format. If the length of the records in the color mapping table resource is greater than the logical-record length defined for the data set, you must specify variable-length records. For variable-blocked records with machine carriage control characters, use VBM for the record format.

You can include more than one inline color mapping table resource in a print data set and you can change the name in the COLORMAP parameter in the JCL on different printing jobs to test different color mapping table resources. However, if the name that is specified in the COLORMAP parameter in the JCL does not match any names of inline color mapping tables resources, PSF looks for the color mapping table resource in the user library or in the object container library.

## Specifying object container libraries in UNIX files

---

Data object resources, including color management resources (CMRs), that are installed with resource access tables (RATs) are stored in the printer, inline in the print data set, or in object container path libraries in UNIX files (HFS or zFS files). PSF uses z/OS UNIX System Services to access data object resources in path libraries specified in the OBJCPATH parameter on the PRINTDEV statement of the PSF startup procedure or the USERPATH parameter on the OUTPUT JCL statement.

**Note:** PSF supports the OBJCPATH parameter only in deferred-printing mode and on printers that support Unicode. See your printer documentation to determine whether your printer supports Unicode.

To specify object container path libraries for RAT-installed data object resources, you or the system administrator must do these steps:

1. Make sure that you have access to one of these products:
  - The z/OS Font Collection, a base feature of z/OS, which contains prebuilt RATs for TrueType and OpenType fonts that are installed during the z/OS installation. For more information about the z/OS Font Collection, see *z/OS Font Collection*.
  - A resource installer program to install data object resources in path libraries. The resource installer creates RATs, in the appropriate resource directories, that map the data objects to a path and file name that PSF can access.

**Note:** Be sure that the resource installer is set up with the correct permissions to access the path libraries.
2. Identify the path libraries where the resources are found:
  - Identify the system object container path libraries in the PSF startup procedure with the OBJCPATH parameter on the PRINTDEV statement. See *PSF for z/OS: Customization*.
  - Identify user path libraries with the USERPATH parameter on the OUTPUT JCL statement. See *"USERPATH"* on page 107.
3. Use z/OS UNIX System Services to set access permissions to the directories and files in the user path and object container path libraries. PSF must have read permissions set for each path directory



and file it is accessing. If the permissions for each directory or file are not set correctly, PSF cannot access the path library. For more information about z/OS UNIX System Services, see [z/OS UNIX System Services Command Reference](#).

For information about CMRs and color printing, see [Appendix F, “Color and grayscale printing,”](#) on page 225.

## Finishing your output

You can specify a form definition that requests specific finishing functions if you are sending output to an AFP printer with finishing capabilities (for example, an InfoPrint 60 with the finisher feature).

**Note:** Suppressing interrupt message pages is suggested when PSF is attached to a printer with finisher capability; otherwise, interrupt message pages separate finished documents into two finished groups with the inserted page in the middle.

### Examples:

1. This example shows a form definition that requests that each document within a data set is stapled in the upper left corner.

```
FORMDEF STAPLE REPLACE YES
FINISH SCOPE ALL OPERATION CORNER REFERENCE TOPLEFT
BIN 1 DUPLEX NO
OFFSET 0 0 ;
```

For more information about coding finishing in your form definitions, see [Page Printer Formatting Aid: User's Guide](#).

2. This example shows the JCL used to call the form definition with the finishing function.

```
//AFPUSER JOB ...
//STEP1 EXEC PGM=USERA
//OUT1 OUTPUT FORMDEF=STAPLE
//PRINT DD SYSOUT=A,OUTPUT=(*.OUT1)
```

For more information about form definitions with the finishing function, see [Appendix A, “Form definitions supplied with PSF,”](#) on page 175.

You can also use a form definition to specify finishing functions on a copy group. These finishing functions apply only to the copy group in which they are specified.

3. This example shows a copy group in a form definition that requests that the data set is stapled in the upper left corner.

```
FORMDEF MEDC8 REPLACE YES
      OFFSET .5 in .5 in
      PRESENT PORTRAIT
      DIRECTION ACROSS;
COPYGROUP STAPLES
      FINISH SCOPE BEGCOLL OPERATION CORNER;
COPYGROUP STAPLEC
      FINISH SCOPE CONTCOLL OPERATION CORNER;
```

**Note:** When the finisher is installed, and the form definition requests a finishing option, the InfoPrint 60 uses the appropriate output bin by default; therefore, you do not need to specify the output bin when you submit the print job.

It is also possible for you to configure PSF to finish the entire job, including the header, data set separator, message, and trailer pages. This function is called *print job finishing* and is set up by your system programmer.

See [PSF for z/OS: Customization](#) for more information about print job finishing and interrupt message pages.

## Printing on printers that support multiple resolutions

---

When you are printing on a printer that can print at more than one resolution (pel density), you might see fidelity imperfections in your printed output. Fidelity problems arise when your data was formatted with resources (for example, fonts) at one resolution, but the printer used resources at a different resolution to print the job. Some fidelity problems that can occur are:

- Data no longer exactly fits in a box on a form.
- Columns for tabular data overlap or are spaced wider than intended.
- Right-aligned data might show a ragged right margin.
- Minor typeface differences might occur.

To avoid fidelity problems in such cases, your system programmer can define separate resource libraries for each resolution that the printer supports. For example, your system programmer can define 240-pel font, overlay, and page-segment libraries and 300-pel font, overlay, and page-segment libraries. When PSF is printing on printers that support multiple resolutions (that is, printers printing in automatic mode), PSF can select resources from the appropriate resource library.

To select the correct resource library, PSF must know the resolution of the resources that you used when you formatted your data. The format resolution can be specified in several ways:

- Your system programmer can use the PSF installation exit APSUX07 to specify a format resolution.
- For fonts mapped in MO:DCA-P data, the application can specify the Font Resolution and Metric Technology triplet (X'84') on the MCF2 structured field directly in the print data set or resource to indicate the resolution. See *Mixed Object Document Content Architecture Reference* for more information about this triplet. Two applications that generate this triplet are DCF and OGL. See the documentation for your application for more information.

PSF uses the resolution in the MCF2 structured field to select the font library only; PSF does not use this resolution to select the overlay or page segment library.

- You can specify the RESFMT keyword on the OUTPUT statement (see “RESFMT” on page 103).

PSF uses the resolution format that is specified in the RESFMT parameter to select the overlay and page segment libraries. PSF used the RESFMT parameter to select the font library only if the MCF2 structured field does not specify a font resolution.

- PSF can use the default system library.

If the printer is in automatic mode, which means it is capable of accepting resources at multiple resolutions, PSF looks for the first specification of the format resolution in these areas and order:

1. The PSF installation Exit 7 for Resource Management begin-data-set call
2. The Printer Inventory or the PSF installation Exit 7 for Resource Management initialization call
3. The data stream in the Map Coded Font format 2 (MCF2) structured fields in the Font Resolution and Metric Technology triplet (X'84')
4. The JCL OUTPUT statement RESFMT keyword

If the printer is not in automatic mode, which means it is accepting resources at a single resolution only, PSF uses the resolution reported by the printer to select the resource libraries.

PSF uses the determined format resolution only when it is selecting a system library to search for resources. If inline resources exist or if you specify a user library, PSF searches them first.

Whether you specify the format resolution to PSF or not, you can control what action PSF takes if PSF finds a mismatch between the resolution of the fonts that are used to format the data and the resolution of the fonts that are used to print the data. A font-resolution mismatch might occur if, for example, the printer is not printing in automatic mode or the system programmer did not set up separate resource libraries for each resolution. If a font-resolution mismatch occurs, the default action is that PSF continues processing the data set.

**Example:** In this example, a print data set that was formatted and an overlay that was created by using 240-pel fonts are to be printed on an InfoPrint 4000 printer, which can print in automatic mode. Automatic mode means that the printer can accept resources at multiple resolutions. Assume that your system programmer defined a 240-pel font library and a 240-pel overlay library. Because font fidelity is important to this application, you want PSF to stop processing the page if PSF finds a font-resolution mismatch.

This example specifies these resources and parameters:

- Font GT10, which resides in the 240-pel PSF system font library.
- An overlay that is named O1FOVLY, which resides in the 240-pel PSF system overlay library. (This overlay is to be placed on the front side of each sheet.)
- A form definition that is named F1STOP, which requests that PSF stop processing if the fonts are not 240-pel fonts.
- A resolution of 240-pels in the RESFMT parameter, which tells PSF to use resources (fonts and overlays) from the 240-pel resource libraries.

```
//AFPUSEB JOB ...  
//STEP1 EXEC PGM=USERA  
//OUT1 OUTPUT CHARS=GT10,RESFMT=P240,OVERLAYF=O1FOVLY,FORMDEF=STOP,PAGEDEF=A06462  
//PRINT DD SYSOUT=A,OUTPUT=(*,OUT1)
```

For a complete description of how to handle printing on printers that support multiple resolutions, including a description of the system programmer tasks that are involved, see [PSF for z/OS: Customization](#).

## Specifying printer checkpoints

Use the CKPTPAGE or CKPTSEC parameter on the OUTPUT statement (see “CKPTPAGE” on page 87 and “CKPTSEC” on page 87) to specify when you want the checkpoint data recorded.

### Examples:

1. This example specifies that the printer checkpoint is at 20 pages. For printing a large data set, see “Considerations”.

```
//OUT2 OUTPUT CKPTPAGE=20  
//DD2 DD SYSOUT=class,OUTPUT=(*.OUT2)
```

2. This example specifies that the printer checkpoint is at 120 seconds:

```
//OUT1 OUTPUT CKPTSEC=120  
//DD1 DD SYSOUT=class,OUTPUT=(*.OUT1)
```

### Considerations:

1. CKPTPAGE and CKPTSEC do not apply to direct-printing mode.
2. Do not specify both CKPTPAGE and CKPTSEC. If you do, the parameter that is used depends on the defaults for your installation. See the JCL reference publication for your operating system.
3. If you do not specify the CKPTPAGE or the CKPTSEC parameter, JES uses the system default. If you are using the “Page-Printer Defaults” form, the default checkpoint values for the system are shown. See Appendix D, “Page-printer defaults form,” on page 221 for a blank form. If no system default is given, PSF does not record checkpoints.
4. PSF takes internal checkpoints at the specified intervals and transmits a request to JES to record the checkpoint data it gathers. The operator can use these internal checkpoints when the commands are issued to the printer.
5. A smaller checkpoint interval causes more internal checkpoints to be taken during the processing of a data set. This enables PSF to find the target of the operator command with less processing usage. However, too small a checkpoint interval can cause excessive use of virtual storage. Be aware of the need to balance usage between processing operator commands and using virtual storage.

6. If a system failure occurs, PSF does not guarantee that checkpoint information is used to restart the data set.
7. When the PSF attachment is Systems Network Architecture (SNA) or TCP/IP, the checkpoint interval specifies the maximum number of pages or amount of time after which PSF requests an acknowledgment from the printer. If acknowledgments are requested too often, printing performance can be adversely affected.
8. When the PSF attachment is SNA or TCP/IP and after a session is ended, PSF restarts printing from the most recent checkpoint. Therefore, more frequent checkpoints can reduce the number of pages reprinted after restart.

---

## Chapter 8. Using color mapping tables

With PSF, you can use color mapping tables for printers that support them. A color mapping table is a printer resource object that defines a translation from certain MO:DCA structured fields to new color structured fields used by printers. You can define translations from non-color fields to color, old color fields to new color fields, and from new color fields to different new color fields. Therefore, you can use existing applications and documents with new color fields without having to change the documents or applications. Also, you can use various color mappings with a single document to print the document by using color in different ways without changing the original document.

**Note:** When you are using both color mapping tables and color management resources (CMRs) for a job, the color mapping is applied first and then the CMRs are applied to the mapped colors.

---

### Understanding color mapping tables

A color mapping table consists of a base part, a set of source groups, and a set of target groups. The base part identifies the color mapping table with a type of reset or normal.

The simplest possible color mapping table is a reset color mapping table, which tells the printer to do no transformations on the color information found in the document. A reset color mapping table has no source or target groups; all other color mapping tables have at least one source and one target group.

See *PSF for z/OS: Customization* for information about color devices.

### Source groups

Each source group has an identification (ID) number, which is used to match the source group with a corresponding target group. You can have unique ID numbers; however, if you create the color mapping table without the Color Mapping Tool and want to map several source groups to a single target group, you can use the same ID number on multiple source groups. Valid ID values are 1 - 127.

**Note:** The Color Mapping Tool generates pairs of sources and targets and assigns increasing, sequential IDs to each pair. For more information about the Color Mapping Tool, see [“Creating color mapping tables” on page 156](#).

Each source group must be classified as one of these color spaces:

#### Highlight color

Highlight color is used when your existing documents describe color in terms of the percent to be covered and the percent to be shaded for a color number. The colors are device-dependent. For example, you might have a printer that allows the use of three colors for highlighting. You can specify percent coverage and percent shading for colors 1, 2, or 3. Your printer setup determines what those colors would be.

#### Standard Object Content Architecture (OCA)

Standard OCA uses defined combinations of red, green, and blue to create: blue, red, pink/magenta, green, turquoise/cyan, and yellow. Standard OCA also defines several defaults such as white on a black medium, black on a white medium, and the same color as the medium. The medium might be, for example, paper or a display.

#### GOCA pattern fill

GOCA pattern fill defines patterns for filling areas that you might want to map to colors with a color mapping table.

See *Mixed Object Document Content Architecture Reference* for detailed explanations about these color spaces.

You can use a color mapping table to pick specific object types to map, such as:

- Object area

- IM image data
- PTOCA data
- Page presentation space
- GOCA data
- Overlay presentation space
- BCOCA data
- IOCA data (bi-level, FS10)
- All PTOCA, GOCA, BCOCA, IOCA FS10, and IM object data
- All objects, object areas, and presentation spaces

After you pick the color spaces and object types you want to map, you can specify exact values for the fields you want to map.

## Target groups

If you create the color mapping table without the Color Mapping Tool, you must specify a target group ID number to match the source groups for a target. Valid ID values are 1 - 127.

**Note:** The color mapping table tool generates pairs of sources and targets and assigns increasing, sequential IDs to each pair. For more information about the Color Mapping Tool, see [“Creating color mapping tables” on page 156](#).

Each target group must be classified as a color space, which is the kind of color you want as output. The color spaces that are defined in the matching source groups are transformed to this color space. The color mapping table allows RGB, CMYK, highlight, and CIELAB, but your actual hardware determines your available choices. For example, if your printer supports only highlight color, then your target group must use highlight color.

You can specify exact values for output colors. For example, if your printer supports three highlight colors, you can specify colors 1, 2, or 3 with coverage and shading percentages that are supported by your printer.

## Creating color mapping tables

---

You can create a color mapping table on z/OS with the Color Mapping Tool.

## Color Mapping Tool components

This tool is supplied with PSF and includes:

### **APSRGMT**

Load module for the color mapping table in SYS1.LINKLIB

### **APSRRCMT**

Sample JCL to run the Color Mapping Tool in SYS1.SAMPLIB

### **APSRSCMT**

Sample input source file for the color mapping table in SYS1.SAMPLIB

## Using the Color Mapping Tool

To use the Color Mapping Tool to create a color mapping table:

1. Copy the sample input source file (APSRSCMT) and edit it to reflect your color mapping needs (see [Figure 47 on page 158](#) - [Figure 49 on page 160](#)).
2. Copy the sample JCL (APSRRCMT) to call the Color Mapping Tool (see [Figure 46 on page 157](#)) and make these changes:
  - Change the JOB card to reflect your ID and system information.

- Change the STDIN statement to point to the input source data set for the color mapping table.
- Change the STDOUT statement to point to the output data set in the PSF object container library.

**Note:** The PARM statement is case-sensitive; therefore, lowercase and uppercase characters in the PARM statement are important.

```
//APSRRCMT JOB 'account #','name',MSGLEVEL=(1,1)
//**START OF SPECIFICATIONS*****/
//*
//*      NAME: APSRRCMT
//*
//*      DESCRIPTIVE NAME: JCL to run the Color Mapping Table Tool
//*
//*      FUNCTION: This JCL will invoke Color Mapping Table Tool.
//*
//*      Licensed Materials - Property of IBM
//*      5655-B17
//*      (C) Copyright IBM Corp. 1999, 2000
//*
//*      NOTES:
//*      - Change the JOB card to meet installation requirements
//*      - Replace 'hlq' with your high-level qualifier
//*      - Replace 'vvvvvv' with the volume serial number for
//*        the output data set
//*      - Note that the PARM field is case-sensitive:
//*        lower and upper case characters are important
//*
//*      CHANGE ACTIVITY:
//*      $00=LAPS0008,HPRF320,000425,BDKULMM: Initial version
//**END OF SPECIFICATIONS*****/
//*
//CMT      EXEC PGM=APSRRCMT,REGION=4M,
//          PARM='/-i //DD:STDIN -o //DD:STDOUT'
//STDIN    DD DSN=hlq.MYCMT.SOURCE(M1COLOR),DISP=SHR
//STDOUT   DD DSN=hlq.CMT.AFPDS(M1COLOR),
//          DISP=(NEW,CATLG),
//          DCB=(RECFM=VBM,LRECL=8205,BLKSIZE=8209),
//          UNIT=SYSALLDA,VOL=SER=vvvvvv,SPACE=(TRK,(5,1,2))
//SYSUDUMP DD SYSOUT=*
```

Figure 46. Sample JCL for Color Mapping Tool

3. Submit the APSRRCMT JCL. The Color Mapping Tool reads the source statements and generates a color mapping table object data set.
4. In order for PSF to find the new color mapping table object, you must do one of these:
  - Ask your system programmer to put the new color mapping table object in an existing object container library that the PSF startup procedure already uses.
  - Put the new color mapping table object in a new partitioned data set. Then, ask your system programmer to add the partitioned data set to the list of object container libraries in the startup procedure.
  - Specify the new color mapping table data set on the USERLIB parameter in the OUTPUT JCL statement of your print-submission job request.
  - Code the color mapping table inline in the print data set.
5. Specify your color mapping table object name on the COLORMAP parameter in the OUTPUT JCL statement of your print-submission job request.

## Sample color mapping table source file

Figure 47 on page 158 - Figure 49 on page 160 show a sample input file that the Color Mapping Tool uses as the source file for the color mapping table.

```

#-----
#
# Licensed Materials - Property of IBM
# 5655-B17
# (C) Copyright IBM Corp. 1999,2000
#
# This file is a Color Mapping Table source file. One uses it to
# create CMT mappings from source colors to target colors with the
# APSRCMT utility.
#
# Attributes appear in Attribute/Values pairs. If you choose the
# default value, do not provide the Attribute (Comment it out by
# placing the # character in front of the Attribute).
# Unnecessary values following an attribute will be ignored.
#
# Attribute values are not case-sensitive.
#
# This example is the same color mapping table that
# a highlight color printer comes up with after cycling power.
#-----

#-----
# BeginMappingDef:
# Required, starts a Color Mapping Definition.
# One definition for each Source to Target mapping
#-----
BeginMappingDef:

#-----
# BeginSourceDef:
# Required, starts the Source Parameters
#-----
BeginSourceDef:

#-----
# ColorSpace:
# Required, values = OCA | Highlight | GOCA
#-----
ColorSpace: OCA

#-----
# ColorValue:
# Required, values depend on Color Space, see manual page for cmt
#-----
ColorValue: BLUE

```

*Figure 47. Sample color mapping table source file (Part 1 of 3)*



```

#-----
# ObjectType:
# Optional, values = ObjArea | ImageData | IocaData | PTOCData |
#   GOCADData | BCOCADData | Al1OCA | Page | Overlay | ObjsAll
# default = ObjsAll
#-----
#ObjectType:

#-----
# PercentShading:
# Optional, only valid for SourceColorSpace: Highlight,
# values = 0 .. 100, 255 (all percentages), default = 100
#-----
#PercentShading:

#-----
# PercentCoverage:
# Optional, only valid for SourceColorSpace: Highlight,
# values = 0 .. 100, 255 (all percentages), default = 100
#-----
#PercentCoverage:

#-----
# EndSourceDef:
# Required, ends the Source Parameters
#-----
EndSourceDef:

#-----
# BeginTargetDef:
# Required, starts the Target Parameters
#-----
BeginTargetDef:

#-----
# ColorSpace:
# Required, values = RGB | CMYK | Highlight | CIELAB
#-----
ColorSpace: Highlight

#-----
# ColorValue:
# Required, values depend on Color Space, see manual page for cmt
#-----
ColorValue: 1

#-----
# PercentShading:
# Optional, only valid for TargetColorSpace: Highlight,
# values = 0 .. 100, default = 100
#-----
#PercentShading:

```

Figure 48. Sample color mapping table source file (Part 2 of 3)

```

#-----
# PercentCoverage:
# Optional, only valid for TargetColorSpace: Highlight,
# values = 0 .. 100, default = 100
#-----
#PercentCoverage:

#-----
# EndsTargetDef:
# Required, ends the Target Parameters
#-----
EndTargetDef:

#-----
# EndMappingDef:
# Required, ends a Color Mapping Definition
#-----
EndMappingDef:

#
# Map OCA RED to Highlight 2
#
BeginMappingDef:
  BeginSourceDef:
    ColorSpace:   OCA
    ColorValue:   RED
  EndSourceDef:
  BeginTargetDef:
    ColorSpace:   Highlight
    ColorValue:   2
  EndTargetDef:
EndMappingDef:

#
# Map OCA PINK to Highlight 3
#
BeginMappingDef:
  BeginSourceDef:
    ColorSpace:   OCA
    ColorValue:   PINK
  EndSourceDef:
  BeginTargetDef:
    ColorSpace:   Highlight
    ColorValue:   3
  EndTargetDef:
EndMappingDef:

```

*Figure 49. Sample color mapping table source file (Part 3 of 3)*

---

## Chapter 9. Using the AFP Reblocking Program

When you transfer AFP files from your workstation to the z/OS host, the file transfer program blocks the data records so all record lengths are equal. Because PSF for z/OS cannot process AFP files in this format, it cannot send them to the printer. However, a program that is included with PSF can convert the records to a format that can be used by PSF. This program, the AFP Reblocking Program (AFRREBLK), reblocks the AFP file data that is transferred from your workstation to the z/OS system. You can then print this reblocked data in the same way you print other z/OS documents.

The AFP Reblocking Program includes:

### **AFRREBLK**

A profile that the user can optionally set up to specify the default naming conventions that are used when an AFP file is reblocked. Found in the SYS1.SAMPLIB library.

### **AFRREBLK**

The exec that reblocks AFP files. Found in the SYS1.SAPSEXEC library.

### **AFRREMSG**

The exec that provides status and error messages. Found in the SYS1.SAPSEXEC library.

This information describes:

- “How the AFP Reblocking Program works” on page 161
- “Setting up the AFRREBLK profile” on page 162
- “Uploading AFP files to z/OS” on page 164
- “Reblocking data sets on z/OS” on page 164

---

## How the AFP Reblocking Program works

To reblock AFP file data, you use the AFRREBLK command, which specifies an input data set or a z/OS File System (zFS/HFS) file and, optionally, an output data set. If the output data set is not specified on the AFRREBLK command, the reblocking program determines the type of AFP object that is to be reblocked, such as overlay or object container, and then creates a reblocked data set in one of these ways:

- The reblocking program uses the AFRREBLK profile to select the data set name for the AFP object type. For example, this line in the AFRREBLK profile (see Figure 50 on page 163),

```
Doc240 userid().afirreblk.doc240          /* 240 pel Document Names */
```

indicates that when a document formatted for a 240 pel resolution printer is encountered, the reblocked file becomes a member of the partitioned data set *userid.afirreblk.doc240*, where *userid* is your user ID. Therefore, when AFRREBLK reblocks the input data set *userid.GRTNOVEL.UPLD*, it recognizes the file as a 240-pel document from fields in the AFP data and changes its name, according to the AFRREBLK profile, to *userid.afirreblk.doc240(GRTNOVEL)*. The second qualifier of the input data set, GRTNOVEL, becomes the member name of the output data set.

The profile can also specify a sequential data set by using only one qualifier. For example, if the profile specifies:

```
Doc240 doc240                             /* 240 pel Document Names */
```

the file that is reblocked is renamed *userid.dsname.DOC240*, where *dsname* is the second qualifier of the input data set. Therefore, the input data set, *userid.GRTNOVEL.UPLD*, is reblocked with the name *userid.GRTNOVEL.DOC240*.

- If no profile exists or if no data set can be determined from the profile, the reblocking program allocates a sequential output data set with a first qualifier of your user ID, the same second qualifier as the input data set, and a third qualifier determined by the AFP object type. For example:

`userid.dsname.objtype`

Table 7 on page 162 shows the third qualifier values that the reblocking program associates with the default AFP object types. The list also shows the profile keywords associated with the AFP object types.

Table 7. Data set values for default AFP object types		
AFP Object Type	Third Qualifier	Profile Keyword
Unbounded Box Font	FONT3820	FontUB
240 Bounded Box Font	FONT3820	Font240
300 Bounded Box Font Name	FONT300	Font300
Outline Font Name	FONTOLN	FontOLN
Page Definition Name	PDEF38PP	PageDef
Form Definition Name	FDEF38PP	FormDef
240 pel Document Name	LIST3820	Doc240
Resolution Independent Document	LISTAFP	DocAFP
240 pel Overlay	OVLY38PP	Ovly240
Resolution Independent Overlay	OVLYAFP	OvlyAFP
IOCA Image	IOCA	IOCA
Object Container	AFPOBJ	AFPOBJ
Page Segment-240 pel IM1 Image	PSEG3820	Pseg240
Page Segment-300 pel IM1 Image	PSEGAFF	PsegAFP

Therefore, when AFRREBLK reblocks the input data set `userid.OVERLAY.UPLD`, it recognizes the file as a 240-pel overlay from fields in the AFP data and changes its name, according to the default AFP object type list, to `userid.OVERLAY.OVLY38PP`.

- If the reblocking program does not recognize the AFP object type, it sets the third qualifier in the sequential output data set to AFPDS. For example, if the type of AFP data set is not recognized, the reblocking program uses:

`userid.dsname.AFPDS`

where `userid` is the user identifier of the person who is reblocking the document and `dsname` is the second qualifier of the input data set being reblocked. Therefore, if `userid.UNKNOWN.UPLD` is reblocked and not recognized by the profile or in the default AFP object type list, the data set is named `userid.UNKNOWN.AFPDS`. The second qualifier (UNKNOWN) is the same as the data set being reblocked. The third qualifier is always AFPDS.

- When the input is a zFS/HFS file, the AFRREBLK program follows the same rules as when the input is a data set, except for the naming of the second qualifier of the output data set. The second qualifier is the first 8 characters of the zFS/HFS filename, after removing underscore characters (or all remaining characters, if there are fewer than 8). The input filename extension suffix is not included. For example, if `/usr/lpp/PSF/fonts/cdepage/flnm1_nm2.ECP` is reblocked and not recognized by the profile or in the default AFP object type list, the output data set is named `userid.FLNM1NM2.AFPDS`. The third qualifier is always AFPDS.

## Setting up the AFRREBLK profile

To set up an optional AFRREBLK profile, you copy the sample profile in `SYS1.SAMPLIB` (AFRREBLK) to a data set named `userid.AFRREBLK.PROFILE`, where `userid` is your user ID.

The AFRREBLK profile includes a list of possible AFP object types that the reblocking program can recognize. You can set up the AFRREBLK profile to include the default data set names that you want. To specify a partitioned data set, use the high-level qualifier and the data set name. To specify a sequential data set, use only one qualifier. If you specify the name as a partitioned data set, you must preallocate the data set. For a sequential data set, the one qualifier that is specified is used as the third qualifier in the data set name. [Figure 50 on page 163](#) shows the sample AFRREBLK profile that is included with PSF.

```

/*****
/* Function: AFRREBLK MVS Profile - used when the output file is          */
/*          not specified. The program will determine the type           */
/*          of AFPDS object and store it to the specified                 */
/*          location.                                                     */
/*          */
/* Syntax   : Keyword DatasetName                                         */
/*          */
/*          The DatasetName may be a partition dataset name or a          */
/*          single name that will be used to create a sequential          */
/*          dataset. Partition dataset names may use 'userid()'           */
/*          as a high level qualifier and will be resolved to             */
/*          the userid of the person running the program.                 */
/*          Partition datasets must be preallocated.                     */
*****/

FontUB   userid().afrrreblk.fontub           /* Unbounded-Box Font Names      */
Font240  userid().afrrreblk.font240          /* 240 Bounded-Box Font Names   */
Font300  userid().afrrreblk.font300          /* 300 Bounded-Box Font Names   */
FontOLN  userid().afrrreblk.fontOLN          /* AFP Outline Font Names       */

PageDef  userid().afrrreblk.pagedef          /* Page Definition Names        */
FormDef  userid().afrrreblk.formdef          /* Form Definition Names        */

Doc240   userid().afrrreblk.doc240           /* 240 pel Document Names      */
DocAFP   userid().afrrreblk.docafp          /* Resolution Independent Doc   */

Ovly240  userid().afrrreblk.ovly240          /* 240 pel Overlays            */
OvlyAFP  userid().afrrreblk.ovlyafp          /* Resolution Independent Ovly  */

IOCA     userid().afrrreblk.ioca             /* IOCA Images                  */
PageSeg  userid().afrrreblk.pageseg         /* IM1 Image - Page Segments   */

Pseg240  userid().afrrreblk.pseg240          /* 240 Pel Image Descriptor     */
PsegAFP  userid().afrrreblk.psegafp         /* 300 Pel Image descriptor     */
AFP0BJ   userid().afrrreblk.afpobj          /* Object Container             */

LRECL    32756                             /* Record Length                */
BLKSIZE  32760                             /* Blocksize                    */

```

*Figure 50. Sample AFRREBLK profile*

Each line of the AFRREBLK profile contains one keyword and its value, which are separated by one or more spaces. Any line with an unrecognized keyword is ignored. You can enter keywords in uppercase or lowercase. Valid keywords are:

#### **FONTUB**

Library for unbounded-box font objects.

#### **FONT240**

Library for 240-pel font objects.

#### **FONT300**

Library for 300-pel font objects.

#### **FONTOLN**

Library for AFP outline font objects.

#### **PAGEDEF**

Library for page definitions.

#### **FORMDEF**

Library for form definitions.

#### **DOC240**

Library for 240-pel documents.

**DOCAFP**

Library for resolution-independent documents.

**OVLY240**

Library for 240-pel overlays.

**OVLYAFP**

Library for resolution-independent overlays.

**IOCA**

Library for IOCA images.

**AFPOBJ**

Library for color mapping table object container and wrapped EPS object container.

**PAGESEG**

Library for page segments.

**PSEG240**

Library for 240-pel image descriptor.

**PSEGAFF**

Library for resolution-independent image descriptor.

**LRECL**

Record length for the output data set.

**BLKSIZE**

Block size for the output data set.

## Uploading AFP files to z/OS

---

Before you can use the reblocking program, you must upload the AFP files from your workstation to the z/OS system. You can use any file transfer program to do this; however, keep these in mind:

- AFP files must be uploaded as binary files.
- The record format must be variable.
- The most efficient logical record length is 32756.

This command shows how to upload an AFP file called GRTNOVEL .AFP (created with the AFP Printer Driver for Windows) if you are using eNetwork Personal Communications file transfer program:

```
SEND d:\directoryname\GRTNOVEL.AFP sessionId:'userid.GRTNOVEL.UPLD' LRECL(32756)
```

## Reblocking data sets on z/OS

---

After you upload the AFP files to z/OS, you can reblock one or more data sets with the AFRREBLK command.

**Note:** If you are reblocking to a partitioned data set, you must preallocate the partitioned data set.

### AFRREBLK command

You can issue the AFRREBLK command on the TSO command line or from a data set selection list in ISPF 3.4. The command syntax for the AFRREBLK command is:

**AFRREBLK** *input\_dsn* [*output\_dsn*] [ ( ( [FONTLIB *fontlib*] [NOMSG] [DEL] [LL] [TSOERROR] ) ) ]

The values are:

***input\_dsn***

Specifies the name of the sequential or partitioned data set that is to be reblocked. Partitioned data set names must include a member name.

For example, if you uploaded GRTNOVEL . UPLD from your workstation to z/OS, you would issue this command to reblock the file:

```
AFRREBLK userid.GRTNOVEL.UPLD
```

**Note:** AFRREBLK reserves AFPDS for use as an output data set name extension; therefore, you must not use it as the third qualifier for the input data set name.

#### **output\_dsn**

Specifies an optional output data set name. It is the fully qualified name of a sequential or partitioned data set, without quotation marks. Partitioned data set names must include a member name.

If the output data set name exists, it is used as a fully qualified name. Otherwise, a new data set is allocated, prefixed by the user ID. To change the user ID, refer to the customize information in the AFRREBLK REXX program.

If this value is not specified, the value in the AFRREBLK profile is used. If no value can be determined from the profile, a sequential output data set is allocated with a first qualifier of your user ID, the same second qualifier as the input data set, and a third qualifier determined by the AFP object type. If the reblocking program does not recognize the AFP object type, it sets the third qualifier to AFPDS.

#### **FONTLIB fontlib**

Specifies the output font library, where *fontlib* is UB, 240, 300, or OLN.

FONT is prefixed to the *fontlib* value to form a keyword that is used to find the library name in the AFRREBLK profile. For example, when 240 is specified, the data set the FONT240 keyword specifies in the profile is used by the reblocking program.

You must specify this value if the file is a code page or coded font because, unlike other AFP objects, code pages and coded fonts do not contain any information that AFRREBLK can use to determine the library. AFRREBLK prompts you for the FONTLIB value if you do not specify it for code pages and coded fonts. For other files, the FONTLIB value is ignored if you specify it.

#### **NOMSG**

Specifies that messages are not displayed.

#### **DEL**

Specifies that a sequential input data set is deleted after processing. DEL is ignored if the input data set is partitioned.

#### **LL**

Specifies that each record is preceded by a 2-byte logical length field. The logical length field is stripped from the output.

#### **TSOERROR**

Specifies that the TSO message facility is turned on to display any error messages from TSO commands that this program uses.

## **Reblocking more than one data set at a time**

If you upload many files to z/OS but do not want to separately enter the AFRREBLK command to reblock each data set, you can use ISPF to display the names of the data sets, and then enter AFRREBLK one time in the command section to reblock all the data sets.

For example, if you uploaded numerous AFP documents to the host, all with a third qualifier of UPLD, UPLD240, or UPLD300, go to ISPF 3.4 and specify a DSNAMES LEVEL that corresponds to your uploaded data sets, such as:

```
userid.*.UPLD*
```

In the command section of the listing, enter AFRREBLK to the left of the first data set you want to reblock and = to the left of any other data sets you want to reblock (see [Figure 51 on page 166](#)).

```

DSLIST - DATA SETS BEGINNING WITH userid.*.UPLD* ----- ROW 1 OF 8
COMMAND ==> SCROLL ==> CSR

COMMAND      NAME                                     MESSAGE                                     VOLUME
-----
AFRREBLK  userid.C0428000.UPLD240                                USER25
=          userid.C0428000.UPLD300                                MIGRAT
=          userid.F1FRMDEF.UPLD                                  USER25
=          userid.GRTNOVEL.UPLD                                  USER25
=          userid.O10VERLY.UPLD                                  USER25
=          userid.P1PGEDEF.UPLD                                  USER25
=          userid.S1PGESEG.UPLD                                  USER25
=          userid.T1000850.UPLD                                  USER25
=          userid.X0423210.UPLD                                  USER21
***** END OF DATA SET LIST *****

```

Figure 51. Reblocking numerous data sets in ISPF

## Sample batch JCL

A sample job, APSWREBK, shows how to run the AFRREBLK program in a batch job. APSWREBK can be found in the SYS1.SAMPLIB library.

```

//** ** ** ** **
//* You can use this sample job to:
//* 1) Reblock an AFP file into an MVS dataset
//** ** ** **
//*
//*****
//* Delete Old Output Dataset
//*****
//DALLOC EXEC PGM=IEFBR14
//SYSPRINT DD SYSOUT=*
//SYSUT2 DD SYSOUT=*
//SYSIN DD DUMMY
//DELDS DD DSN=output.dsname <-- CHANGE TO YOUR DSN
// DISP=(MOD,DELETE,DELETE),UNIT=dasd,SPACE=(TRK.(1))
//*
//*****
//* Allocate New Output Dataset
//*****
//ALLOC EXEC PGM=IEFBR14
//SYSPRINT DD SYSOUT=*
//SYSUT2 DD SYSOUT=*
//SYSIN DD DUMMY
//ALLOCD DSN=output.dsname <-- CHANGE TO YOUR DSN
// DISP(NEW,CATLG,CATLG),
// UNIT=dasd,
// SPACE=(TRK,(1500,150),RLSE, <-- Sequential DS
//* SPACE=(TRK,(1500,150.6),RLSE, <-- Partition DS
// DCB=(LRECL=32756,RECFM=VB,BLKSIZE=0)
//*
//*****
//* Reblock AFP file
//*Input can be from an MVS data set or USS file
//*****
//REBLOCK EXEC PGM=IKJEFT01
//SYSPROC DD DSN=SYS1.SAPSEXEC,DISP=SHR <-- Location of AFRREBLK
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSTSIN DD DATA,DLM='*/'
AFRREBLK -
/usr/lpp/PSF/fonts/codepage/T1B00037.ECP -
output.dsname
*/
//* If output data set is a Partition DS, include member name:
//* output.dsname(member)

```

Figure 52. APSWREBK sample batch job to execute AFRREBLK program

In Figure 52 on page 166, the first two steps are optional. If you explicitly specify an output data set in the AFRREBLK command, and the data set already exists, and you do not want to clear out the contents of the data set, remove the first two steps. The syntax for the AFRREBLK command is described in “AFRREBLK command” on page 164.



---

## Chapter 10. Obtaining AFP statistics

You can use the AFP Statistics (AFPSTATS) option to obtain detailed information about a print file. You specify it individually for each print file and you can view the data online or print it. PSF presents the data in an AFPSTATS report, which provides details so you can:

- Determine in which resource library PSF found each resource.
- Determine the MO:DCA Interchange Set level.
- Diagnose some resource selection problems.
- Obtain statistical data about how a print file printed, such as the total number of pages, the number of times a specific resource was referenced, and the number of significant events. It is important to note that these statistics might contain some inaccuracies caused by error recovery and repositioning within the print file. Therefore, do not use these statistics for accounting purposes.
- Diagnose some print file printing performance problems.

The AFPSTATS report summarizes these resource types:

- Character set
- Coded font
- Code page
- Form definition
- Object container
- Overlay
- Page definition
- Page segment
- TrueType and OpenType fonts

**Note:** When the AFPSTATS report is active, PSF is collecting data and writing it to the AFPSTATS repository. These extra activities during PSF processing might cause degraded performance. Whenever you experience a problem with performance, first ensure that all extra activity, such as PSF traces and reports, are disabled before you confirm the performance problem.

---

### AFPSTATS repository

Before an AFPSTATS report can be generated, the system programmer must change the PSF startup procedure to define the AFPSTATS repository (the file where AFPSTATS reports are written) and then allocate the data set. This repository must be an existing PDSE data set. For information about defining the AFPSTATS repository, see *PSF for z/OS: Customization*.

PSF adds a member to this data set for every request it gets to produce an AFPSTATS report. PSF generates the member name and records this name in the message data set with message APS4001I. See the messages at the end of the print data set to determine where PSF placed the AFPSTATS report.

---

### Requesting an AFPSTATS report

You can request an AFPSTATS report for any PSF print file you own. The AFPSTATS report option is only activated if your system programmer added the appropriate AFPSTATS DD statement to the PSF startup procedure. Use one of these methods to request an AFPSTATS report:

- Use the AFPSTATS keyword on the OUTPUT JCL statement (see [“AFPSTATS” on page 85](#)). The valid values for AFPSTATS are YES, Y, NO, and N. NO is the default:

```
//OUT1 OUTPUT AFPSTATS=YES,...
//PRINT1 DD SYSOUT=A,OUTPUT=*.OUT1...
//
```

- Use installation Exit 7. The bit XTP7ASAP in the Exit 7 control structure specifies whether an AFPSTATS report is generated. For information about using Exit 7 to request an AFPSTATS report, see [PSF for z/OS: Customization](#).

**Note:** Any value that you specify for AFPSTATS in the OUTPUT JCL can be overridden by the XTP7ASAP bit in Exit 7. If the value is overridden by Exit 7, message APS7004I is printed in the message data set.

This example shows a job stream that produces a print file and an AFPSTATS report.

```
//JOB1 JOB ...
//STEP1 EXEC PGM=MYAPPL
//OUTMP OUTPUT AFPSTATS=YES
//MYPRINT DD SYSOUT=A,OUTPUT=*.OUTMP
//
```

The softcopy AFPSTATS report is stored on your system in the AFPSTATS repository and can be viewed or you can format it and print a hardcopy version.

## Softcopy report

This version of the AFPSTATS report is stored on your system in the AFPSTATS repository and can be viewed or formatted and printed. PSF generates a unique member name for the report and records this name in message APS4001I in the message data set for the print file.

## Format

The softcopy report is composed of variable-length records, with a maximum length of 512 characters per record. All records begin with a 10-character Layout ID or format identifier. [Table 8 on page 168](#) describes the overall format of a softcopy record. Individual AFPSTATS records are described in [Table 9 on page 169](#) and detailed information about the AFPSTATS softcopy report format is listed in [“Softcopy record details” on page 197](#).

Table 8. The basic format of the softcopy AFPSTATS report			
Offset	Length	Field Content	Description
<b>0</b>	10	Layout ID	The 10-byte record format ID for the formatting PAGEDEF. This value can also help identify the record type within the report. See <a href="#">Table 9 on page 169</a> .
<b>10</b>	2	Reserved	Reserved
<b>12</b>	1		Column separator – blank
<b>13</b>	1–499	Record specific	This data varies by record type. See <a href="#">“Softcopy record details” on page 197</a> .

## Records

The softcopy report is composed of records that begin with a 10-character Layout ID. [Table 9 on page 169](#) lists all the possible softcopy records and describes how the records are used.

Table 9. Softcopy AFPSTATS records

Layout ID	Purpose
<b>COMMENT</b>	This record is used to add a text string to the report; typically blank lines to make the report more readable.
<b>EVENT</b>	This record identifies significant events that occurred during PSF processing. For example, a "Repositioning" event indicates that PSF is repositioning itself to a different point in the input data stream, possibly because of an operator request to backspace, an I/O error, or typical PSF processing.
<b>EVENTFM</b>	This record identifies when PSF maps a font from one type of supported font to another type of supported font. The supported font mappings are: GRID to raster font names, GRID to outline font names, outline font to raster font, and raster font to outline font.  <b>Note:</b> If a font was previously mapped to a font, that map event is not recorded again in this record.
<b>EVENT-LIST</b>	This record is used to list all of the events that occurred while the print file is processing.
<b>EVENTRL</b>	This record identifies the result of a resource reload request that occurred during Exit 7 processing.
<b>EVENTRS</b>	This record identifies the result of a resource substitution request that occurred during Exit 7 processing.
<b>HEADING</b>	This record contains the headings for report sections that are presented in a tabular format.  <b>Note:</b> This record is generated only for PSF 4.6.0 and earlier releases.
<b>HEADINGP</b>	This record contains the headings for report sections that are presented in a tabular format.
<b>HEADINGR</b>	This record contains the AFP Download Plus remote system headings, presented in tabular format.
<b>HEADING-DP</b>	This record contains the headings for the Disposition Summary subsection.
<b>HEADING-EL</b>	This record contains the headings for the Significant Events subsection.
<b>HEADING-LC</b>	This record contains the headings for the Location Summary subsection.
<b>HEADING-RF</b>	This record contains the headings for the Reference Summary subsection.
<b>HEADING-SP</b>	This record contains the headings for the Summary of Pages subsection.
<b>HEADING-UU</b>	This record contains the headings for the unused inline resources subsection.
<b>META-LIST</b>	This record is used to list all the valid inline metadata objects that are found in the print file.
<b>NOTE</b>	This record is used to add important comments to the report.
<b>PAGEFOOTER</b>	This record is used to identify the page footer that is used in the hardcopy report. It is typically the same as the TITLE record section.
<b>PRINTFILE</b>	This record identifies the print file that is processed when the report is generated. It also presents some information about PSF and identifies the printer.  <b>Note:</b> This record is generated only for PSF 4.6.0 and earlier releases.

Table 9. Softcopy AFPSTATS records (continued)

Layout ID	Purpose
<b>PRINTFILEP</b>	This record identifies the print file that is processed when the report is generated. It also presents some information about PSF and identifies the printer.
<b>PRINTFILR</b>	This record contains print file remote system receiver information when the report is generated for AFP Download Plus.
<b>PRTFILEX</b>	This record identifies print file extension information for the print file that is processed.
<b>REPORTLVL</b>	This record identifies the specific report format level, which might change because of service and future PSF development.
<b>RESOURCE</b>	This record identifies individual resources that are used and the order PSF processes them.
<b>RESOURCEDF</b>	This record identifies the first use of a TrueType font resource with a name longer than eight characters.
<b>RESOURCEDO</b>	This record identifies the first use of a data object resource or extended code page.
<b>RESOURCEGR</b>	This record identifies the first use of a GRID font that is activated in the printer without being mapped.
<b>SECTION</b>	This record identifies when a report section is divided into a subsection.
<b>SUMM-DISP</b>	This record identifies a resource type and how PSF satisfied the resource request for that type of resource.
<b>SUMM-DSN</b>	This record identifies a repository or data set from which PSF retrieved a resource. PSF obtains resources from resource repositories that are identified to it in the PSF startup procedure or in the USERLIB parameter on the OUTPUT JCL statement.
<b>SUMM-DSN-R</b>	This record contains a list of up to six resources that are retrieved from the resource repository that is defined in a SUMM_DSN record.
<b>SUMM-LOC</b>	This record identifies a resource type and where PSF found the resources for that type. PSF can locate resources in various resource repositories, including inline, user specified libraries, and system libraries.
<b>SUMM-NAME</b>	This record identifies a resource and provides statistics about how the resource was introduced, where PSF found the resource, and how PSF satisfied the resource request. This record is for all resources except GRID font, TrueType font, and OpenType font.
<b>SUMM-NAMEX</b>	This record identifies a GRID font, TrueType font, or OpenType font resource and provides statistics about how the resource was introduced, where PSF found the resource, and how PSF satisfied the resource request.
<b>SUMM-PAGE</b>	This record provides statistics about the source print file, including number of pages processed, page size, and number of records.
<b>SUMM-PATH</b>	This record identifies a path directory from which PSF retrieves a resource while it processes a print file. PSF obtains resources from resource path libraries that are identified to it in the USERPATH parameter on the OUTPUT JCL statement.
<b>SUMM-PTH-R</b>	This record identifies the resource name that is found in the path directory. The resource name is cross-referenced to the resource name in the RESOURCEDF record, which contains the resource file name.

Table 9. Softcopy AFPSTATS records (continued)	
Layout ID	Purpose
<b>SUMM-REF</b>	This record identifies a resource type and the various ways it was referenced.
<b>TITLE</b>	This record identifies the report sections that are found in the AFPSTATS report.
<b>UIR-LIST</b>	This record identifies an inline resource in the inline resource group that is not used during data set processing.

## Example

For an example of the softcopy report, see “[Sample softcopy report](#)” on page 214.

## Hardcopy report

The hardcopy report contains the same information as the softcopy report, but it is formatted in sections with page numbers. Each section is formatted to 8.5 x 11 inches in a combination of landscape and portrait orientation. [Table 10 on page 171](#) lists the sections that make up a hardcopy report.

Table 10. Sections in the hardcopy AFPSTATS report		
Report Section Title	Description	Corresponding Softcopy Records
<b>Print File Information</b>	Contains data collected about the print file, PSF, and the printer	PRINTFILE (PSF 4.6.0), PRINTFILEP
<b>Print File Extension Information</b>	Contains additional data collected about the print file.	PRTFILEX
<b>Print File Remote System Information</b>	Contains AFP Download Plus information collected about the remote system.	PRINTFILER
<b>Processing Detail</b>	Contains data about the first reference to each resource used by the print file	RESOURCE, RESOURCEDF, RESOURCEGR
	Contains a record of significant events that occurred while the print file is processing, including when PSF maps a font from one type of supported font to another type of supported font, the result of a resource reload request, and the result of a resource substitution request	EVENT, EVENTFM, EVENTRL, EVENTRS
<b>Resource Summary by Name</b>	Contains summary data about each resource referenced by the print file	SUMM-NAME, SUMM-NAMEX
<b>Resource Summary by Data Set</b>	Lists all the objects that are retrieved from the specified data sets; lists the resource name that is found in the specified path directory	SUMM-DSN, SUMM-DSN-R, SUMM-PATH, SUMM-PTH-R

Table 10. Sections in the hardcopy AFPSTATS report (continued)

Report Section Title	Description	Corresponding Softcopy Records
<b>Resource Summary by Resource Type</b>	Contains summary information about the print file by resource type	SUMM-REF, SUMM-LOC, SUMM-DISP
<b>Processing Summary</b>	Summarizes the pages that are processed, lists the processing events, lists unused inline resources, and lists valid metadata objects that are specified in the print file.	SUMM-PAGE, EVENT-LIST, UIR-LIST, META-LIST

## Generating a hardcopy report

To generate a hardcopy AFPSTATS report, use the following PSF-supplied page definition and form definition to format the softcopy report:

- Page definition: P1ASAP04
- Form definition: F1ASAP01

The recommended page definition uses PPFA record formatting and conditional processing constructs to define the resulting AFPSTATS report. It uses proportional spaced, sans-serif fonts from AFP Font Collection, Program Number 5648-B33, or the z/OS Font Collection.

This example shows a job stream that formats an existing AFPSTATS report for printing:

```
//JOB1      JOB      ...
//STEP1     EXEC     PGM=IEBGENER
//SYSPRINT  DD       SYSOUT=*
//SYSIN     DD       DUMMY
//OUTRL     OUTPUT   PAGEDEF=ASAP04,FORMDEF=ASAP01
//SYSUT2    DD       SYSOUT=*, OUTPUT=*.OUTRL
//SYSUT1    DD       DSN=WRITES600.AFPSTATS(A0317900),DISP=SHR
//
```

## Example

See [“Sample hardcopy report” on page 216](#) for an example of the hardcopy report.

---

# Chapter 11. Diagnosing incorrect printer output

When your output does not print correctly, you might need help diagnosing the problem if you do not see any messages or you do not understand the messages.

---

## Messages are not visible

If your output does not print correctly and you do not see any messages, one of these situations might have occurred:

### **Messages are redirected**

See [“Specifying whether you want error messages to be printed” on page 139](#) for information about where your messages might have been sent.

### **Errors are blocked**

The DATAACK JCL parameter specifies whether you want the printer to block print-positioning and incorrect-character errors. Do one of these if the DATAACK parameter is blocking errors:

- Specify DATAACK=UNBLOCK.
- Remove the DATAACK parameter from your JCL.

### **Certain types of errors are not reported**

In a form definition, fidelity triplets in the Presentation Fidelity Control (PFC) command determine whether certain data errors stop PSF processing, are not reported, or both. If PFC fidelity triplets are specified, verify that the triplet has a REPORT setting that is set to report the error.

---

## Messages do not help

If your output does not print correctly and you received an error message, you might not understand how to solve the problem from the message text alone. Be sure to read the complete help for the message, including the explanation and user's response, in [PSF for z/OS: Messages and Codes](#).





## Appendix A. Form definitions supplied with PSF

This information describes the form definitions that IBM supplies with PSF.

The source modules are stored in the SYS1.SAPSPDFD library. You can use the source code to customize form definitions and page definitions for your organization.

### Form definition naming convention

Table 11 on page 175 shows the naming conventions for the form definitions that are supplied with PSF. These form definitions have up to eight positions in the form definition names, such as F1A00010. If you begin at the left character (F) and count positions from left to right, you can determine the meaning of the form definition characters. For example, F1A00010 means:

F1: Form definition

A0: An AFP printer other than the 3800

00: Offset 0,0

1: Bin 1

0: None (Simplex)

Table 11. Form definition naming convention from left to right		
Positions	Values	Description
1–2	F1	Form definition
3–4	A1 or A0	All AFP printers other than the 3800
	CP	HP printers (PCL4, PCL5) through a print server for Windows
	C1 or C0	3800 compatibility
	H1	3-hole punched paper
	FC	Finisher with corner staple
	FE	Finisher with edge stitch
	FS	Finisher with saddle stitch
	FZ	Finisher with Z-fold
	N2	N_UP (2 up)
5–6	00	Offset 0,0
	01	Offset 0.0165,0.0165
	10	Offset 0,0; Duplex=None (Simplex)
	11	Offset 0,0; Duplex=Normal (Duplex)
	12	Offset 0,0; Duplex=Tumble (Duplex)
7–8	LA	Landscape across
	LD	Landscape down
	PA	Portrait across
	PD	Portrait down

Table 11. Form definition naming convention from left to right (continued)		
Positions	Values	Description
<b>7</b>	E	Envelope
	M	Manual
	<i>n</i>	Bin number
<b>8</b>	0	None (Simplex)
	1	Normal (Duplex)
	2	Tumble (Duplex)

## Form definition for the 3800 printer

Table 12 on page 176 describes the form definition supplied with PSF for the 3800 printer. The form definition specifies:

- One copy
- No overlays
- No offset stacking or copy marking

Table 12. Form definition for the 3800 printer			
Form Definition Name	Copy Group	Page Position in Inches	Flash
<b>F10101</b>	F20101	0.0, 0.5	No

## Form definitions for printers other than the 3800, PCL4, and PPDS printers

Many form definitions in the tables here use Bin 1. By default, 64xx and 65xx printers emulate a 4234 printer, which supports only bin number 255. Using these form definitions with a 4234 printer or a 64xx or 65xx printer emulating a 4234 printer causes an error. Before using these form definitions with a 64xx printer or a 65xx printer, configure the 64xx printer to emulate a 64xx printer and configure the 65xx printer to emulate a 6408 printer, not a 4234 printer.

Table 13 on page 176 describes the form definitions for printers other than the 3800, PCL4, and PPDS printers, with the name and a description of each. On some printers, printing near the edge of the paper can result in poor print quality in the border area. If the printer has a third paper source, you can use it with some of the form definitions. For the limitations on your printer, see the publications for your printer. Each of these form definitions specifies:

- One copy
- No overlays
- No offset stacking or copy marking

Table 13. Form definitions for all printers other than the 3800, PCL4, and PPDS printers				
Form Definition Name	Copy group	Page Position in Inches	Duplex	Paper Source
<b>F1A10110</b>	F2A10110	0.165, 0.165	Duplex off	Primary
<b>F1I30110</b>	F2I30110	0.165, 0.165	Duplex off	Primary
<b>F1A10120</b>	F2A10120	0.165, 0.165	Duplex off	Alternative
<b>F1A10130</b>	F2A10130	0.165, 0.165	Duplex off	Third
<b>F1A10111</b>	F2A10111	0.165, 0.165	Normal duplex	Primary
<b>F1I30111</b>	F2I30111	0.165, 0.165	Normal duplex	Primary

Table 13. Form definitions for all printers other than the 3800, PCL4, and PPDS printers (continued)

Form Definition Name	Copy group	Page Position in Inches	Duplex	Paper Source
<b>F1A10121</b>	F2A10121	0.165, 0.165	Normal duplex	Alternative
<b>F1A10131</b>	F2A10131	0.165, 0.165	Normal duplex	Third
<b>F1A10112</b>	F2A10112	0.165, 0.165	Tumble duplex	Primary
<b>F1A10122</b>	F2A10122	0.165, 0.165	Tumble duplex	Alternative
<b>F1A10132</b>	F2A10132	0.165, 0.165	Tumble duplex	Third
<b>F1A10140</b>	F2A10140	0.165, 0.165	Duplex off	Fourth
<b>F1A10141</b>	F2A10141	0.165, 0.165	Normal duplex	Fourth
<b>F1A10142</b>	F2A10142	0.165, 0.165	Tumble duplex	Fourth

**Note:** The 3820 lines up B4-size paper differently from other paper because of the B4 paper length. To compensate for this, you can create a form definition with a page position of 0.10, 0.00 inch.

Table 14 on page 177 lists form definitions that you can use to print envelopes or use the manual input bin on the 4028 printer.

Table 14. Form definitions for printing envelopes on the 4028

Form Definition Name	Copy group	Page Position in Inches	Duplex	Paper Source
<b>F1A101E0</b>	F2A101E0	0.165,0.165	Duplex off	Envelope
<b>F1A000E0</b>	F2A000E0	0,0	Duplex off	Envelope
<b>F1A101M0</b>	F2A101M0	0.165,0.165	Duplex off	Manual
<b>F1A000M0</b>	F2A000M0	0,0	Duplex off	Manual

Table 15 on page 177 lists form definitions that specify a 0,0 offset. These form definitions are for printing on printers other than the 3800 printer.

Table 15. Form definitions with a 0,0 offset

Form Definition Name	Copy group	Page Position in Inches	Duplex	Paper Source
<b>F1A00010</b>	F2A00010	0,0	Duplex off	Primary
<b>F1A00011</b>	F2A00011	0,0	Normal duplex	Primary
<b>F1A00012</b>	F2A00012	0,0	Tumble duplex	Primary
<b>F1A00020</b>	F2A00020	0,0	Duplex off	Alternative
<b>F1A00021</b>	F2A00021	0,0	Normal duplex	Alternative
<b>F1A00022</b>	F2A00022	0,0	Tumble duplex	Alternative
<b>F1A00030</b>	F2A00030	0,0	Duplex off	Third
<b>F1A00031</b>	F2A00031	0,0	Normal duplex	Third
<b>F1A00032</b>	F2A00032	0,0	Tumble duplex	Third
<b>F1A00040</b>	F2A00040	0,0	Duplex off	Fourth
<b>F1A00041</b>	F2A00041	0,0	Normal duplex	Fourth
<b>F1A00042</b>	F2A00042	0,0	Tumble duplex	Fourth

Table 16 on page 178 lists form definitions for N\_UP 2 printing. These form definitions define two pages on a side of a sheet. For more information about using the N\_UP subcommand to create N\_UP form definitions, see [Page Printer Formatting Aid: User's Guide](#).

Table 16. Form definitions for N\_UP 2 printing

Form Definition Name	Copy group	Page Position in Inches	Duplex	Paper Source
<b>F1N20110</b>	F2N20110	0.165,0.165	Duplex off	Primary
<b>F1N20111</b>	F2N20111	0.165,0.165	Normal duplex	Primary
<b>F1N20112</b>	F2N20112	0.165,0.165	Tumble duplex	Primary
<b>F1N20130</b>	F2N20130	0.165,0.165	Duplex off	Third
<b>F1N20131</b>	F2N20131	0.165,0.165	Normal duplex	Third
<b>F1N20132</b>	F2N20132	0.165,0.165	Tumble duplex	Third

Table 17 on page 178 lists form definitions to use with three-hole punched paper when you are printing with page definitions also designed for use with three-hole punched paper. You can use any of these form definitions with any of the page definitions for three-hole punched paper.

Table 17. Form definitions for three-hole punched paper

Form Definition Name	Copy group	Page Position in Inches	Duplex	Paper Source
<b>F1H10110</b>	F2H10110	1.000,0.165	Duplex off	Primary
<b>F1H10111</b>	F2H10111	1.000,0.165 0.165,0.165	Normal duplex	Primary
<b>F1H10112</b>	F2H10112	1.000,0.165 1.000,0.165	Tumble duplex	Primary
<b>F1H10120</b>	F2H10120	1.000,0.165	Duplex off	Alternative
<b>F1H10121</b>	F2H10121	1.000,0.165 0.165,0.165	Normal duplex	Alternative
<b>F1H10122</b>	F2H10122	1.000,0.165 1.000,0.165	Tumble duplex	Alternative
<b>F1H10130</b>	F2H10130	1.000,0.165	Duplex off	Third
<b>F1H10131</b>	F2H10131	1.000,0.165 0.165,0.165	Normal duplex	Third
<b>F1H10132</b>	F2H10132	1.000,0.165 1.000,0.165	Tumble duplex	Third
<b>F1H10140</b>	F2H10140	1.000,0.165	Duplex off	Fourth
<b>F1H10141</b>	F2H10141	1.000,0.165 0.165,0.165	Normal duplex	Fourth
<b>F1H10142</b>	F2H10142	1.000,0.165 0.000,0.165	Tumble duplex	Fourth

Table 18 on page 178 lists form definitions to use when you want to rotate pages on the paper. These form definitions are for printing on printers other than the 3800.

Table 18. Form definitions for rotating pages on the paper

Form Definition Name	Presentation Mode	Print Direction	Page Position in Inches	Duplex
<b>F1A010LA</b>	Landscape	Across	0,0	Duplex off
<b>F1A010LD</b>	Landscape	Down	0,0	Duplex off
<b>F1A010PA</b>	Portrait	Across	0,0	Duplex off
<b>F1A010PD</b>	Portrait	Down	0,0	Duplex off
<b>F1A011LA</b>	Landscape	Across	0,0	Normal duplex

Table 18. Form definitions for rotating pages on the paper (continued)

Form Definition Name	Presentation Mode	Print Direction	Page Position in Inches	Duplex
<b>F1A011LD</b>	Landscape	Down	0,0	Normal duplex
<b>F1A011PA</b>	Portrait	Across	0,0	Normal duplex
<b>F1A011PD</b>	Portrait	Down	0,0	Normal duplex
<b>F1A012LA</b>	Landscape	Across	0,0	Tumble duplex
<b>F1A012LD</b>	Landscape	Down	0,0	Tumble duplex
<b>F1A012PA</b>	Portrait	Across	0,0	Tumble duplex
<b>F1A012PD</b>	Portrait	Down	0,0	Tumble duplex

Table 19 on page 179 lists form definitions that can be used to select print quality at a 64xx printer or a 65xx printer. To facilitate printing edge-to-edge, these form definitions specify a page position of 0,0. These form definitions also specify a bin number of 1.

**Note:** By default, a 64xx printer emulates a 4234 printer, which supports only a bin number of 255. These form definitions specify bin number 1; therefore, using these form definitions with a 64xx printer that emulates a 4234 printer causes an error. Before you use these form definitions with a 64xx printer, configure the printer to emulate a 64xx printer and not a 4234 printer. The 65xx printer also defaults to emulate a 4234 printer. Before you use these form definitions with a 65xx printer, configure the 65xx printer to emulate a 6408 printer and not a 4234 printer.

Table 19. Form definitions used to select print quality on a 64xx or 65xx printer

FORMDEF	Page position in inches	Print Quality	Duplex
<b>F1Q10010</b>	0.0, 0.0	Draft	Duplex off
<b>F1Q50010</b>	0.0, 0.0	DP	Duplex off
<b>F1QA0010</b>	0.0, 0.0	NLQ	Duplex off

## Form definitions for HP PCL4 and PPDS printers

Table 20 on page 179 describes the form definitions supplied with PSF for printing on HP PCL4 or PPDS printers through a print server for Windows. These form definitions must be used in combination with page definitions intended for these printers. The form definitions specify:

- One copy
- No overlays
- No offset stacking or copy marking

Table 20. Form definitions for HP PCL4 and PPDS printers

Form Definition Name	Copy group	Page Position in Inches	Duplex	Paper Source
<b>F1CP0110</b>	F2CP0110	0.250,0.200	Duplex off	Primary
<b>F1CP0120</b>	F2CP0120	0.250,0.200	Duplex off	Alternative
<b>F1CP0111</b>	F2CP0111	0.250,0.200	Normal duplex	Primary
<b>F1CP0121</b>	F2CP0121	0.250,0.200	Normal duplex	Alternative
<b>F1CP0112</b>	F2CP0112	0.250,0.200	Tumble duplex	Primary
<b>F1CP0122</b>	F2CP0122	0.250,0.200	Tumble duplex	Alternative

## Compatibility form definitions

Table 21 on page 180 and Table 22 on page 180 describe compatibility form definitions that are supplied with PSF. These form definitions can be used to print data that is formatted for the 3800 or cut-sheet printers on an AFCCU continuous-forms printer such as a 3900, InfoPrint 3000, or InfoPrint 4000/4100. Use one of the compatibility form definitions when you are printing these on a continuous-forms printer:

- Data that is formatted for landscape presentation on a cut-sheet printer.
- Data that is formatted for either portrait or landscape presentation on a 3800 printer. The form definition that is used depends on whether the data was formatted for printing on wide or narrow paper.

These form definitions specify:

- One copy group <sup>17</sup>
- One copy
- No offset stacking or copy marking
- Primary paper source
- No overlays

Form Definition Name	Compatible with	Presentation Mode	Print Direction	Duplex	Page Position in Inches
<b>F1C10110</b>	Cut-sheet printers	Landscape	Down	Duplex off	0.165, 0.165
<b>F1C10111</b>	Cut-sheet printers	Landscape	Down	Normal duplex	0.165, 0.165
<b>F1C10112</b>	Cut-sheet printers	Landscape	Down	Tumble duplex	0.165, 0.165
<b>F10101PD</b>	3800 printer wide forms	Portrait	Down	Duplex off	0.00, 0.50
<b>F10101LA</b>	3800 printer wide forms	Landscape	Across	Duplex off	0.00, 0.50
<b>F10101PA</b>	3800 printer narrow forms	Portrait	Across	Duplex off	0.00, 0.50
<b>F10101LD</b>	3800 printer narrow forms	Landscape	Down	Duplex off	0.00, 0.50

Table 22. N\_UP compatibility form definitions for AFCCU continuous-forms printers. The use of the N\_UP form definitions in this table on the 3800 printer cause PSF to issue message APS283I or APS284I. The output is printed appropriately.

Form Definition Name	Compatible with	Presentation Mode	Print Direction	Duplex	Page Position in Inches	N_UP
<b>F1N201PD</b>	3800 printer wide forms (N_UP of None)	Portrait	Down	Duplex off	0.00, 0.50	2
<b>F1N201LA</b>	3800 printer wide forms (N_UP of None)	Landscape	Across	Duplex off	0.00, 0.50	2
<b>F1N201PA</b>	3800 printer narrow forms (N_UP of None)	Portrait	Across	Duplex off	0.00, 0.50	2
<b>F1N201LD</b>	3800 printer narrow forms (N_UP of None)	Landscape	Down	Duplex off	0.00, 0.50	2

<sup>17</sup> The name of the copy group is the same as the name of the form definition, except for the prefix. For example, form definition F10101PA contains one copy group named F20101PA.

Table 23 on page 181 describes compatibility form definitions that are supplied with PSF. These form definitions can be used to print data that is formatted for the 3800 printer on a cut-sheet printer, such as the InfoPrint 11xx series, InfoPrint 2000, and InfoPrint 2085 and 2105. When you are printing on a cut-sheet printer, use one of the compatibility form definitions with data formatted for either portrait or landscape presentation on a 3800 printer. The form definition that you use depends on whether the data was formatted for printing on wide or narrow paper.

<i>Table 23. 3800 compatibility form definitions for cut-sheet printers</i>					
<b>Form Definition Name</b>	<b>Compatible with</b>	<b>Presentation Mode</b>	<b>Print Direction</b>	<b>Duplex</b>	<b>Page Position in Inches</b>
<b>F1C010LA</b>	3800 printer wide forms	Landscape	Across	Duplex off	0.00, 0.50
<b>F1C010PD</b>	3800 printer wide forms	Portrait	Down	Duplex off	0.00, 0.50
<b>F1C010LD</b>	3800 printer narrow forms	Landscape	Down	Duplex off	0.00, 0.50
<b>F1C010PA</b>	3800 printer narrow forms	Portrait	Across	Duplex off	0.00, 0.50
<b>F1C011LA</b>	3800 printer wide forms	Landscape	Across	Normal duplex	0.00, 0.50
<b>F1C011PD</b>	3800 printer wide forms	Portrait	Down	Normal duplex	0.00, 0.50
<b>F1C011LD</b>	3800 printer narrow forms	Landscape	Down	Normal duplex	0.00, 0.50
<b>F1C011PA</b>	3800 printer narrow forms	Portrait	Across	Normal duplex	0.00, 0.50
<b>F1C0102LA</b>	3800 printer wide forms	Landscape	Across	Tumble duplex	0.00, 0.50
<b>F1C012PD</b>	3800 printer wide forms	Portrait	Down	Tumble duplex	0.00, 0.50
<b>F1C012LD</b>	3800 printer narrow forms	Landscape	Down	Tumble duplex	0.00, 0.50
<b>F1C012PA</b>	3800 printer narrow forms	Portrait	Across	Tumble duplex	0.00, 0.50

For more information about compatibility form definitions, see [“Page-presentation compatibility”](#) on page 59.

## Form definitions for special purpose jobs

Table 24 on page 181 describes the form definitions supplied with PSF for all special purposes except finishing. You can use these form definitions on any AFP printers. Each of these form definitions specifies:

- One copy
- No offset stacking or copy markings
- Primary paper source
- Duplex off
- No forms flash

<i>Table 24. Form definitions supplied for special purposes</i>			
<b>Form Definition Name</b>	<b>Copy Group</b>	<b>Page Position in Inches</b>	<b>Overlay</b>
<b>F1IBM</b>	IBM	0.0, 0.5	O1IBM
<b>F1OGL</b>	OGL	0.0, 0.0	None

Form definition F1IBM is provided for use in verifying the installation and prints a supplied overlay that contains the IBM logo. This form definition would ordinarily be used only during system installation to verify that the PSF product is working correctly.

F1OGL is provided for use with the Overlay Generation Language (OGL) program. Use F1OGL to print the overlay sample created during the processing of your OGL statements. The overlay sample is a page, identical to the overlay resource, that can be sent directly to the printer. F1OGL positions the overlay sample page at the medium origin (0.0, 0.0), which ensures that the print position of the overlay sample matches the print position of the overlay resource when it is called for printing by a user-specified form definition. For more information about using OGL to produce overlays and overlay samples, see *Overlay Generation Language/370 User's Guide and Reference*.

You can use F1OGL for printing any job in which you want the page to be positioned at the medium origin. However, if you position a page at the medium origin, you must ensure that data in the page does not fall outside the printable area of the sheet. For more information about the areas on which your printer can print, see the documentation provided with the printer.

## Form definitions for finishing your output

Table 25 on page 182 describes the form definitions that are supplied with PSF for printers that support finishing operations. You cannot use these form definitions to specify finishing operations on individual copy groups.

Table 25. Form definitions for printers that support finishing					
Form Definition Name	Copy Group	Finishing	Page Position in Inches	Paper Source	Duplex
<b>F1FC0010</b>	F1FC0010	Upper left corner staple	0,0	Bin 1	Duplex Off
<b>F1FC0011</b>	F1FC0011	Upper left corner staple	0,0	Bin 1	Normal duplex
<b>F1FC0012</b>	1FC0012	Upper left corner staple	0,0	Bin 1	Tumble duplex
<b>F1FE0010</b>	F1FE0010	Left edge stitch	0,0	Bin 1	Duplex off
<b>F1FE0011</b>	F1FE0011	Left edge stitch	0,0	Bin 1	Normal duplex
<b>F1FE0012</b>	F1FE0012	Left edge stitch	0,0	Bin 1	Tumble duplex
<b>F1FS0010</b>	F1FS0010	Saddle stitch	0,0	Bin 1	Display On
<b>F1FS0011</b>	F1FS0011	Saddle stitch	0,0	Bin 1	Normal duplex
<b>F1FZ0030</b>	F1FZ0030	Z-fold	0,0	Bin 3	Normal duplex

Table 26 on page 182 describes more form definitions that are supplied with PSF for printers that support finishing operations. These form definitions are exceptions to the naming convention formula.

Table 26. Additional form definitions for printers that support finishing					
Form Definition Name	Copy Group	Finishing	Page Position in Inches	Paper Source	Duplex
<b>F1FEC010</b>	F1FEC010	Left edge stitch, cover sheet	0,0	Bin 7	None simplex
<b>F1FEL010</b>	F1FEL010	Left edge staple	0,0	Bin 1	None simplex
<b>F1FER010</b>	F1FER010	Right edge staple	0,0	Bin 1	None simplex
<b>F1FEZ010</b>	F1FEZ010	Edge stitch with Z fold	0,0	Bin 3	None simplex
<b>F1FS2030</b>	F1FS2030	Saddle stitch, 2 up	0,0	Bin 3	Normal duplex
<b>F1FZ1021</b>	F1FZ1021	Z-fold ledger, 1_Up, Landscape	0,0	Bin 2	Normal duplex



<i>Table 26. Additional form definitions for printers that support finishing (continued)</i>					
<b>Form Definition Name</b>	<b>Copy Group</b>	<b>Finishing</b>	<b>Page Position in Inches</b>	<b>Paper Source</b>	<b>Duplex</b>
<b>F1FZ2021</b>	F1FZ2021	Z-fold ledger, 2_Up, Portrait	0,0	Bin 2	Normal duplex

## Form definitions for printing PSF reports

Table 27 on page 183 describes the form definitions supplied with PSF for printing PSF reports.

<i>Table 27. Form definitions for printing PSF reports</i>	
<b>Form Definition Name</b>	<b>Report Type</b>
<b>F1ASAP01</b>	AFP Statistics (AFPSTATS)
<b>F1DPI01</b>	Printer Information
<b>F1PPCT01</b>	Tier Level (produced by the Point Counting Tool)



## Appendix B. Page definitions supplied with PSF

This information describes the page definitions supplied with PSF. These page definitions work only with traditional line data. Page definitions are available for the various paper sizes used by the printers supported by PSF. For information about the paper sizes your printer uses, see the publications for your printer.

### Page definitions for the 3800 printer

The forms control buffer (FCB) modules available with the 3800 Model 1 (a line printer) are converted to page definitions for the 3800 page printers (Models 3, 6, and 8). Table 28 on page 185 lists the FCBs and the names of the corresponding page definitions with their descriptions. In the tables, the abbreviation *lpi* stands for lines per inch.

Table 28. 3800 Model 1 FCBs and corresponding page definitions for 14.88 x 11-inch paper

FCB	Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
FCB3STD1	P1STD1	13.87 x 8.5	51 at 6 lpi	29/0	GT10	Across	Landscape
FCB3STD2	P1STD2	13.87 x 10.0	60 at 6 lpi	29/0	GT10	Across	Landscape
FCB3STD3	P1STD3	13.87 x 10.0	80 at 8 lpi	23/0	GT12	Across	Landscape
FCB36	P16	13.87 x 8.5	51 at 6 lpi	29/0	GT10	Across	Landscape
FCB38	P18	13.87 x 8.5	68 at 8 lpi	23/0	GT12	Across	Landscape

More page definitions for the 3800 printer are available for formatting three of the more common paper sizes, 12 x 8.5, 9.5 x 11, and 14.88 x 11 inches, and for 2-up printing applications (placing two pages of application data on a form).

Many of these page definitions have 5-character names that follow the P1 prefix. To specify such a page definition with an FCB parameter, you must first rename the page definition with a 1- to 4-character name that follows the P1 prefix.

Each of these page definitions specifies a channel-1 (carriage) control character to position printing at the top of the next page.

Table 29 on page 185 describes the page definitions for paper that measures 12 x 8.5 inches.

Table 29. Page definitions for 12 x 8.5-inch paper

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P104560	11.0 x 7.5	45 at 6 lpi	29/0	GT10	Across	Landscape
P106061	7.5 x 10.0	60 at 6 lpi	29/0	GT10	Down	Portrait
P106080	11.0 x 7.5	60 at 8 lpi	23/0	GT12	Across	Landscape
P108081	7.5 x 10.0	80 at 8 lpi	23/0	GT12	Down	Portrait
P1075A0	11.0 x 7.5	75 at 10 lpi	19/0	GT20	Across	Landscape
P1100A1	7.5 x 10.0	100 at 10 lpi	19/0	GT20	Down	Portrait
P1090C0	11.0 x 7.5	90 at 12 lpi	15/0	GT20	Across	Landscape

<sup>18</sup> Position down and position across are shown in logical units, with 240 logical units per inch. For example, 24 logical units are equal to 0.10 inch. Measurements that are specified in logical units are the same for all the printers PSF supports, regardless of the resolution or pel-density of the printer.

Table 30 on page 186 describes the page definitions for paper that measures 9.5 x 11.0 inches.

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P106060	8.5 x 10.0	60 at 6 lpi	29/0	GT10	Across	Portrait
P105161	10.0 x 8.5	51 at 6 lpi	29/0	GT10	Down	Landscape
P108080	8.5 x 10.0	80 at 8 lpi	23/0	GT12	Across	Portrait
P106881	10.0 x 8.5	68 at 8 lpi	23/0	GT12	Down	Landscape
P1100A0	8.5 x 10.0	100 at 10 lpi	19/0	GT20	Across	Portrait
P1085A1	10.0 x 8.5	85 at 10 lpi	19/0	GT20	Down	Landscape
P1120C0	8.5 x 10.0	120 at 12 lpi	15/0	GT20	Across	Portrait
P1102C1	10.0 x 8.5	102 at 12 lpi	15/0	GT20	Down	Landscape

Table 31 on page 186 describes the page definitions for paper that measures 14.88 x 11.0 inches.

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1L06060	13.88 x 10.0	60 at 6 lpi	29/0	GT10	Across	Landscape
P1L08080	13.88 x 10.0	80 at 8 lpi	23/0	GT12	Across	Landscape
P1L100A0	13.88 x 10.0	100 at 10 lpi	19/0	GT20	Across	Landscape
P1L120C0	13.88 x 10.0	120 at 12 lpi	15/0	GT20	Across	Landscape

Table 32 on page 186 describes the page definitions for multiple-up printing.

Page Definition Name	Form Size: Width by Height, in Inches	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1M12060	14.88 x 11.0	13.88 x 10.0	60 at 6 lpi 60 at 6 lpi	29/0 29/1728	GT12	Across	Landscape: Side-by-Side
P1M120C1	12 x 8.5	7.5 x 10.0	60 at 12.5 lpi 60 at 12.5 lpi	15/0 1197/0	GT20	Down	Portrait: Over/Under
P1M120C0	9.5 x 11.0	8.5 x 10.0	60 at 12.5 lpi 60 at 12.5 lpi	15/0 1245/0	GT20	Across	Portrait: Over/Under
P1M16080 <sup>19</sup>	14.88 x 11.0	13.88 x 10.0	80 at 8 lpi 80 at 8 lpi	23/0 23/1728	19	Across	Landscape: Side-by-Side
P1M13280 <sup>20</sup>	12 x 8.5	11.0 x 7.5	66 at 8.8 lpi 66 at 8.8 lpi	15/72 15/1380	GT24	Across	Landscape: Side-by-Side
P1M132C1 <sup>21</sup>	12 x 8.5	7.5 x 11.0	66 at 12.1 lpi 66 at 12.1 lpi	15/0 next/0	GT20	Down	Portrait: Over/Under

The phrases *Side-by-Side* and *Over/Under* in Table 32 on page 186 describe the placement of the subpages on a single physical sheet. Figure 53 on page 187 shows how the subpages appear as separate logical pages on the physical sheet.

<sup>19</sup> P1M16080 prints 80 characters per line when it is using a 12-pitch font, and 100 characters per line when it is using a 15-pitch font.

<sup>20</sup> Font GT24 is specified in this page definition; any CHARS option is ignored.

<sup>21</sup> Font GT20 is specified in this page definition; any CHARS option is ignored. When used with traditional line data that contains no carriage controls, this page definition prints with no break between subpages.

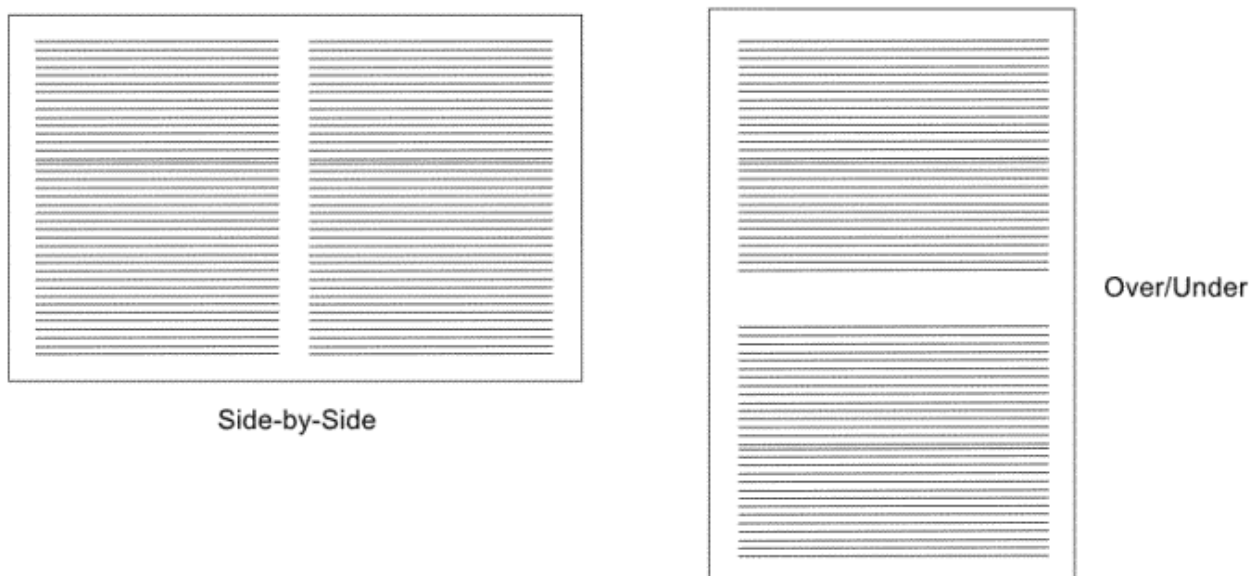


Figure 53. Placement of two subpages on a single physical sheet

## Page definitions for the 4224, 4230, 4234, 4247, and 6400 printers

Page definitions for the 4224, 4230, 4234, 4247, and 6400 printers are supplied for formatting some of the more common paper sizes, as the tables here show. These page definitions are designed for printing on continuous-forms paper on any of the printers named. They are not designed for printing with cut-sheet paper or with the document-on-demand feature.

The 4224, 4230, 4234, 4247, and 6400 printers support different fonts, depending on the print-quality level set for the printer. Therefore, select a font that is supported at the print-quality level set for your printer. To see the print-quality level set for your printer, see the documentation provided with the printer. Each of these page definitions assigns a channel-1 (carriage) control character to the first line of the page.

Table 33 on page 187 describes the page definitions for continuous-forms paper that measures 12 x 8.5 inches.

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1J04964	10.67 x 8.17	49 at 4 lpi	30/0	GT10	Across	Landscape
P1J06484	10.67 x 8.17	64 at 8.01 lpi	30/0	GT12	Across	Landscape

Table 34 on page 187 describes the page definitions for continuous-forms paper that measures 9.5 x 11 inches.

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1A06462	8.17 x 10.67	64 at 6 lpi	30/0	GT10	Across	Portrait
P1A08584	8.17 x 10.67	85 at 8.01 lpi	30/0	GT12	Across	Portrait

Table 35 on page 188 describes the page definitions for continuous-forms paper 14.88 x 11 inches. The width of the printable area in these page definitions is 13.2 inches because the 4224, 4230, 4234, 4247, and 6400 printers have a maximum line length of 13.2 inches. To print records that have a line length of 13.2 inches (for example, 132-byte records printed with a GT10 font), you must use a form definition that

positions the page at the left margin. For example, you can use form definition F1OGL, described in [Table 24 on page 181](#).

Table 35. Page definitions for continuous-forms paper 14.88 x 11 inches						
Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1L06464	13.2 x 10.67	64 at 6 lpi	30/0	GT10	Across	Landscape
P1L08584	13.2 x 10.67	85 at 8.01 lpi	30/0	GT12	Across	Landscape

## Page definitions for HP-CL4 and PPDS printers

Page definitions for HP-CL4 and PPDS printers are supplied for formatting some of the more common paper sizes, as the tables here show. For information about the paper sizes that are supported for your printer, see the publications for your printer.

Each of these page definitions specifies a channel-1 (carriage) control character to position at the top of the next page.

[Table 36 on page 188](#) describes the page definitions for cut-sheet A4 paper, which is 8.27 inches wide by 11.69 inches high.

Table 36. Page definitions for A4 paper						
Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1Q09182	7.77 x 11.29	91 at 8.2 lpi	25/0	GT12	Across	Portrait
P1X04763	10.60 x 7.77	47 at 6.1 lpi	30/0	GT10	Down	Landscape
P1X06483	10.60 x 7.77	64 at 8.2 lpi	24/0	GT12	Down	Landscape
P1X06683	10.60 x 7.77	66 at 8.5 lpi	24/224 (28)	GT15	Down	Landscape.

[Table 37 on page 188](#) describes the page definitions for cut-sheet B4 paper, which is 10.12 inches wide by 14.33 inches high.

Table 37. Page definitions for B4 paper						
Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1T08362	9.62 x 13.93	83 at 6 lpi	30/0	GT10	Across	Portrait
P1T11382	9.62 x 13.93	113 at 8.2 lpi	24/0	GT10	Across	Portrait
P1T05963	13.93 x 9.62	59 at 6.1 lpi	30/0	GT10	Down	Landscape
P1T07983	13.93 x 9.62	79 at 8.2 lpi	24/0	GT 12	Down	Landscape

[Table 38 on page 188](#) describes the page definitions for cut-sheet letter paper, which is 8.5 inches wide by 11 inches high.

Table 38. Page definitions for cut-sheet letter paper						
Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1P06362	8.00 x 10.60	63 at 6 lpi	30/0	GT10	Across	Portrait
P1P08682	8.00 x 10.60	86 at 8.2 lpi	24/0	GT12	Across	Portrait
P1X04763	10.60 x 7.77	47 at 6.1 lpi	30/0	GT10	Down	Landscape
P1X06483	10.60 x 7.77	64 at 8.2 lpi	24/0	GT12	Down	Landscape

Table 38. Page definitions for cut-sheet letter paper (continued)

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1X06683	10.60 x 7.77	66 at 8.5 lpi	24/224 (28)	GT15	Down	Landscape

Table 39 on page 189 describes the page definitions for cut-sheet legal paper, which is 8.5 inches wide by 14 inches high.

Table 39. Page definitions for cut-sheet legal paper

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1R08162	8/00 x 13.60	81 at 6 lpi	30/0	GT10	Across	Portrait
P1R11082	8.00 x 13.60	110 at 8.2 lpi	24/0	GT12	Across	Portrait
P1R04763	13.60 x 8.00	47 at 6 lpi	30/0	GT10	Down	Landscape
P1R06683	13.60 x 8.00	66 at 8.2 lpi	24/0	GT12	Down	Landscape

## Page definitions for all other printers supported by PSF

Table 40 on page 189 describes common page definitions supplied with PSF for all printers other than the 3800, 4224, 4230, 4234, 4247, 6400, PCL4, and PPDS printers. PSF provides these common page definitions to promote the interchange of documents between different printers.

PSF provides page definitions to format the commonly used cut-sheet and continuous-forms paper sizes. The tables here show the page definitions for each paper size. For information about the paper sizes that are supported for your printer, see the publications for your printer.

Each of these page definitions specifies a channel-1 (carriage) control character to position at the top of the next page.

Table 40 on page 189 describes the page definitions for cut-sheet A4 paper, which is 8.27 inches wide by 11.69 inches high.

Table 40. Page definitions for A4 paper

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1C09182	7.94 x 11.36	91 at 8.2 lpi	25/0	GT12	Across	Portrait
P1V04863	10.67 x 7.94	48 at 6.1 lpi	30/0	GT10	Down	Landscape
P1V06483	10.67 x 7.94	64 at 8.2 lpi	24/0	GT12	Down	Landscape
P1V06683	10.67 x 7.94	66 at 8.5 lpi	24/224 <sup>22</sup>	GT15	Down	Landscape

Table 41 on page 189 describes the page definitions for cut-sheet B4 paper, which is 10.12 inches wide by 14.33 inches high.

Table 41. Page definitions for B4 paper

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1D08462	9.79 x 14	84 at 6 lpi	30/0	GT10	Across	Portrait
P1D11382	9.79 x 14	113 at 8.2 lpi	24/0	GT12	Across	Portrait

<sup>22</sup> The user printable area is 9.74 x 7.94 because of the 224-logical-unit offset in Position Down/Position Across column.

Table 41. Page definitions for B4 paper (continued)

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
<b>P1D06063</b>	14 x 9.79	60 at 6.1 lpi	30/0	GT10	Down	Landscape
<b>P1D08083</b>	14 x 9.79	80 at 8.2 lpi	24/0	GT12	Down	Landscape

Table 42 on page 190 describes the page definitions for any of these paper sizes:

- Cut-sheet: Letter, which is 8.5 inches wide by 11 inches high
- Continuous-forms: 12 inches wide by 8.5 inches high
- Continuous-forms: 9.5 inches wide by 11 inches high

Table 42. Page definitions for letter and continuous-forms paper 12 x 8.5 inches or 9.5 x 11 inches

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
<b>P1A06462</b>	8.17 x 10.67	64 at 6 lpi	30/0	GT10	Across	Portrait
<b>P1TT6462</b>	8.17 x 10.67	64 at 6 lpi	30/0	TrueType WT Sans Duo height 10	Across	Portrait
<b>P1A08682</b>	8.17 x 10.67	86 at 8.2 lpi	24/0	GT12	Across	Portrait
<b>P1V04863</b>	10.67 x 7.94	48 at 6.1 lpi	30/0	GT10	Down	Landscape
<b>P1V06483</b>	10.67 x 7.94	64 at 8.2 lpi	24/0	GT12	Down	Landscape
<b>P1TT6483</b>	10.67 x 7.94	64 at 8.2 lpi	24/0	TrueType WT Sans Duo height 8	Down	Landscape
<b>P1V06683</b>	10.67 x 7.94	66 at 8.5 lpi	24/224 <sup>22</sup>	GT15	Down	Landscape

Table 43 on page 190 describes the page definitions for any of these paper sizes:

- Cut-sheet: Legal, which is 8.5 inches wide by 14 inches high
- Continuous-forms: 14.88 inches wide by 11 inches high

Table 43. Page definitions for legal and continuous-forms paper 14.88 x 11 inches

Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
<b>P1B08262</b>	8.17 x 13.67	82 at 6 lpi	30/0	GT10	Across	Portrait
<b>P1B11082</b>	8.17 x 13.67	110 at 8.2 lpi	24/0	GT12	Across	Portrait
<b>P1B04963</b>	13.67 x 8.17	49 at 6 lpi	30/0	GT10	Down	Landscape
<b>P1B06683</b>	13.67 x 8.17	66 at 8.2 lpi	24/0	GT12	Down	Landscape

Table 44 on page 191 describes the page definitions for multiple-up printing on any of these paper sizes:

- Cut-sheet: Letter, which is 8.5 inches wide by 11 inches high
- Cut-sheet: A4, which is 8.27 inches wide by 11.69 inches high
- Continuous-forms: 12 inches wide by 8.5 inches high
- Continuous-forms: 9.5 inches wide by 11 inches high



Table 44. Multiple-up page definitions						
Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1W120C2	7.94 x 10.67	60 at 12 lpi 60 at 12 lpi	16/160 1344/160	GT20	Across	Portrait: Over/Under
P1W12883	10.67 x 7.94	64 at 8.2 lpi 64 at 8.2 lpi	24/0 24/1281	GT15	Down	Landscape: Side-by-Side
P1W240F3	10.67 x 7.94	60 at 15.2 lpi 60 at 15.2 lpi 60 at 15.2 lpi	16/48 968/48 16/1322 968/1322	GT24	Down	Landscape: Over/Under Side-by-Side

The phrases *Side-by-Side* and *Over/Under* in Table 44 on page 191 describe the placement of the subpages on a single physical sheet. Figure 54 on page 191 shows how the subpages appear as separate logical pages on the physical page.

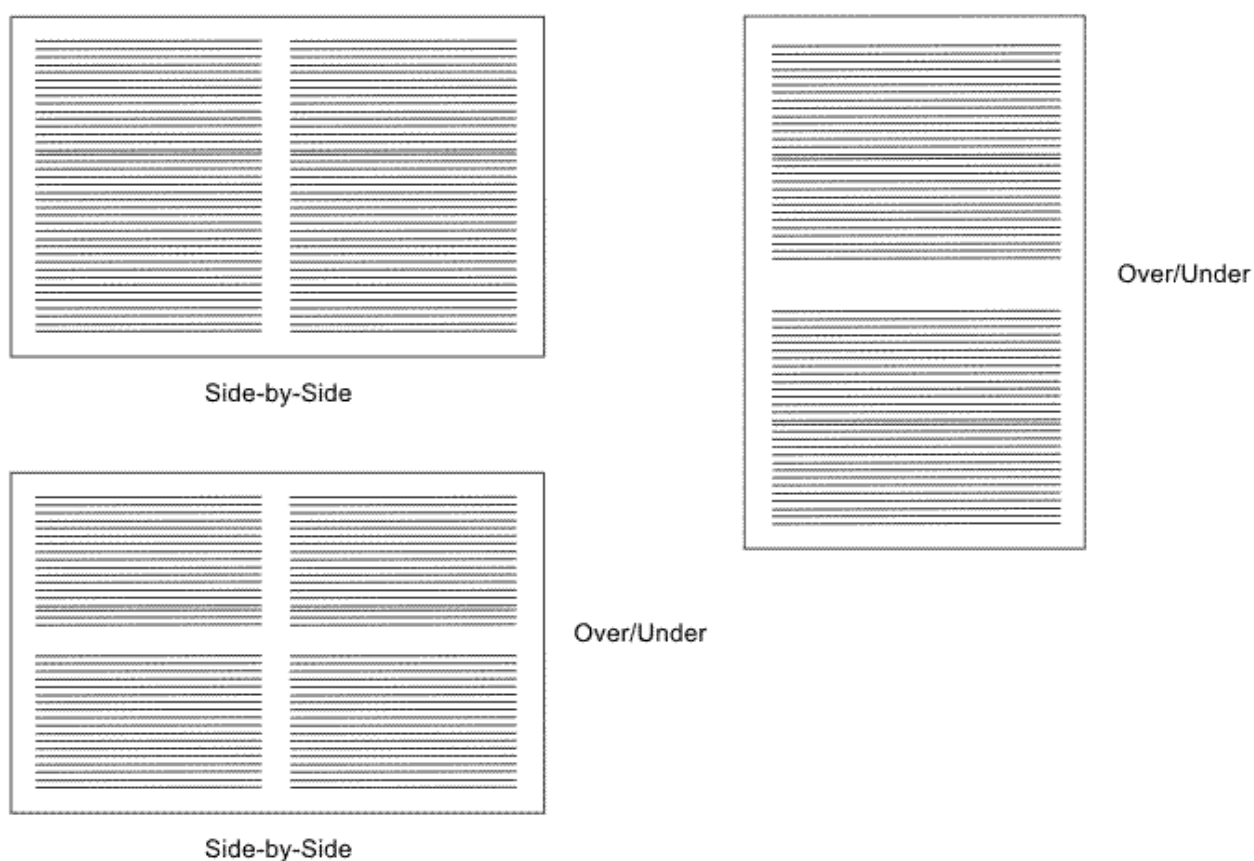


Figure 54. Placement of multiple-up logical pages on the physical sheets

Table 45 on page 191 lists page definitions for printing on various sizes of three-hole punched paper. The page definitions specify that the printing direction is Up, and they offset printing to accommodate three-hole punched paper. You must use these page definitions with the form definitions whose names begin with F1H.

Table 45. Page definitions for printing on three-hole punched paper						
Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1B0446B	13.67 x 7.34	44 at 6 lpi	30/0	GT10	Up	Landscape
P1B0608B	13.67 x 7.34	60 at 8.2 lpi	24/0	GT12	Up	Landscape

Table 45. Page definitions for printing on three-hole punched paper (continued)						
Page Definition Name	Printable Area: Width by Height, in Inches	Print Lines Per Page	Page Position: <sup>18</sup> Down/Across	Recommended Font	Printing Direction	Page Presentation
P1D0556B	14.0 x 8.96	55 at 6.1 lpi	30/0	GT10	Up	Landscape
P1D0748B	14.0 x 8.96	74 at 8.2 lpi	24/0	GT12	Up	Landscape
P1V0436B	10.67 x 7.11	43 at 6.1 lpi	30/0	GT10	Up	Landscape
P1V0588B	10.67 x 7.11	58 at 8.2 lpi	24/0	GT12	Up	Landscape
P1V0608B	10.67 x 7.11	60 at 8.5 lpi	24/224	GT15	Up	Landscape
P1W1168B	10.67 x 7.11	58 at 8.2 lpi 58 at 8.2 lpi	24/0 24/1281	GT15	Up	Landscape
P1W216FB	10.67 x 7.11	54 at 15.2 lpi 54 at 15.2 lpi 54 at 15.2 lpi 54 at 15.2 lpi	16/48 890/48 16/1322 890/1322	GT24	Up	Landscape

## Page definitions for printing PSF reports

Table 46 on page 192 describes the page definitions that you can use to format PSF reports for printing.

Table 46. Page definitions for printing PSF reports	
Page Definition Name	Report Type
P1ASAP04	AFP Statistics (AFPSTATS)
P1DPI01	Printer Information
P1PPCT01	Tier Level (produced by the Point Counting Tool)

## Page definition line-spacing values and fonts

The tables here list the line-spacing values for some of the page definitions supplied with PSF. The tables also list FOCA fonts by group according to the lines per inch at which each font is printed.

Table 47 on page 192 shows the line-spacing values for the 3800 printer.

Table 47. Cross-reference of line spacing and page definitions for the 3800 printer	
Line Spacing	Page Definition Name
6 lpi	P104560 P105161 P106060 P106061 P1L06060 P1M12060 P1STD1 P1STD2 P16
8 lpi	P106080 P106881 P108080 P108081 P1L08080 P1M16080 P1STD3 P18

Table 47. Cross-reference of line spacing and page definitions for the 3800 printer (continued)

Line Spacing	Page Definition Name
<b>10 lpi</b>	P1075A0 P1085A1 P1100A0 P1100A1 P1L100A0
<b>12 lpi</b>	P1090C0 P1120C0 P1102C1 P1L120C0
<b>12.5 lpi</b>	P1M120C0 P1M120C1
<b>Special</b>	P1M13280 <sup>23</sup> P1M132C1 <sup>24</sup>

Table 48 on page 193 shows the line-spacing values for each of the page definitions for the 4224, 4230, 4234, 4247, and 6400 printers.

Table 48. Cross-reference of line spacing and page definitions for the 4224, 4230, 4234, 4247, and 6400 printers

Line Spacing	Page Definition Name
<b>6 lpi</b>	P1J04964 P1J06462 P1L06464
<b>8.01 lpi</b>	P1J06484 P1A08584 P1L08584

Table 49 on page 193 shows the line-spacing values for each of the page definitions for cut-sheet A4 paper for the PCL4 and PPDS printers.

Table 49. Cross-reference of line spacing and page definitions for cut-sheet A4 paper for the PCL4 and PPDS printers

Line Spacing	Page Definition Name
<b>6.1 lpi</b>	P1X04763
<b>8.2 lpi</b>	P1Q09182 P1X06483
<b>8.5 lpi</b>	P1X06683

Table 50 on page 193 shows the line-spacing values for each of the page definitions for cut-sheet B4 paper for the PCL4 and PPDS printers.

Table 50. Cross-reference of line spacing and page definitions for cut-sheet B4 paper for the PCL4 and PPDS printers

Line Spacing	Page Definition Name
<b>6 lpi</b>	P1T08362
<b>6.1 lpi</b>	P1T05963

<sup>23</sup> P1M13280 specifies font GT24, which is printed at 8.8 lines per inch. Any CHARS option is ignored.

<sup>24</sup> Fonts AE20 and GT20 are printed at 12.8 lines per inch. Page definition P1M132C1 defines printing at 12.1 lines per inch, specifying font GT20. If you specify any other font with a CHARS option, the specification is ignored. To print an application with font AE20, use PPFA to change the source for P1M132C1, specifying font AE20. Rename the object and store it in the system page definition library.

*Table 50. Cross-reference of line spacing and page definitions for cut-sheet B4 paper for the PCL4 and PPDS printers (continued)*

<b>Line Spacing</b>	<b>Page Definition Name</b>
<b>8.2 lpi</b>	P1T11382 P1T07983

Table 51 on page 194 shows the line-spacing values for each of the page definitions for all other printers.

*Table 51. Cross-reference of line spacing and page definitions for other printers*

<b>Line Spacing</b>	<b>Page Definition Name</b>
<b>6 lpi</b>	P1D08462 P1B08262 P1B04963 P1A06462 P1B0446B
<b>6.1 lpi</b>	P1D06063 P1V04863 P1D0556B P1V0463B
<b>8.2 lpi</b>	P1C09182 P1D11382 P1D08083 P1A08682 P1V06483 P1B11082 P1B06683 P1W12883 P1D0748B P1V0588B P1W1168B
<b>8.5 lpi</b>	P1V06683 P1V0608B
<b>12 lpi</b>	P1W120C2
<b>15.2 lpi</b>	P1W240F3 P1W216FB

Any font that can be printed at the specified line spacing (or at a larger line-spacing value) can be used with a page definition. Table 52 on page 195 describes the line spacing for commonly used monospace FOCA fonts provided with PSF.

Table 52. Cross-reference of line spacing and commonly used PSF monospaced fonts

Line Spacing	FOCA Fonts
<b>6 lpi</b>	CE10 CE12 CI10 CR10 GB10 GF10 GS10 GT10 LB12 LR12 PB12 PI12 PR10 PR12
<b>8 lpi</b>	GB12 GF12 GF15 GI12 GP12 GS12 GS15 GT12 GT15
<b>10 lpi</b>	GC15
<b>12 lpi</b>	GFC GSC GUC
<b>Special</b>	AE20 <sup>25</sup> GT20 <sup>25</sup> GT24 <sup>26</sup>

For more information about these fonts or about fonts that are not listed in this table, see the font technical references and the font samples publications listed in [“Bibliography” on page 277](#).

<sup>25</sup> Fonts AE20 and GT20 are printed at 12.8 lines per inch. Font GT20 is specified in page definition P1M132C1. To print an application with font AE20, you can use the PPFA program to change the source for P1M132C1 to specify font AE20. Rename the page definition and store it in the system page definition library.

<sup>26</sup> Font GT24 is printed at 8.8 lines per inch as defined in page definition P1M13280.



## Appendix C. AFPSTATS report

This information contains softcopy record details for the AFPSTATS report and sample softcopy and hardcopy reports generated by PSF.

## Softcopy record details

This record gives detailed information about the AFPSTATS softcopy report format. It lists each record that can be in the report, along with a detailed description. These records are written by both PSF and AFP Download Plus.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: COMMENT
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	499	Record comment up to 499 characters.

characters.

REPORTLVL record identifies the AFP Statistics report level, which can be used to identify changes to the AFP Statistics report format.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: REPORTLVL
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Report ID. Value: AFPSTATS
21	(15)	1	Column separator.
22	(17)	8	PSF version. Example: 4.6.0
30	(1E)	1	Column separator.
31	(1F)	4	Report level. Example: 0001
35	(23)	1	Column separator.
36	(24)	30	Report description. Value: PSF AFP Statistics Report

TITLE record provides report titles so the report is more readable.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: TITLE
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	499	Record title up to 499 characters.

Values:

Print File Information.  
Print File Extension Information.  
Processing Detail.  
Resource Summary by Name.  
Resource Summary by Data

Set.

Resource Summary by Resource

Type.

Processing Summary.

Figure 55. Softcopy record details (Part 1 of 18)

SECTION record identifies report sections so the report is more readable.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: SECTION
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	499	Record section up to 499 characters.
Values:			Disposition Summary. Inline in Print File. Inline Metadata. Location Summary. Printer Resident. PSF Default. Reference Summary. Security Libraries. Significant Events. Summary of Events. Summary of Pages. System Libraries. System Path. Unused Inline Resource. User Libraries.

HEADING record provides column headings for the softcopy report.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Values: HEADING HEADINGP HEADINGR HEADING-DP HEADING-EL HEADING-LC HEADING-PM HEADING-RF HEADING-SE HEADING-SP HEADING-UU
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	499	Record heading up to 499 characters. Values vary by HEADING record.

PAGEFOOTER record provides footing text for the hardcopy report.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: PAGEFOOTER
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	499	Record footer up to 499 characters.
Values:			Print File Information. Print File Extension Information. Processing Detail. Resource Summary by Name. Resource Summary by Data Set. Resource Summary by Resource Type. Processing Summary.

Figure 56. Softcopy record details (Part 2 of 18)



NOTE record provides notes at appropriate places in the softcopy report.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: NOTE
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	499	Record note up to 499 characters.

Value:

The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.

PRINTFILE record identifies the print file. There is one PRINTFILE record at the beginning of each softcopy report.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: PRINTFILE
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	JES job identifier.
21	(15)	1	Column separator.
22	(16)	8	Job name for the print file from
JCL.			
30	(1E)	1	Column separator.
31	(1F)	8	Step name for the print file from
JCL.			
39	(27)	1	Column separator.
40	(28)	18	Data source. Values: Application direct Deferred-spool
58	(3A)	1	Column separator.
59	(3B)	10	Print date in mm/dd/yyyy
format.			
69	(45)	1	Column separator.
70	(46)	10	Print time in hh:mm:ss
format.			
80	(50)	1	Column separator.
81	(51)	20	PSF version and release.
101	(65)	1	Column separator.
102	(66)	18	PSF attachment. Values: Deferred-printing Direct-printing
120	(78)	1	Column separator.
121	(79)	8	Printer name.
129	(81)	24	Printer type name.
153	(99)	1	Column separator.
154	(9A)	8	System name.
162	(A2)	1	Column separator.
163	(A3)	12	CPU identifier.
175	(AF)	1	Column separator.
176	(B0)	7	Printer type and model.
183	(B7)	1	Column separator.

**Note:** The PRINTFILE record is generated only for PSF 4.6.0.

Figure 57. Softcopy record details (Part 3 of 18)

184 printer.	(B8)	12	Intermediate device between PSF and Values: DPF None PSF Direct RPM2 RPM3 WPM
196	(C4)	1	Column separator.
197	(C5)	10	Printer physical attachment. Values: CHANNEL TCP/IP VTAM
207	(CF)	1	Column separator.
208	(D0)	12	Printer port number.
220	(DC)	1	Column separator.
221	(DD)	64	Printer identifier: IP address, LU name, or
CUU.			
285	(11D)	1	Column separator.
286	(11E)	30	PSF: Reserved. AFP Download Plus (AFPDP): Remote system
name.			
316	(13C)	1	Column separator.
317	(13D)	30	PSF: Reserved. AFPDP: Remote operating system.
347	(15B)	1	Column separator.
348	(15C)	30	PSF: Reserved. AFPDP: Remote system version.
378	(17A)	1	Column separator.
379	(17B)	30	PSF: Reserved. AFPDP: Remote system compile date.
409	(199)	1	Column separator.
410	(19A)	15	Interchange set for print file. Values: AFP/A AFP/A,
IS/3			IS/3 Not Specified Unknown

Figure 58. Softcopy record details (Part 4 of 18)

PRINTFILEP record identifies the print file. There is one PRINTFILEP record at the beginning of each softcopy report.

=====			
=====			
OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	=====
0	(0)	10	Layout ID. Value: PRINTFILEP
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	JES job identifier.
21	(15)	1	Column separator.
22	(16)	8	Job name for the print file from
JCL.			
30	(1E)	1	Column separator.
31	(1F)	8	Step name for the print file from
JCL.			
39	(27)	1	Column separator.
40	(28)	18	Data source. Values: Application direct Deferred-spool
58	(3A)	1	Column separator.
59	(3B)	10	Print date in mm/dd/yyyy
format.			
69	(45)	1	Column separator.
70	(46)	10	Print time in hh:mm:ss
format.			
80	(50)	1	Column separator.
81	(51)	20	PSF version and release.
101	(65)	1	Column separator.
102	(66)	18	PSF attachment. Values: Deferred-printing Direct-printing
120	(78)	1	Column separator.
121	(79)	7	Printer name.
128	(80)	1	Column separator.
129	(81)	24	Printer type name.
153	(99)	1	Column separator.
154	(9A)	8	System name.
162	(A2)	1	Column separator.
163	(A3)	12	CPU identifier.
175	(AF)	1	Column separator.
176	(B0)	7	Printer type and model.
183	(B7)	1	Column separator.
184	(B8)	12	Intermediate device between PSF and
printer.			
			Values: DPF None PSF Direct RPM2 RPM3 WPM
196	(C4)	1	Column separator.
197	(C5)	10	Printer physical attachment. Values: TCP/IP VTAM
207	(CF)	1	Column separator.
208	(D0)	12	Printer port number.
220	(DC)	1	Column separator.
221	(DD)	64	Printer identifier: IP address, LU name, or
CUU.			
285	(11D)	1	Column separator.
286	(11E)	15	Interchange set for print file. Values: AFP/A AFP/A,
IS/3			
			IS/3 Not Specified Unknown

Figure 59. Softcopy record details (Part 5 of 18)

RESOURCE record identifies the first use of one unique resource for this print file.			
=====			
OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	
0	(0)	10	Layout ID. Value: RESOURCE
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Resource member name.
21	(15)	1	Column separator.
22	(16)	18	Resource type. Values:
			Character Set
			Coded Font
			Code Page
			Form Definition
			Object Container
			Overlay
			Page Definition
			Page Segment
40	(28)	1	Column separator.
41	(29)	8	Library type where the resource was found.
Values:			Inline
			PSF default
			Resident
			Security
			System
			User
49	(31)	1	Column separator.
50	(32)	14	Size of incoming resource in bytes.
64	(40)	1	Column separator.
65	(41)	14	Relative page number in this document.
79	(4F)	1	Column separator.
80	(50)	44	Data set name where resource was found
(blank			when resource is inline, default, or
resident).			
124	(7C)	1	Column separator.
125	(7D)	6	Volume serial number.
131	(83)	1	Column separator.
132	(84)	20	How resource was handled.
Values:			Activated
			Already processed
			Download
			Integrated into page
			PSF memory
			?
152	(98)	1	Column separator.
153	(99)	14	Transmission count.
167	(A7)	1	Column separator.
168	(A8)	14	Page number.

Figure 60. Softcopy record details (Part 6 of 18)

RESOURCEGR record identifies the first use of one unique GRID font for this print file.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: RESOURCEGR
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Global resource identifier (GRID). If > 8 characters, first 5 characters and '...'
show name			
21	(15)	1	continuation.
22	(16)	18	Column separator.
Font			
40	(28)	1	Resource type. Value: GRID
41	(29)	8	Column separator.
Resident			
49	(31)	1	Library type. Value:
50	(32)	14	Column separator.
64	(40)	1	Size of incoming resource in bytes.
65	(41)	14	Column separator.
79	(4F)	1	Relative page number in this document.
80	(50)	9	Column separator.
89	(59)	16	Resource keyword. Value: Resource=
105	(69)	1	Resource GRID.
106	(6A)	13	Column separator.
119	(77)	14	Transmission keyword. Value: Transmission
133	(85)	1	Transmission count.
134	(86)	12	Column separator.
146	(92)	14	Page number keyword. Value: Page number=
			Page number.

RESOURCEDF record identifies the first use of one unique resource that can have a name longer than eight characters for this print file.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: RESOURCEDF
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Resource name. If > 8 characters, first 5 characters and '...' show name continuation.
21	(15)	1	Column separator.
22	(16)	18	Resource type. Value: True/Open
Type			
40	(28)	1	Column separator.
41	(29)	8	Library type. Values:
			Inline
			Resident
			Security
			System
			User
			?
49	(31)	1	Column separator.
50	(32)	14	Size of incoming resource in bytes.
64	(40)	1	Column separator.
65	(41)	14	Relative page number in this document.
79	(4F)	1	Column separator.
80	(50)	6	Font collection index keyword. Value: Index=
86	(56)	4	Font index or "N/A".
90	(5A)	1	Column separator.
91	(5B)	13	Font relationship keyword. Value:
Relationship=			
104	(68)	10	Relationship of font. Values:
			Base Font
			Linked Font

Figure 61. Softcopy record details (Part 7 of 18)

114	(72)	1	Column separator.
115	(73)	12	Font disposition keyword. Value:
Disposition=			
127	(7F)	17	Disposition of the font. Values:
			Activate
			Already Processed
			Download
144	(90)	1	Column separator.
145	(91)	9	Resource keyword. Value: Resource=
154	(9A)	90	Resource long name.
244	(F4)	10	Resource file name keyword. Value: File
Name=			
254	(FE)	90	Resource file name including extension, if
any.			
344	(158)	5	Resource path keyword. Value: Path=
349	(15D)	100	Resource path.
449	(1C1)	1	Column separator.
450	(1C2)	13	Transmission keyword. Value: Transmission
463	(1CF)	14	Transmission count.
477	(1DD)	1	Column separator.
478	(1DE)	12	Page number keyword. Value: Page number=
490	(1EA)	14	Page number.

RESOURCE0 record identifies the first use of one data object resource for this print file.

```
=====
=====
=====
```

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	
0	(0)	10	Layout ID. Value: RESOURCE0
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Resource name. If > 8 characters, first 5 characters and '...' show name continuation.
21	(15)	1	Column separator.
22	(16)	18	Resource type. Values:
			Code Page
			Object Container
40	(28)	1	Column separator.
41	(29)	8	Library type. Values:
			Inline
			PSF Default
			Resident
			Security
			System
			User
			?
49	(31)	1	Column separator.
50	(32)	14	Size of incoming resource in bytes.
64	(40)	1	Column separator.
65	(41)	14	Relative page number in this print file.
79	(4F)	1	Column separator.
80	(50)	12	Disposition keyword. Value:
Disposition=			
92	(5C)	17	How resource was handled.
Values:			
			Activated
			Already processed
			Download
			Integrated into page
			PSF memory
			?
109	(6D)	1	Column separator.
110	(6E)	9	Resource name keyword. Value: Resource=
119	(77)	90	Resource long name.
209	(D1)	1	Column separator.
210	(D2)	10	Resource file name keyword. Values:
			Data Set=
			File Name=

Figure 62. Softcopy record details (Part 8 of 18)

220	(DC)	90	Resource file name, "?", or "Internal".
310	(136)	1	Column separator.
311	(137)	5	Path keyword. Values: Path= VSER=
316	(13C)	100	Path of the font, "?", or "n/a".
416	(1A0)	1	Column separator.
417	(1A1)	13	Transmission keyword.
430	(1AE)	14	Transmission count.
444	(1BC)	1	Column separator.
445	(1BD)	12	Page number keyword.
457	(1C9)	14	Page number.
471	(1D7)	1	Column separator.
472	(1D8)	9	Color management resource (CMR) scope
keyword.			
481	(1E1)	9	Value: CMRScope= CMR scope. Values: Document Object Overlay Printfile Sheet n/a
490	(1EA)	1	Column separator.
491	(1EB)	10	CMR PRMODE keyword. Value: CMRPRMode=
501	(1F5)	11	CMR processing mode. Values: Audit Instruction Link n/a

EVENT records identify significant processing exceptions that occur while processing the print file.

```
=====
```

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	
0	(0)	10	Layout ID. Value: EVENT
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	6	Event keyword. Value:
Event=			
19	(13)	24	Event identification. Values: Early termination Misplaced Metadata Misplaced resource group Processing complete PSF - Reclaiming HAIDS Repositioning Resource not found Terminate Page
43	(2B)	1	Column separator.
44	(2C)	5	Page number keyword. Value: Page=
49	(31)	14	Relative page number.
63	(3F)	1	Column separator.
64	(40)	9	Resource name keyword. Value:
Resource=			

Figure 63. Softcopy record details (Part 9 of 18)

73	(49)	8	Resource name or "?".
81	(51)	1	Column separator.
82	(52)	13	Transmission keyword. Value: Transmission=
95	(5F)	14	Transmission count.
109	(6D)	1	Column separator.
110	(6E)	12	Page number keyword. Value: Page number=
122	(7A)	14	Page number.

EVENTRL records identify resource reload event requests that occur during EXIT 7 processing.

```
=====
=====
  OFFSET  OFFSET
  DECIMAL  HEX
  =====  =====
  LENGTH
  =====
  =====
  0        (0)        10      Layout ID. Value: EVENTRL
  10       (A)         2      Reserved.
  12       (C)         1      Column separator.
  13       (D)         6      Event keyword. Value: Event=
  19      (13)        24      Event ID. Values:
                             Complete
                             Ignored-Can not unload
                             Ignored-GRID font n/s
                             Ignored-Included n/s
                             Ignored-Inline resource
                             Ignored-OC reload n/s
                             Ignored-Required res
                             Ignored-Resource in OEG
                             Ignored-Select del n/s
  43       (2B)         1      Column separator.
  44       (2C)         5      Page number keyword. Value: Page=
  49       (31)        14      Relative page number.
  63       (3F)         1      Column separator.
  64       (40)         9      Resource name keyword. Value: Resource=
  73       (49)        19      Resource name.
  92       (5C)         1      Column separator.
  93       (5D)        14      Resource type keyword. Value: Resource Type=
  107      (6B)        18      Resource type. Values:
                             Character Set
                             Coded Font
                             Code Page
                             Object Container
                             Overlay
                             Page Segment
  125      (7D)         1      Column separator.
  126      (7E)        13      Transmission keyword. Value: Transmission
  139      (8B)        14      Transmission count.
  153      (99)         1      Column separator.
  154      (9A)        12      Page number keyword. Value: Page number=
  166      (A6)        14      Page number.
```

EVENTFM records identify when PSF maps one type of supported font for another type of supported font.

```
=====
=====
  OFFSET  OFFSET
  DECIMAL  HEX
  =====  =====
  LENGTH
  =====
  =====
  0        (0)        10      Layout ID. Value: EVENTFM
  10       (A)         2      Reserved.
  12       (C)         1      Column separator.
  13       (D)         6      Event keyword. Value: Event=
  19      (13)        24      Event ID. Values:
                             Map GRID to Outline
                             Map GRID to Raster
                             Map Outline to Raster
                             Mapped CPGID to CP name
                             Map Raster to Outline
```

Figure 64. Softcopy record details (Part 10 of 18)



43	(2B)	1	Column separator.
44	(2C)	5	Page number keyword. Value: Page=
49	(31)	14	Relative page number.
63	(3F)	1	Column separator.
64	(40)	14	Resource name keyword. Value: From Resource=
78	(4E)	19	Original resource name, GRID name, or the
value			"CP/FN Pair".
97	(61)	1	Column separator.
98	(62)	3	Code page keyword. Value: CP=
101	(65)	8	Code page name.
109	(6D)	1	Column separator.
110	(6E)	3	Font name keyword. Value: FN=
113	(71)	8	Character set name.
121	(79)	1	Column separator.
122	(7A)	12	Resource name keyword. Value: To Resource=
134	(86)	19	Mapped resource name, GRID name, or the
value			"CP/FN Pair".
153	(99)	1	Column separator.
154	(9A)	3	Code page keyword. Value: CP=
157	(9D)	8	Code page name.
165	(A5)	1	Column separator.
166	(A6)	3	Font name keyword. Value: FN=
169	(A9)	8	Character set name.
177	(B1)	1	Column separator.
178	(B2)	13	Transmission keyword. Value: Transmission=
191	(BF)	14	Transmission count.
205	(CD)	1	Column separator.
206	(CE)	12	Page number keyword. Value: Page number=
218	(DA)	14	Page number.

EVENTRS records identify resource substitution event requests that occur during EXIT 7 processing.

=====			
=====			
OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====			
0	(0)	10	Layout ID. Value: EVENTRS
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	6	Event keyword. Value: Event=
19	(13)	24	Event ID. Values: Ignored-Inline resource Ignored-OC map by name Ignored-OC map by OID Ignored-OC not system Ignored-OC secondary Ignored-User library Substitution successful
43	(2B)	1	Column separator.
44	(2C)	5	Page number keyword. Value: Page=
49	(31)	14	Relative page number.
63	(3F)	1	Column separator.
64	(40)	14	Resource name keyword. Value: From
Resource=			Original resource name or GRID.
78	(4E)	19	Column separator.
97	(61)	1	Resource name keyword. Value: To Resource=
98	(62)	12	Substituted resource name or GRID.
110	(6E)	19	Column separator.
129	(81)	1	Resource type keyword. Value: Resource
130	(82)	14	Resource type. Values: Character Set Coded Font Code Page Form Definition
Type=			
144	(90)	18	

Figure 65. Softcopy record details (Part 11 of 18)

			Object Container
			Overlay
			Page Definition
			Page Segment
162	(A2)	1	Column separator.
163	(A3)	13	Transmission keyword. Value: Transmission=
176	(B0)	14	Transmission count.
190	(BE)	1	Column separator.
191	(BF)	12	Page number keyword. Value: Page number=
203	(CB)	14	Page number.

SUMM-NAME records contain information about a resource by resource name.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: SUMM-NAME
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Resource member name.
21	(15)	1	Column separator.
22	(16)	18	Resource type.
Values:			Character Set
			Coded Font
			Code Page
			Form Definition
			Object Container
			Overlay
			Page Definition
			Page Segment
40	(28)	1	Column separator.
41	(29)	12	Library type. Values:
			Inline
			PSF Default
			Resident
			Security
			System
			User
			?
53	(35)	1	Column separator.
54	(36)	14	Total references to this resource.
68	(44)	1	Column separator.
69	(45)	14	Number of mapped references.
83	(53)	1	Column separator.
84	(54)	14	Number of included references.
98	(62)	1	Column separator.
99	(63)	14	Number of JCL references to this resource.
113	(71)	1	Column separator.
114	(72)	14	Number of other references.
128	(80)	1	Column separator.
129	(81)	14	Number of resources integrated on the page.
143	(8F)	1	Column separator.
144	(90)	14	Number of resources downloaded to the printer.
158	(9E)	1	Column separator.
159	(9F)	14	Number activated as printer resident resources.
173	(AD)	1	Column separator.
174	(AE)	14	Number of resources loaded into PSF memory only.
188	(BC)	1	Column separator.
189	(BD)	14	Number of resources processed by prior reference.

Figure 66. Softcopy record details (Part 12 of 18)

SUMM-NAMEX record contains information about resources that can have a name longer than eight characters.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	
0	(0)	10	Layout ID. Value: SUMM-NAMEX
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Resource name. If > 8 characters, first 5
characters			and '...' show name continuation.
21	(15)	1	Column separator.
22	(16)	18	Resource type. Values:
			Object Container
			GRID Font
			True/Open Type
40	(28)	1	Column separator.
41	(29)	12	Library type. Values:
			Inline
			PSF Default
			Resident
			System
			User
			?
53	(35)	1	Column separator.
54	(36)	14	Total references to this resource.
68	(44)	1	Column separator.
69	(45)	14	Number of mapped references.
83	(53)	1	Column separator.
84	(54)	14	Number of included references.
98	(62)	1	Column separator.
99	(63)	14	Number of JCL references to this resource.
113	(71)	1	Column separator.
114	(72)	14	Number of other references.
128	(80)	1	Column separator.
129	(81)	14	Number of resources integrated on the page.
143	(8F)	1	Column separator.
144	(90)	14	Number of resources downloaded to the
printer.			
158	(9E)	1	Column separator.
159	(9F)	14	Number activated as printer resident
resources.			
173	(AD)	1	Column separator.
174	(AE)	14	Number of resources loaded into PSF memory
only.			
188	(BC)	1	Column separator.
189	(BD)	14	Number of resources processed by prior
reference.			
203	(CB)	1	Column separator.
204	(CC)	9	Resource keyword. Value: Resource=
213	(D5)	125	Resource long name.

SUMM-DSN records identify the data sets from which resources are obtained while processing this print file.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	
0	(0)	10	Layout ID. Value: SUMM-DSN
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	9	Data set keyword. Value: Data set=
22	(16)	44	Data set name.
66	(42)	9	Column separator.
75	(4B)	8	Volume keyword. Value: VOL=SER=
83	(53)	6	Volume serial number.

Figure 67. Softcopy record details (Part 13 of 18)

SUMM-DSN-R records identify the resources found in each data set while processing this print file.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: SUMM-DSN-R
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	8	Resource name.
21	(15)	4	Column separator.
25	(19)	8	Resource name.
33	(21)	4	Column separator.
37	(25)	8	Resource name.
45	(2D)	4	Column separator.
49	(31)	8	Resource name.
57	(39)	4	Column separator.
61	(3D)	8	Resource name.
69	(45)	4	Column separator.
73	(49)	8	Resource name.
81	(51)	4	Column separator.

SUMM-PATH records identify the paths from which resources are obtained while processing this print file.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: SUMM-PATH
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	5	Path keyword. Value: Path=
18	(12)	255	Path name.

SUMM-PTH-R records identify the resources found in each path while processing this print file.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: SUMM-PTH-R
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	9	Resource keyword. Value: Resource=
22	(16)	125	Resource name.

SUMM-REF record identifies the totals for all resource references by resource type.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: SUMM-REF
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	18	Resource type.

Values:

All resource types  
Character Set  
Coded Font  
Code Page  
Form Definition  
Object Container  
Overlay  
Page Definition  
Page Segment  
True/Open Type

Figure 68. Softcopy record details (Part 14 of 18)

	31	(1F)	1	Column separator.
type.	32	(20)	14	Number of unique resources of this
	46	(2E)	1	Column separator.
this type.	47	(2F)	14	Number of JCL references for resources of
	61	(3D)	1	Column separator.
of	62	(3E)	14	Number of mapped references for resources
	76	(4C)	1	this type.
of	77	(4D)	14	Column separator.
	91	(5B)	1	Number of included references for resources
	92	(5C)	14	this type.
	106	(6A)	1	Column separator.
	107	(6B)	14	Number of other references for resources of
				this type.
				Column separator.
				Number of total references for resources of
				this type.
SUMM-LOC record identifies the totals for where resources were found for each resource				
type.				
=====				
=====				
	OFFSET	OFFSET		
	DECIMAL	HEX	LENGTH	DESCRIPTION
	=====	=====	=====	
	0	(0)	10	Layout ID. Value: SUMM-LOC
	10	(A)	2	Reserved.
	12	(C)	1	Column separator.
Values:	13	(D)	18	Resource type.
				All resource types
				Character Set
				Coded Font
				Code Page
				Form Definition
				Object Container
				Overlay
				Page Definition
				Page Segment
				True/Open Type
	31	(1F)	1	Column separator.
type.	32	(20)	14	Number of unique resources of this
	46	(2E)	1	Column separator.
this type.	47	(2F)	14	Number of inline resources of this resource
	61	(3D)	1	Column separator.
type.	62	(3E)	14	Number of user resources of this resource
	76	(4C)	1	Column separator.
resource	77	(4D)	14	Number of security resources of this
	91	(5B)	1	type.
	92	(5C)	14	Column separator.
type.	106	(6A)	1	Number of system resources of this resource
	107	(6B)	14	Column separator.
resource				Number of PSF default resources of this
	121	(79)	1	type.
this type.	122	(7A)	14	Column separator.
				Number of printer resident resources of

Figure 69. Softcopy record details (Part 15 of 18)

SUMM-DISP record identifies the totals of how the resource requests were handled for each resource type.

=====			
=====			
OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	=====
0	(0)	10	Layout ID. Value: SUMM-DISP
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	18	Resource type.
Values:			
			All resource types
			Character Set
			Coded Font
			Code Page
			Form Definition
			Object Container
			Overlay
			Page Definition
			Page Segment
			True/Open Type
31	(1F)	1	Column separator.
32	(20)	14	Number of unique resources of this
type.			
46	(2E)	1	Column separator.
47	(2F)	14	Number of soft resources that were
integrated with			
printer.			
61	(3D)	1	Column separator.
62	(3E)	14	Number of resources downloaded to the
printer as a			
resource.			
76	(4C)	1	Column separator.
77	(4D)	14	Number of resources activated instead of
downloaded			
to the printer.			
91	(5B)	1	Column separator.
92	(5C)	14	Number of resources that are not printer
objects			
and only get loaded into PSF memory.			
106	(6A)	1	Column separator.
107	(6B)	14	Number of resources that had already been
processed.			

Figure 70. Softcopy record details (Part 16 of 18)

SUMM-PAGE record Contains statistics about the page data for the print file.			
=====			
=====			
OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	=====
0	(0)	10	Layout ID. Value: SUMM-PAGE
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	14	Total number of pages in the source print
file.			
27	(1B)	4	Column separator.
31	(1F)	14	Number of records in the source print
file.			
45	(2D)	4	Column separator.
49	(31)	14	Number of bytes in the source print file.
63	(3F)	4	Column separator.
67	(43)	14	Average source file page size in
bytes.			
81	(51)	4	Column separator.
85	(55)	14	Smallest source file page size in
bytes.			
99	(63)	4	Column separator.
103	(67)	14	Largest source file page size in
bytes.			
117	(75)	4	Column separator.
121	(79)	14	Relative page number of the smallest page.
135	(87)	4	Column separator.
139	(8B)	14	Relative page number of the largest page.
153	(99)	4	Column separator.
157	(9D)	14	PSF: Reserved.
			AFP Download Plus: Transformed bytes.
EVENT-LIST records provide information about one significant event that occurred while processing this print file.			
=====			
=====			
OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
=====	=====	=====	=====
0	(0)	10	Layout ID. Value: EVENT-LIST
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	14	Relative page number.
27	(1B)	1	Column separator.
28	(1C)	24	Event identification. Values vary by EVENT,
EVENTFM,			
			EVENTRL, and EVENTRS
records.			
52	(34)	1	Column separator.
53	(35)	63	Offending resource name, GRID number, or
"?".			

Figure 71. Softcopy record details (Part 17 of 18)

UIR-LIST records identify unused inline resources.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: UIR-LIST
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	18	Resource type. Value: Character Set Coded Font Code Page Form Definition Object Container Overlay Page Definition Page Segment True/Open Type NONE ?
31	(1F)	1	Column separator.
32	(20)	125	Resource name.

PRTFILEX record identifies the print file extension - There might be PRTFILEX records following the PRINTFILE information record.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	20	Interchange set for printfile. Values: AFP/A AFP/A,
IS/3			IS3 Not Specified Unknown (X'xxxx')
20	(14)	1	Column separator.
21	(15)	19	MO:DCA function set. Values: Graphic Arts Not Specified Unknown (X'xxxx')

META-LIST record identifies a valid metadata object that is found inline in the print file - There is one record for each valid metadata object.

OFFSET DECIMAL	OFFSET HEX	LENGTH	DESCRIPTION
0	(0)	10	Layout ID. Value: META-LIST
10	(A)	2	Reserved.
12	(C)	1	Column separator.
13	(D)	125	Metadata name or the value "None".

**Note:** In the Interchange set for print file, xxxx is the ISid value from the X'18'triplet. In the MO:DCA function set, xxxx is the FctSetID value from the X'8F'triplet.

Figure 72. Softcopy record details (Part 18 of 18)

## Sample softcopy report

Because each record of the softcopy AFPSTATS report can be 512 characters wide, you might need to scroll to see all of the information. The example in [Figure 73 on page 215](#)–[Figure 74 on page 216](#) is truncated to fit on the page. See [“Softcopy record details” on page 197](#) for descriptions of each record in the report.



**Note:** This sample report is an example of an AFPSTATS report that you might see but it does not represent actual data.

```

REPORTLVL  AFPSTATS 4.7.0 0001 PSF AFP Statistics Report
TITLE       Print File Information.
PAGEFOOTER Print File Information.
HEADING     Job ID Jobname Stepname Data Source      Print Date Print Time Level
CPU ID      Type  Intm Device Attachment Port Number Identifier
PRINTFILEP  J0B03594 DCMR103 STEP1 Deferred-spool  07/17/2019 11:11:50 PSF 4.7.0 FOR z/OS
FF020F472827 1380-01 None TCP/IP 05001 HAPPYBUBBA1A
COMMENT
TITLE       Print File Extension Information.
PAGEFOOTER Print File Extension Information.
HEADING     BPF Interchange Set MO:DCA Function Set
PRTFILEX    Not Specified Not Specified
COMMENT
TITLE       Processing Detail.
PAGEFOOTER Processing Detail.
HEADING     Res Name Resource Type Lib Type Resource Size Relative Page Data Set Name Volume Disposition
Transmission Page Number
RESOURCE    P1CMR103 Page Definition User 1 1,298 1 PSFMVS.FVT.R420.RESOURCE USR085 PSF
memory      1
RESOURCE    F10FF0 Form Definition User 1 331 1 PSFMVS.FVT.R420.RESOURCE USR085 PSF
memory      1
EVENT       Event=Repositioning Page=
RESOURCECDO M1RESET Object Container PSF defa 0 Transmission= 1 Page number= 1
1 Disposition=Download Resource=M1RESET
Data set= Internal
VSER=n/a Transmission= 1 Page number= 1
CMRScope= n/a CMRPRMode= n/a
RESOURCE    X0GT12 Coded Font 1 System 79 1 PSFMVS.FVT.R450.RESOURCE USR073 PSF
memory      1
RESOURCE    T100BASE Code Page 1 System 2,153 1 /usr/lpp/PSF/fonts/codepage/ ?
Download    1
RESOURCE    C0D0GT12 Character Set 1 System 48,231 1 COMMON.FONT4028 PSF000
Download    1
RESOURCECDO I1SCHLAF Object Container User 876 1
Disposition=Integrated into p Resource=I1SCHLAF
Data set= PSFMVS.FVT.R420.RESOURCE
VSER=USR085 Transmission= 1 Page number= 1
CMRScope= n/a CMRPRMode= n/a
RESOURCE    G1CMR12 Page Segment User 3,489 2 PSFMVS.FVT.R420.RESOURCE USR085 Integrated into
page 1 2
RESOURCE    B1I0B3 Page Segment User 253 3 PSFMVS.FVT.R420.RESOURCE USR085 Integrated into
page 1 3
RESOURCECDO Train... Object Container User 788,185 4
Disposition=Integrated into p Resource=Train_with_CMYKeuro_profile
File Name=iccr_TrainCMYKeuroScaleCoatedV2.jpg
DOR10/ Transmission= Path=/usr/lpp/PSF/RESOURCE/dor/
n/a 1 Page number= 4 CMRScope= n/a CMRPRMode=
EVENT       Event=Processing complete Page= 4 Transmission= 1 Page number= 4
COMMENT
NOTE        The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.
COMMENT
TITLE       Resource Summary by Name.
PAGEFOOTER Resource Summary by Name.
HEADING     Res Name Resource Type Lib Type Total Mapped Included JCL Other Soft
Download    Activate Memory Exists
SUMM-NAME   B1I0B3 Page Segment User 0 1 0 0 0 1
0 0 0
SUMM-NAME   C0D0GT12 Character Set 0 3 0 0 0 4 0
1 0 0
SUMM-NAME   F10FF0 Form Definition User 1 1 0 1 0 0
0 1 0
SUMM-NAME   G1CMR12 Page Segment User 0 1 0 1 0 1
0 0 0
SUMM-NAMEX  I1SCHLAF Object Container User 0 1 0 0 0 1
0 0 0 Resource=I1SCHLAF
SUMM-NAMEX  M1RESET Object Container PSF default 2 0 0 2 0 0
1 0 0 Resource=M1RESET
SUMM-NAME   P1CMR103 Page Definition User 1 1 0 1 0 0
0 1 0
SUMM-NAMEX  Train... Object Container User 0 1 0 1 0 1
0 0 0 Resource=Train_with_CMYKeuro_profile
SUMM-NAME   T100BASE Code Page 0 3 0 0 0 4 0
1 0 0
SUMM-NAME   X0GT12 Coded Font 0 3 0 0 0 0
0 1 0
NOTE        The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.
COMMENT
TITLE       Resource Summary by Data Set.
PAGEFOOTER Resource Summary by Data Set.
COMMENT
SECTION     User Libraries.
SUMM-DSN    Data set=PSFMVS.FVT.R420.RESOURCE VOL=SER=USR085
SUMM-DSN-R  B1I0B3 G1CMR12 I1SCHLAF
COMMENT
SUMM-DSN    Data set=PSFMVS.FVT.R420.RESOURCE VOL=SER=USR085
SUMM-DSN-R  F10FF0 P1CMR103
COMMENT
SECTION     System Libraries.
SUMM-DSN    Data set=PSFMVS.FVT.R450.RESOURCE VOL=SER=USR073
SUMM-DSN-R  X0GT12
COMMENT
SUMM-DSN    Data set=COMMON.FONT4028 VOL=SER=PSF000
SUMM-DSN-R  C0D0GT12
COMMENT
SECTION     PSF Default.
SUMM-DSN-R  M1RESET
COMMENT
SECTION     User Path.
SUMM-PATH   Path=/usr/lpp/PSF/RESOURCE/dor/DOR10/
SUMM-PTH-R  Resource=Train_with_CMYKeuro_profile
COMMENT

```

Figure 73. AFPSTATS softcopy report (Part 1 of 2)

```

SECTION      System Path.
SUMM-PATH    Path=/usr/lpp/PSF/fonts/codepage/
SUMM-PTH-R   Resource=T1D0BASE
COMMENT
NOTE         The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.
COMMENT
TITLE        Resource Summary by Resource Type.
PAGEFOOTER   Resource Summary by Resource Type.
COMMENT
SECTION      Reference Summary.
HEADING-RF   Resource Type
SUMM-REF     Unique      JCL      Mapped      Included      Other      Total
SUMM-REF     Page Definition 1      1      0      0      0      1
SUMM-REF     Form Definition 1      0      4      0      0      4
SUMM-REF     Coded Font 1      0      0      0      4      4
SUMM-REF     Character Set 1      0      0      0      4      4
SUMM-REF     Code Page 1      0      0      0      4      4
SUMM-REF     Page Segment 2      0      0      2      0      2
SUMM-REF     Overlay 0      0      0      0      0      0
SUMM-REF     Object Container 3      2      0      2      0      4
SUMM-REF     True/Open Type 0      0      0      0      0      0
COMMENT
SUMM-REF     All resource types 10      4      4      4      8      20
COMMENT
SECTION      Location Summary.
HEADING-LC   Resource Type
SUMM-LOC     Unique      Inline      User      Security      System      PSF Default      Resident
SUMM-LOC     Page Definition 1      0      1      0      0      0      0
SUMM-LOC     Form Definition 1      0      1      0      0      0      0
SUMM-LOC     Coded Font 1      0      0      0      1      0      0
SUMM-LOC     Character Set 1      0      0      0      1      0      0
SUMM-LOC     Code Page 1      0      0      0      1      0      0
SUMM-LOC     Page Segment 2      0      2      0      0      0      0
SUMM-LOC     Overlay 0      0      0      0      0      0      0
SUMM-LOC     Object Container 3      0      2      0      0      1      0
SUMM-LOC     True/Open Type 0      0      0      0      0      0      0
COMMENT
SUMM-LOC     All resource types 10      0      6      0      3      1      0
COMMENT
SECTION      Disposition Summary.
HEADING-DP   Resource Type
SUMM-DISP     Unique      Soft      Download      Activate      Memory      Exists
SUMM-DISP     Page Definition 1      0      0      0      1      1
SUMM-DISP     Form Definition 1      0      0      0      1      1
SUMM-DISP     Coded Font 1      0      0      0      1      3
SUMM-DISP     Character Set 1      0      1      0      0      3
SUMM-DISP     Code Page 1      0      1      0      0      3
SUMM-DISP     Page Segment 2      2      0      0      0      0
SUMM-DISP     Overlay 0      0      0      0      0      0
SUMM-DISP     Object Container 3      2      1      0      0      1
SUMM-DISP     True/Open Type 0      0      0      0      0      0
COMMENT
SUMM-DISP     All resource types 10      4      3      0      3      12
COMMENT
NOTE         The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.
COMMENT
TITLE        Processing Summary.
PAGEFOOTER   Processing Summary.
COMMENT
SECTION      Summary of Pages.
HEADING-SP   Total Pages      Records      File Size      Avg Page Size      Smallest      Largest      Small Page      Large Page
SUMM-PAGE     4      35      1,269      317      262      376      3      4
COMMENT
NOTE         The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.
COMMENT
SECTION      Significant Events.
HEADING-EL   Page number Event type      Resource
EVENT-LIST    1 Repositioning
EVENT-LIST    4 Processing complete
COMMENT
SECTION      Unused Inline Resource.
HEADING-UU   Resource Type      Res Name
UIR-LIST      NONE
COMMENT
SECTION      Inline Metadata.
HEADING-PM   Name
META-LIST     NONE

```

Figure 74. AFPSTATS softcopy report (Part 2 of 2)

## Sample hardcopy report

The sample hardcopy AFPSTATS report contains the same information as in [Figure 73 on page 215](#) - [Figure 74 on page 216](#) but in printable format.

### Notes:

1. The fonts in the printed version of your AFPSTATS might be different than the fonts shown in the following figures.
2. This sample report is an example of an AFPSTATS report that you might see but it does not represent actual data.

# AFPDP Statistics Report - 4.7.0 Level 0001

## Print File Information

Print File Information		Sender	
Job ID	J0B03748	Level	AFPDP 4.7.0 for z/OS
Jobname	DCMR103	FSA	PRT619
Stepname	STEP1	System	DEV2
Data Source	Deferred-spool	CPU ID	FF020F472827
Trans Date	07/25/2019	FSA Attachment	Deferred-printing
Trans Time	11:52:30		
BPF Interchange	Not Specified		

Receiver	
Type	1380-01
Printer Title	RICOH Pro VC60000
Intm Device	None
Attachment	TCP/IP
Port Number	05001
Identifier	HAPPYBUBBA1A

Print File Information

Page - 1

Figure 75. AFPSTATS report (Page 1 of 8)

## Print File Extension Information

BPF Interchange Set	M0:DCA Function Sets
Not Specified	Not Specified

Print File Extension Information

Page - 2

Figure 76. AFPSTATS report (Page 2 of 8)

## Processing Detail

<b>P1CMR103</b>			
Resource type=Page Definition	Library type=User	Relative page= 1	Disposition=PSF memory
Size= 1,298	Volume=USR085	DSN=PSFMVS.FVT.R240.RESOURCE	
Transmission Count= 1	Page number= 1		
<b>F10FF0</b>			
Resource type=Form Definition	Library type=User	Relative page= 1	Disposition=PSF memory
Size= 331	Volume=USR085	DSN=PSFMVS.FVT.R240.RESOURCE	
Transmission Count= 1	Page number= 1		
<b>Event=Repositioning</b>			
Transmission Count= 1	Page = 1		
<b>M1RESET</b>			
Resource type=Object Container	Library type=PSF data	Relative page= 1	Disposition=Download
Size= 0	CMR Scope=n/a	Processing mode=n/a	
Path=n/a		File=Internal	
Transmission Count= 1	Page number= 1		
<b>X0GT12</b>			
Resource type=Coded Font	Library type=System	Relative page= 1	Disposition=Download
Size= 79	Volume=?	DSN=PSFMVS.FVT.R450.RESOURCE	
Transmission Count= 1	Page number= 1		
<b>T1D0BASE</b>			
Resource type=Code Page	Library type=System	Relative page= 1	Disposition=Download
Size= 2,153	Volume=?	DSN=/usr/lpp/PSF/fonts/codepage/	
Transmission Count= 1	Page number= 1		
<b>C0D0GT12</b>			
Resource type=Character Set	Library type=System	Relative page= 1	Disposition=Download
Size= 48,231	Volume=PSF000	DSN=COMMON.FONT.4028	
Transmission Count= 1	Page number= 1		

Processing Detail

Page - 3

Figure 77. AFPSTATS report (Page 3 of 8)

**I1SCHLAF**  
Resource type=Object Container Library type=User Relative page= 1 Disposition=Integrated into page  
Size= 876 CMR Scope=n/a Processing mode=n/a  
Path=USR085 File=PSFMVS.FVT.R420.RESOURCE  
Transmission Count= 1 Page number= 1

**G1CMR12**  
Resource type= Page Segment Library type=User Relative page= 2 Disposition=Integrated into page  
Size= 3,489 Volume=USR085 DSN=PSFMVS.FVT.R420.RESOURCE  
Transmission Count= 1 Page number= 1

**B1IOB3**  
Resource type= Page Segment Library type=User Relative page= 3 Disposition=Integrated into page  
Size= 253 Volume=USR085 DSN=PSFMVS.FVT.R420.RESOURCE  
Transmission Count= 1 Page number= 3

**Train\_with\_CMYKeuro\_profile**  
Resource type=Object Container Library type=User Relative page= 4 Disposition=Integrated into page  
Size=788,185 CMR Scope=n/a Processing mode=n/a  
Path=/usr/lpp/PSF/RESOURCE/dor/DOR10 File=PSFMVS.FVT.R420.RESOURCE  
Transmission Count= 1 Page number= 4

**Event=Processing complete**  
Page = 1  
Transmission Count= 1 Page number= 4

**Note:** The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.

Processing Detail Page - 4

Figure 78. AFPSTATS report (Page 4 of 8)

Resource Summary by Name												
Res Name	Resource Type	Lib Type	Reference						Disposition			
			Total	Mapped	Included	JCL	Other		Soft	Inlined	Ignored	Memory Exists
B1IOB3	Page Segment	User	1	0		1	0	0	1	0	0	0
C0D0GT12	Character Set	System	4	0	0	0	4		0	1	0	3
F10FF0	Form Definition	User	1	0	0	1	0		0	0	0	1
G1CMR12	Page Segment	User	1	0	1	0	0		1	0	0	0
I1SCHLAF	Object Container	User	1	0	1	0	0		1	0	0	0
M1RESET	Object Container	PSF default	2	0	0	2	0		0	1	0	1
P1CMR103	Page Definition	User	1	0	0	1	0		0	0	0	1
Train_with_CMYKeuro_profile	Object Container	User	1	0	1	0	0		1	0	0	0
T1D0BASE	Code Page	System	4	0	0	0	4		0	1	0	3
X0GT12	Coded Font	System	4	4	0	0	0		0	0	0	1

**Note:** The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.

Resource Summary by Name Page - 5

Figure 79. AFPSTATS report (Page 5 of 8)

Resource Summary by Data Set			
User Libraries	Data_set=PSFMVS.FVT.R420.RESOURCE		VOL=SER=USR085
	B1I0B3	G1CMR12	I1SCHLAF
	Data_set=PSFMVS.FVT.R420.RESOURCE		VOL=SER=USR085
	F10FF0	P1CMR103	
System Libraries	Data_set=PSFMVS.FVT.R450.RESOURCE		VOL=SER=USR073
	X0GT12	T1D0BASE	
	Data_set=COMMON.FONT4028		VOL=SER=USR073
	C0D0GT12		
PSF Default	M1RESET		
User Path	Path=/usr/lpp/PSF/RESOURCE/dor/D0R10/		
	Train_with_CMYKeuro_profile		
System Path	Path=/usr/lpp/PSF/fonts/codepage		
	T1D0BASE		
<b>Note:</b> The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.			
Resource Summary by Data Set			Page - 6

Figure 80. AFPSTATS report (Page 6 of 8)

Resource Summary by Resource Type							
<b>Reference Summary</b>							
Resource Type	Unique	JCL	Mapped	Included	Other	Total	
Page Definition	1	1	0	0	0	1	
Form Definition	1	1	0	0	0	1	
Coded Font	1	0	4	0	0	4	
Character Set	1	0	0	0	4	4	
Code Page	1	0	0	0	4	4	
Page Segment	2	0	0	2	0	2	
Overlay	0	0	0	0	0	0	
Object Container	3	2	0	2	0	4	
True/Open Type	0	0	0	0	0	0	
All resource types	10	4	4	4	8	20	
<b>Location Summary</b>							
Resource Type	Unique	Inline	User	Security	System	PSF Default	Resident
Page Definition	1	0	1	0	0	0	0
Form Definition	1	0	1	0	0	0	0
Coded Font	1	0	0	0	1	0	0
Character Set	1	0	0	0	1	0	0
Code Page	1	0	0	0	1	0	0
Page Segment	2	0	2	0	0	0	0
Overlay	0	0	0	0	0	0	0
Object Container	3	0	2	0	0	1	0
True/Open Type	0	0	0	0	0	0	0
All resource types	10	0	6	0	3	1	0
<b>Disposition Summary</b>							
Resource Type	Unique	Soft	Download	Activate	Memory	Exists	
Page Definition	1	0	0	0	1	1	
Form Definition	1	0	0	0	1	1	
Coded Font	1	0	0	0	1	3	
Character Set	1	0	1	0	0	3	
Code Page	1	0	1	0	0	3	
Page Segment	2	2	0	0	0	0	
Overlay	0	0	0	0	0	0	
Object Container	3	2	1	0	0	1	
True/Open Type	0	0	0	0	0	0	
All resource types	10	4	3	0	3	12	
<b>Note:</b> The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.							
Resource Summary by Resource Type							Page - 7

Figure 81. AFPSTATS report (Page 7 of 8)

Processing Summary			
Summary of Pages			
Total pages	4		
Records	35		
File size	1,269		
Average page size	317		
Smallest page size	262	Page number= 3	
Largest page size	376	Page number= 4	
Transformed bytes			
Note: The statistics in this report may contain inaccuracies caused by error recovery and operator actions making it unsuitable for accounting purposes.			
Significant Events			
	Page number	Event type	Resource
	1	Repositioning	
	4	Processing complete	
Unused Inline Resources			
	Resource Type	Resource Name	
	NONE		
Inline Metadata			
	Name		
	NONE		
Processing Summary			Page - 8

Figure 82. AFPSTATS report (Page 8 of 8)

## Appendix D. Page-printer defaults form

You can copy the page-printer defaults form in [Figure 83 on page 221](#) and complete it with the defaults assigned to the printers in your installation. Obtain the default information from your system-support group, and complete a copy for each different CLASS, DEST, FLASH, and FORM value for each printer or group of printers with a unique set of defaults.

### PAGE-PRINTER DEFAULTS FORM

PAGE-PRINTER Type is: \_\_\_\_\_

Printer selection parameters:CLASS= \_\_\_\_\_

DEST= \_\_\_\_\_

FLASH= \_\_\_\_\_

1. Media or form usually loaded in printer:  
For continuous-forms printer, size of form usually loaded: \_\_\_\_\_  
For continuous-forms printer with a burster-trimmer-stacker (BTS), output usually burst? YES | NO  
For cut-sheet printer, type of media usually in each source:  
Bin 1 \_\_\_\_\_ Bin 2 \_\_\_\_\_  
Bin \_\_\_\_\_ Bin \_\_\_\_\_
2. DATAK = BLOCK | UNBLOCK | BLKCHAR | BLKPOS
3. CKPTPAGE = \_\_\_\_\_ or CKPTSEC = \_\_\_\_\_
4. PIMSG = NO | YES Message count = \_\_\_\_\_
5. FORMDEF= \_\_\_\_\_  
Copy Group= \_\_\_\_\_  
Page Position= \_\_\_\_\_  
Paper source for cut-sheet printers: BIN1 | BIN2 | BIN \_\_\_\_\_  
Duplex: NO | NORMAL | TUMBLE | RNORMAL | RTUMBLE  
For 3800 printer:Flash activated? YES | NO
6. PAGEDEF = \_\_\_\_\_  
Form size = \_\_\_\_\_  
No. of print lines = \_\_\_\_\_  
Printable area = \_\_\_\_\_  
Lines per inch = \_\_\_\_\_  
Print direction:ACROSS | DOWN | BACK | UP
7. Default fonts are (listed in TRC sequence):  
0 = \_\_\_\_\_ 1 = \_\_\_\_\_ 2 = \_\_\_\_\_ 3 = \_\_\_\_\_  
If PAGEDEF is not specified in JCL:  
Fonts CAN | CANNOT be changed using CHARS \*
8. Fonts supported by printer: Resident | Host | Both  
Double-byte fonts supported? YES |NO  
If yes, default PRMODE: SOSI1 |SOSI2 |SOSI3 |SOSI4  
PSF Exit 7 has raster fonts automatically mapped to outline fonts for this printer: YES | NO
9. Printer attached to DPF or PSF direct? YES | NO
10. Other characteristics of this printer: \_\_\_\_\_
11. COMSETUP = \_\_\_\_\_ (microfilm device only)
12. Where are the messages sent: Printed with output | Printed on printer | On output class
13. COLORMAP = \_\_\_\_\_

\* Fonts can be changed unless a JES default forms control buffer (FCB) (page definition) defined for this printer specifies fonts.

Figure 83. Page-printer defaults form





---

## Appendix E. Microfilm device considerations

With PSF, you can use the same data set for printing jobs on paper output or on microfilm. You specify the COMSETUP parameter in your JCL to transfer microfilm setup resource data to the microfilm device for processing (see [“Specifying JCL parameters for microfilm jobs”](#) on page 148); otherwise, the host default setup resource that your system programmer created is used, and your output might not appear as you expected. For information about creating a microfilm setup resource, see [PSF for z/OS: Customization](#).

This information presents considerations for when you are creating a microfilm job both for printing to paper and for sending to microfilm devices.

---

### Images on a page

Image data that represents characters cannot be accessed as extracted text on the microfilm device. Data that is used in text extraction processing must be defined by code point.

Carefully consider the shading patterns used in the job. Jobs that are printed on paper can use much finer gradations of shading than jobs printed on microfilm. For microfilm jobs, IBM recommends that jobs use only three shading patterns: none, 15% for 50% shading, and 30% to 100% for 100% shading.

Complicated images that use different gray levels might not print on microfilm with the same quality as on paper. This quality consideration applies to logos created as bitmaps (raster images) or with graphic drawing orders.

---

### Graphics on a page

Graphic data that represents characters cannot be accessed as extracted text on the microfilm device.

Complicated graphics that use different gray levels might not print on microfilm with the same quality as on paper. This quality consideration applies to logos created as bitmaps (raster images) or with graphic drawing orders.

---

### Form definitions

If you are printing on cut-sheet printers, use a form definition with N\_UP 1 or PORTRAIT ACROSS. Otherwise, your output can rotate incorrectly.

If you are printing on a 3800 printer, your application must be compatible with the 3900 continuous-forms printers. Otherwise, your output can rotate incorrectly. See [“Page-presentation compatibility”](#) on page 59.

Consider using different form definitions and page definitions for AFP print jobs and for microfilm jobs. For example, print jobs might need to contain constant forms, but microfilm jobs might not. Also, microfilm devices print in simplex mode.

A form definition, a page definition, and the microfilm setup resource data interact together to present an AFP page on microfilm. Coordination between resources is necessary, and in some cases a test run is recommended. Work with your system programmer to set up a test run.

If you use N\_UP to place multiple pages in partitions on a sheet, your output might be unreadable on a microfilm device.

---

### Using FOCA fonts

Carefully consider the point size of the fonts used in your job. Fonts smaller than six points might produce unreadable data.

Work with your system programmer to set up different job classes for 120-pel or 240-pel resolution jobs and for 300-pel resolution jobs.

Using 120-pel resolution gives a higher printing rate than a higher resolution, but it might not produce the quality of output you need. Run some valid samples before you archive the data at 120-pel resolution.

Fonts for printing on microfilm devices must contain global resource identifier (GRID) data. If you are using fonts in user libraries or inline in your data stream, consult your system programmer to make sure that the fonts include GRID data. To see whether your printer supports fonts with GRID data, see the documentation provided with the printer.

## Page segments

---

Text data in a page segment that is not in AFP text (PTOCA) format cannot be used for text extraction on a microfilm device.

## Overlays

---

Text data (image or graphic) in an overlay that is not in AFP text (PTOCA) format cannot be used for text extraction on a microfilm device.

## Specifying parameters

---

Specify `COMSETUP=your comsetup` in your JCL. This COMSETUP parameter must reside in a system library, in a specified user library, or inline in your data stream. For more information about the COMSETUP parameter, see [“Microfilm setup resource” on page 29](#).

To prevent data checks from being reported, specify `DATACK=BLOCK` in your JCL.

Avoid using checkpointing (the CKPTPAGE and CKPTSEC JCL parameters) for microfilm jobs.

Specify `COPIES=1`; specifying multiple copies produces extra copies on the microfilm.

## Additional parameters to help in distributing output

---

This information can be passed to a microfilm device for extraction and is accessible through commands in your microfilm setup resource:

- `NAME=name`
- `ROOM=name`

See [“Additional parameters to help in distributing output” on page 108](#).

## Printing constant forms

---

Specifying `CONSTANT BACK` and `COPIES` in your form definition causes extra pages on microfilm. You might not want constant data on microfilm.

## Other considerations

---

Your system programmer might cause the error messages that PSF prints at the end of your print data sets to be redirected to another destination. To ensure that the data set printed as you expect, contact your system programmer to determine the destination.

---

## Appendix F. Color and grayscale printing

Printing documents in full color or with high-quality black and white (grayscale) images is more complex than printing black and white or spot-color documents. Understanding some of the principles of color and grayscale printing and how various products can fit into color and grayscale solutions can help you integrate color and grayscale printing with your current operations or expand to implement new color workflows.

---

### AFP color and grayscale solutions

You can assemble printing products in different configurations to support Advanced Function Presentation (AFP) color and grayscale printing, including configurations that use the AFP Color Management Object Content Architecture (CMOCA) to provide optimal performance and color accuracy in high-speed color printing.

#### Color printing without explicit color management

You can include color images or specify colors for AFP objects in your print jobs and send them to an AFP color printer. The color images and objects print in color, based on the default settings in your print server and printer.

If you like the colors that the default color management settings provide, or if it is not essential that you print in exactly the right colors, you probably do not need to implement a full color management solution. However, if you want better control over the consistency and accuracy of your colors for devices, you might consider color management at some point in the future.

To understand some of the basic concepts about color printing, see:

- [“Color printing concepts” on page 226](#)
- [“Grayscale printing concepts” on page 229](#)

#### Resources included inline

Most often, print bureaus use document composition software to generate highly customized and personalized color output. AFP color management is largely built into the document composition tools that support their processes. The software puts all the resources that the printer needs into the print job and sends it to a print server. The print server sends the print job to the printer, and the printer appropriately uses the resources.

By using this method, you know that the required resources, including the resources that are required for color management, are available for any print job that is sent to the printer. However, including all the resources can make the print job extremely large, and moving large print jobs through your system might slow down system performance. Also, you might not be able to save resources that are downloaded with a print job on the printer so they can be reused without being downloaded later.

For more information about color printing, see:

- General information about color printing and color management:
  - [“Color printing concepts” on page 226](#)
  - [“Color management” on page 230](#)
- [“Tips for images” on page 240](#)

For a list of the companies that participate in the AFP Consortium and support AFP color management in their products, see [AFP Consortium \(www.afpcinc.org\)](http://www.afpcinc.org).

## Resources stored and managed centrally

To take full advantage of the AFP CMOCA, you can store your color and image resources in a central resource library, and your print system can manage those resources. This option optimizes system performance by:

- Creating some of the color management resources for you automatically
- Reducing the number of color conversion resources that the system creates at print time by generating link color conversion color management resources in advance
- Reducing the size of some images by removing embedded profiles when you store them, yet still retaining the association between the image and the profile
- Allowing you to mark resources as *capturable*, so they can be saved on the printer and used in other print jobs without being downloaded again

For an introduction to the full AFP CMOCA and how you might implement it, see:

- General information about color and grayscale printing and color management:
  - [“Color printing concepts” on page 226](#)
  - [“Grayscale printing concepts” on page 229](#)
  - [“Color management” on page 230](#)
- [“AFP color management” on page 231](#)
- [“AFP resource installer” on page 241](#)

## Color printing concepts

---

Color printing is significantly more complicated than black and white printing. If you understand some of the complexities, you can make the transition from black and white printing to grayscale or color printing more smoothly.

### Color spaces and ICC profiles

Presentation devices, such as computer monitors and printers, create colors differently. Because of these differences, colors must be described differently for each device. The different methods of describing colors are called *color spaces*. In addition, each device might have one or more International Color Consortium (ICC) profiles associated with it. ICC profiles are used when an image or another object is converted to the color space of a different device.

Each device has its own individual color space and range of colors that it can display or print. The color space specifies how color information is represented in an image when it is displayed on a particular device. As the image is passed from one device to the next, the color information about the image is converted from the color space of the source device to the color space of the destination device. Because color spaces do not exactly match between devices, some of the color information can be lost or modified in the conversion process.

A *color space* is a representation of the individual colors that can be combined to create other colors. Some color spaces that are relevant to printing are:

#### RGB

In an RGB color space, red, green, and blue light are combined in different amounts and intensities to create different colors. RGB colors are often specified as single-byte integers numbered from 0 through 255. You can specify 256 levels of intensity for each of the three colors. For example:

- R=0, G=0, B=0 yields black
- R=255, G=255, B=255 yields white
- R=251, G=254, B=141 yields a pale yellow
- R=210, G=154, B=241 yields a light purple

Devices such as monitors, digital cameras, and scanners generally use RGB color spaces to describe colors. Two standard implementations of RGB color spaces are sRGB, which is most often used for web graphics, and Adobe RGB (1998), which is recommended for graphics that are printed.

## CMYK

In a CMYK color space, cyan (bright blue), magenta (bright red-pink), yellow, and black pigments are combined to create different colors. CMYK values are often represented as a percentage. The percentage represents the portion of a particular area of paper that is covered by ink or toner. For example:

- C=0%, M=0%, Y=0%, K=100% yields black
- C=0%, M=0%, Y=0%, K=0% yields a blank area on the page
- C=1.6%, M=0%, Y=44.7%, K=.4% yields a pale yellow
- C=17.6%, M=39.6%, Y=5.5%, K=5.5% yields a light purple

Color printers use the CMYK color space; they are loaded with ink or toner in each color. When the printer places dots of the correct sizes next to and on top of each other on a page, your eye interprets them as the intended color.

Implementations of the CMYK color space vary from printer to printer and from paper to paper. Because the original color space of most images is an RGB color space, it is best to leave images in an RGB color space so they retain their original characteristics. That way, your print server or printer has as much of the original color information as possible when it converts the images to the most appropriate CMYK color space for the printer and paper combination.

If you save an image by using the CMYK color space, make sure that you either save an ICC profile for that color space or use a standard non-device specific CMYK color space like SWOP or Coated FOGRA27 and associate the appropriate ICC profile with the image.

**Note:** Both RGB and CMYK values can be expressed in different ways. For example, in the PostScript data stream, the values range from 0.0 to 1.0, while in some graphic arts programs they can be expressed in hexadecimal numbers or as percentages.

An *ICC profile* contains information for converting an image between a device-specific color space and a *device-independent color space*. A device-independent color space is a color space that does not depend on or relate to the characteristics of any particular device, but rather contains all colors for all gamuts. The ICC identified a specific profile connection space (PCS) as the target device-independent color space for all ICC profiles.

You can use an input ICC profile to translate color data created on one device (such as a digital camera) into the PCS. Then you can use an output ICC profile to convert from the PCS into the native color space of a different device (such as a printer). Converting images from one color space to another is process-intensive and can affect performance in your print system, although it is the best way to maintain consistent color for the devices in your system.

For more information, see [“ICC profiles” on page 230](#).

## Gamut and rendering intent

Every device has a *gamut*, a range of colors or shades of colors that it can display or print. Some devices have larger gamuts than others; some devices have gamuts that are similar sizes, but that contain slightly different colors. When an image or a print job is created on a device with a gamut that is different from the printer, you can use a *rendering intent* to tell the printer how to adjust the colors that are outside the gamut of the printer.

The gamut of a printer is almost always significantly smaller than the gamut of a monitor, digital camera, or scanner. Images or graphics typically must be adjusted to print appropriately because some of the colors that they require might be outside the gamut of the printer.

A rendering intent tells the printer how to adjust the image when it encounters colors that it cannot reproduce. Each rendering intent has different benefits and trade-offs, so you can choose one based on how the print output should look.

For more information, see [“Rendering intents”](#) on page 230.

## Color mixing and calibration

Four standard colors (cyan, magenta, yellow, and black) are blended to create all the colors in the gamut of a printer. A printer mixes colors by printing four layers of a page or an image, one in each color. If the printer registration is not set correctly, the images do not line up properly and the colors appear wrong. In addition, the printer must be calibrated to ensure that all its systems are functioning correctly and that it is in a known good state.

By using the color information that is described in its color space, each device determines the amount of cyan, magenta, yellow, or black to use. Dots of each color are printed in overlapping patterns that, when interpreted by your eyes, blend the colors appropriately. To ensure that the colors are created accurately, the color planes must be perfectly aligned. If they are not, you might see *moire patterns*, unintended patterns in the printed images, or poorly blended colors, which are especially noticeable on the edges of your images.

Color printers must be calibrated regularly, in some cases daily, to ensure that the colors they produce are consistent. In addition, follow the recommended printhead maintenance procedures and schedule to ensure that the printer operates optimally. Even when a printer is calibrated correctly, its gamut is much smaller than that of any monitor, so images do not look the same when they are printed as they do when they are displayed on a monitor.

## Halftones and tone transfer curves

Halftones are used to convert images (such as photographs, drawings, logos, or charts) from the continuous tones that you see on a monitor into a pattern of dots that a printer can put on paper. Tone transfer curves are used to modify the values of a particular color component and thus adjust the appearance of some of the colors. For example, you can apply a tone transfer curve to emphasize the brightest parts of an image.

Halftones and tone transfer curves are used with both color and grayscale print jobs.

Several different kinds of halftones exist, including clustered-dot, stochastic, and error diffusion. For simplicity, this information describes only clustered-dot halftones.

Clustered-dot halftones are characterized by:

### Line screen frequency

Line screen frequency is a measure of the resolution of a halftone, expressed in lines per inch (lpi). A low line screen frequency, such as 80 lpi, creates coarser images because they use larger halftone dots. A high line screen frequency, such as 150 lpi, can produce higher-quality images by using smaller halftone dots.

### Halftone pattern

Halftone dots are printed in various shapes and patterns. For example, dots can generally be round, elliptical, or square, and they can be arranged in slightly different orientations. The halftone pattern also describes how the size of the dot is increased to cover a larger percentage of the total area and yield darker colors. Different patterns might produce better results for some print jobs.

### Rotation

Lines of halftone dots do not run parallel with the top or side of the paper because that might cause unintended patterns to emerge, resulting in lower quality output.

In addition, the dots for each of the four colors in a CMYK printer cannot all be printed at the same angle because they would overlap incorrectly and the colors would not appear as intended. Instead, the lines of dots are printed on the page at specific angles so your eye blends them appropriately.

For example, the black layer of an image might be printed so that the lines of dots run across the page at a 45 degree angle to the top of the paper, while the cyan layer is printed so that its lines of dots are at a 105 degree angle to the top of the paper.

Tone transfer curves are most often used to offset the effects of dot gain. *Dot gain* is the tendency for printed dots to be larger than intended, often because of the way ink reacts with paper. If the ink soaks

into the paper and spreads out, the resulting dot is much larger (and possibly much lighter in color) than the printer intended it to be. Tone transfer curves can increase or reduce the amount of ink used in proportion to the dot gain.

## File size

Color print jobs can have a file size that is much larger than black and white print jobs. The larger file size can lead to longer processing times and increased traffic on your network.

Because color images must contain data about each layer of color, the file might contain three to four times more information than a grayscale file and over 24 times more information than a black and white file. In addition, ICC profiles are embedded in some file types (such as TIFF images). While ICC profiles by themselves might not be very large, they do increase the size of an image. If you have only one image that is repeated throughout a print job, and if you construct your job so the image is downloaded only once, the embedded profile is of little concern.

However, if you use a variety of different images, each with an embedded profile, or if you construct your print job so that each image is downloaded every time that it appears, the embedded profiles can add unnecessary volume to the print job. If you plan to use a wide variety of color images, create or save them with the same color space so they all use the same ICC profile. You can also install color images in a resource library so they can be reused.

## Grayscale printing concepts

---

With grayscale printing, you can reproduce color images as high-quality black and white images by using many shades of gray to represent subtle variations in color and light. Printing solutions that produce high-quality grayscale output use color printing concepts with a black and white printer that supports them to achieve that effect.

Moving to grayscale printing might be a first step in a migration to full color printing. You can start to create color print jobs and print them on an existing printer until you are ready to invest in color printers. In addition, you can use a grayscale printer as a backup system for a full color printer.

Some color concepts are much less important in grayscale printing than they are in color printing:

- The gamut of a black and white printer is much smaller than the gamut of a color printer; essentially all the colors in an image must be adjusted.
- The rendering intent that you choose has little effect on the appearance of the image because the colors are already being changed significantly.
- Page registration is less important. Because the printer uses only one color, you do not have to line up the color planes to create the correct color.
- Paper characteristics have minimal effect on grayscale output; one output profile is typically adequate for all types of paper.

Other color concepts are more essential to grayscale printing.

## Color spaces and ICC profiles

The color space of a black and white printer is much smaller than the color space of a color printer. Even so, printers that can print grayscale images have output ICC profiles, just like color printers. The ICC profiles for black and white printers map colors from the profile connection space (PCS) to shades of gray. Otherwise, the color conversion process is the same.

The print job needs to specify the appropriate input profile; if no input ICC profile is specified, the printer uses a reasonable default. The printer has its own default ICC profile that is installed and available; it is probably adequate for most print jobs.



## Halftones

Grayscale printers apply halftones to print jobs to print them; the printer uses halftones to produce many shades of gray and high-quality images. Generally, the most important characteristic to consider for halftones in grayscale printing is line screen frequency, expressed in lines per inch (lpi). Each printer supports a set of line screen frequencies natively; when you specify the line screen frequency you want in a print job, the printer chooses the available line screen frequency that best matches it.

## Tone transfer curve

Tone transfer curves are used in grayscale printing to adjust the amount of toner that is used at different levels of gray, thus adjusting the appearance of images. You can use the appearance value of a tone transfer curve in grayscale printing to indicate how much the tone transfer curve needs to adjust the color values. Some sample appearance values can be:

- Dark
- Highlight Midtone
- Standard

## Color management

---

Images, graphics, and photographs often appear different depending on the monitor or the printer you use. The colors that are printed by one printer might not match the colors that are printed on another printer, even if they came from the same source. If it is important that colors stay consistent from camera, scanner, or monitor to printer, you must use *color management* practices.

It is almost impossible to accurately reproduce the colors that you see on your monitor on a printed page. Because printers typically have smaller color gamuts than other devices, some of the colors must always be adjusted when images are transformed for printing. With color management, you can control the adjustments so they are less noticeable than they might be if you use the default settings of your image creation software, print server, and printer.

Several factors play significant roles in color management, including ICC profiles, rendering intents, and paper characteristics.

## ICC profiles

The International Color Consortium (ICC) is an organization that established open standards for color management. These standards help products work together by identifying a device-independent color space and defining the elements of an ICC profile.

The device-independent color space that the ICC defined is called the profile connection space (PCS). The PCS is a color space large enough to include all the color gamuts of different input, display, and output devices. An ICC profile contains methods that map the colors that a device can create or display to the values of the corresponding colors in the PCS. The ICC profile can be used to convert an image from a device-specific color space to the PCS, or from the PCS to a device-specific color space.

Product manufacturers create ICC profiles that you can use with their devices. For example, if you take a photograph with a digital camera, you can associate the photograph with the ICC profile for your camera. Then, when you want to print that photograph, the color management system converts the color data from the camera into the PCS. The printer then uses its ICC profile to convert the photograph data from the PCS into its color space, and prints the photograph as accurately as it can.

For more information about the ICC, ICC profiles, and the PCS, see [ICC website](#).

## Rendering intents

Rendering intents indicate what you want a printer to do with colors that are outside its gamut.

ICC profiles support these rendering intents:



**Perceptual**

If an image includes any colors that are out-of-gamut for the printer, the printer adjusts all the colors in the image, even those colors that are already in the gamut of the printer, so they are all in-gamut and maintain their color relationships to each other. The result is an image that is visually pleasing, but is not colorimetrically accurate. The perceptual rendering intent is useful for general reproduction of images, particularly photographs.

**Saturation**

If a print job includes colors that are out-of-gamut for the printer, the printer replaces the out-of-gamut color with the nearest color in the gamut. It also adjusts the in-gamut colors so that they are more vivid. Saturation is the least used rendering intent, but it is useful for business graphics, such as images that contain charts or diagrams.

**Media-relative colorimetric**

If a print job includes colors that are out-of-gamut for the printer, the printer substitutes the nearest in-gamut color; in-gamut colors are not adjusted. Colors that are printed on papers with different media white points might not match visually. The *media white point* is the color of the paper that the print job is printed on. For example, if you print an image on white paper, on off-white paper, and on blue paper by using the media-relative colorimetric rendering intent, the printer uses the same amount of ink or toner for each one and the resulting color is technically the same. However, the images might seem different because your eyes adjust to the color of the background and interpret the color differently. This rendering intent is typically used for vector graphics.

**Absolute colorimetric**

All colors are mapped by using the same method as the media-relative colorimetric rendering intent; however, all colors are adjusted for the media white point. For example, if you print an image on white paper, on off-white paper, and on blue paper by using the absolute colorimetric rendering intent, the printer adjusts the ink or toner used for each one. The resulting color is technically not same, but the images might look the same because of the way your eyes interpret them in relationship to the color of the paper. The absolute colorimetric rendering intent is typically used for logos.

## Paper characteristics

The paper that you use has a significant impact on the colors that you see. Even if you use the same ICC profile and the same printer, printing on a different paper can result in a different color appearance.

Colors can change from paper to paper, particularly if you change from coated to uncoated paper or from sheet-fed to continuous forms paper. The changes can be so noticeable that printer manufacturers generally test and certify papers with certain characteristics for use with their printers. They also create different ICC profiles for their printers based on paper characteristics. Some ICC profiles can be used for groups of papers that have similar characteristics.

When you load paper, you set certain paper characteristics on the printer. When the printer chooses the correct device-specific output profile to use, it takes the characteristics into consideration. The paper characteristics are:

**Media brightness**

The percentage of light that the paper reflects.

**Media color**

The color of the paper.

**Media finish**

The characteristics of the surface of the paper, such as: glossy, satin, matte.

**Media weight**

The basic weight of the paper.

## AFP color management

---

You can use various ways to print color data with Advanced Function Presentation (AFP). However, to implement an AFP color printing solution with full color management, you must use color management

resources (CMRs). You need to also install all of your color images as data objects and associate CMRs with them.

## Color management resources

*Color management resources* (CMRs) are the foundation of color management in AFP print systems. They are AFP resources that provide all the color management information, such as ICC profiles and halftones, that an AFP system needs to process a print job and maintain consistent color from one device to another.

CMRs share some characteristics with other AFP resources, but are different in some important ways.

CMRs are similar to other AFP resources in these ways:

- CMRs can be associated with elements of a print job at various levels of the hierarchy. Typical hierarchy rules apply, so CMRs specified at lower levels override those elements at the higher level. For example, a CMR set on a data object overrides a default CMR set on a print file.
- CMRs can be included in a print job in an inline resource group and referenced in a form definition, page environment, object environment, or an include Object (IOB) structured field.

**Note:** CMRs can vary in size from several hundred bytes to several megabytes. If your print job uses relatively few CMRs, including them in the print file might not have an impact on the performance of your system. However, if your print job uses more than 10 CMRs, the size of the print job can increase so much that file transfer rates and network traffic are affected.

- CMRs can be stored centrally in a resource library, so you do not need to include them in every print job. You can configure all your print servers so they can access the CMRs.
- For the print server to find CMRs, the resource library must be listed in the AFP resource search path on the print server.

CMRs are different from other AFP resources in these ways:

- You cannot copy CMRs into a resource library as you can other AFP resources. To store CMRs in a central resource library, you must install them by using an application such as an AFP resource installer.
- CMRs and data objects must be stored in resource libraries that have resource access tables (RATs). An AFP resource installer creates the RAT when CMRs and data objects are installed. You need to install CMRs and data objects in separate resource libraries and store resources that do not require RATs (such as form definitions, page definitions, and overlays) in other resource libraries.

For more information, see [“AFP resource installer” on page 241](#).

- CMRs installed in a resource library can have names longer than 8 characters, and you can use the names in the print data stream.

These names are created when you install the CMR by using an AFP resource installer and are UTF-16BE encoded.

See [“Resource library management” on page 239](#) for more information about characteristics of resource libraries.

## Types of CMRs

Different situations call for different types of CMRs. Some CMRs are created by product manufacturers so you can download and use them, while others are created by your printer or other color management software. If you have the appropriate information, you can also create CMRs yourself.

Some CMRs are used to interpret input files (similar to the function performed by ICC input profiles), while others are used to prepare the final print job output for a specific printer (similar to the function performed by ICC output profiles).

## **Color conversion CMRs**

Color conversion (CC) CMRs are used to convert colors to and from the ICC profile connection space (PCS), a device-independent color space. You can use them to prepare images for color or grayscale printing.

Color conversion CMRs are an essential element of any AFP color management system because they are ICC profiles encapsulated in AFP structures. The AFP structures add information that your color management system can use, but it leaves the ICC profile unaltered.

You can use color conversion CMRs to produce consistent colors on different devices. In a color system, they help ensure that the colors on your monitor are as close as possible to those colors that are printed. If you move the print job to a different printer, the colors are adjusted again to match the new printer.

In a grayscale system, color conversion CMRs map colors to appropriate shades of gray to produce high-quality black and white images.

Passthrough CMRs are color conversion CMRs that indicate that no color processing is done if the color space of the presentation device is the same as the color space of the CMR. Passthrough CMRs contain no data.

## **Link color conversion CMRs**

Link color conversion CMRs combine the processing information that is required to directly convert an image from the color space of an input device to the color space of the output device. Essentially, link color conversion CMRs replace a pair of color conversion CMRs.

Converting color images to and from the PCS takes a significant amount of processing resources, in part because the process includes two conversions. Link color conversion CMRs combine the two conversions and make them more efficient. The printer can use the link color conversion CMR to convert colors directly from the color space of the input device to the color space of the output device with the same color fidelity they would have if the printer did both of the conversions. As a result, link color conversion CMRs can improve system performance.

The two types of link color conversion CMRs are:

### **Link CMRs**

Link (LK) CMRs are unique. You cannot create a link CMR yourself and you do not include references to link CMRs in your print jobs. The print system creates and uses link CMRs automatically.

If you use an AFP resource installer, link CMRs are generated automatically when you create or install a color conversion CMR. As a result, your resource library always contains link CMRs for every combination of color conversion CMRs in audit (input) and instruction (output) processing modes. When link CMRs are created, the AFP resource installer marks them as *capturable*, so the printer can save them to be used in other print jobs.

If you do not use an AFP resource installer, your printer might create link CMRs when it processes print jobs. For example, if you send a print job to an InfoPrint 5000, the printer controller looks at the audit color conversion CMRs that are specified. Then, the print controller looks at the link CMRs that it has available to find one that combines the audit color conversion CMR with the appropriate instruction color conversion CMR. If it does not find one, the print controller creates the link CMR and uses it. The print controller might save the link CMRs that it creates, but they can be removed during normal operation; for example, if the printer runs out of storage or is shut down. If the link is removed, the printer must create a new link CMR the next time it is needed.

When a link CMR is created, the print system evaluates the conversion algorithms to and from the PCS. The system then combines the algorithms, so a data object can be converted directly from one color space to the other without being converted to the PCS.

### **Device link CMRs**

Device link (DL) CMRs use an ICC device link profile to convert directly from an input color space to an output color space without reference to an audit-mode or instruction-mode CMR. An ICC device link profile is a special type of ICC profile that is used to convert the input device color space to the color space of an output or display device. ICC device link profiles are not embedded in images.

You can create, install, and uninstall device link CMRs yourself. Device link CMRs are referenced in the MO:DCA data stream and take precedence over audit color conversion CMRs. A device link CMR specifies its own rendering intent, which is indicated in the header of the ICC device link profile. This rendering intent overrides any other rendering intent that is active.

The biggest advantage of using device link CMRs is that they preserve the black channel (K component) of the input color space when CMYK is converted to CMYK.

### **Halftone CMRs**

Halftone (HT) CMRs carry the information that a printer uses to convert print jobs into a pattern of dots that it can put on paper. Halftone CMRs can be used with both color and grayscale print jobs.

Halftone CMRs generally specify the line screen frequency, halftone pattern, and rotation of the halftone that they carry. Device-specific halftone CMRs might also include the printer resolution.

A printer that uses AFP color management to print color or grayscale print jobs must use a halftone CMR to convert the print job into a format that the printer can reproduce in ink or toner. If a halftone CMR is not specified in the print job, the printer applies a default halftone CMR.

**Note:** If you send your color print jobs to an InfoPrint 5000 printer, halftones are applied by the print engine. As a result, the printer ignores halftone CMR requests.

You can associate device-specific halftone CMRs or generic halftone CMRs with print jobs:

- If you know which printer is printing the job, you can associate a device-specific halftone CMR with the print job (or with AFP resources inside the print job). The printer uses the halftone CMR that you specify.
- If you do not know which printer is printing the job, but you want to ensure that it uses a halftone CMR that has certain characteristics, such as a specific line screen frequency, you can associate a generic halftone CMR with the print job.

Because it is difficult to know which halftone CMRs must be used for the current conditions on the current printer, you need to specify halftone CMRs generically and let the printer choose the most appropriate CMR that it has available.

### **Generic halftone CMRs**

You can use generic halftone CMRs when you want to choose one or more characteristics of the halftone CMR for a print job, but you do not know exactly which halftone CMRs are available.

When a print job specifies a generic halftone CMR, the print server looks in the resource library for halftone CMRs that match the printer device type and model. If the print server finds an appropriate CMR, it sends the device-specific halftone CMR to the printer with the print job. If the print server does not find an appropriate halftone CMR, it sends the generic halftone CMR to the printer.

If a print job that arrives at the printer is requesting a generic halftone CMR, the printer compares the requested characteristics with the available device-specific halftone CMRs. If a match exists, the printer uses the selected device-specific halftone CMR when it processes the print job. If no match exists, the printer uses the halftone CMR whose line screen frequency value is closest to the one requested.

The Color Management Object Content Architecture (CMOCA) defines various generic halftone CMRs, which cover the most common line screen frequencies and halftone types. A print server that supports CMOCA can interpret generic halftone CMRs if it has device-specific halftone CMRs available to it in a resource library. If you use an AFP resource installer, the generic halftone CMRs are installed in every resource library that you create and populate by using the AFP resource installer.

Printers that support CMOCA must be able to interpret those generic CMRs and associate them with device-specific halftone CMRs.

## ***Indexed CMRs***

Indexed (IX) CMRs map indexed colors in the data to presentation device colors or colorant combinations.

Indexed CMRs provide rules about how to render indexed colors. Indexed CMRs apply to indexed colors that are specified by using the highlight color space. They do not apply to indexed colors found within PostScript or other non-IPDS data objects. For Indexed CMRs, both instruction and audit processing modes are valid. However, only indexed CMRs with an instruction processing mode are used; those CMRs with an audit processing mode are ignored. The tags in the indexed CMR let the CMR use various color spaces in the descriptions. These color spaces can be grayscale, named colorants, RGB, CMYK, or CIELAB.

## ***Tone transfer curve CMRs***

Tone transfer curve (TTC) CMRs are used to carry tone transfer curve information for an AFP print job, so you can modify the values of a particular color component and adjust the appearance of some of the colors by increasing or decreasing the amount of ink that is used to emphasize or reduce the effects of dot gain on the final output.

Like halftone CMRs, tone transfer curve CMRs are associated with print jobs specifically or generically. If they are specified generically, the print server looks in the resource library for tone transfer curve CMRs that match the printer device type and model. If the print server finds an appropriate CMR, it sends the device-specific tone transfer curve CMR to the printer with the print job. If the print server does not find an appropriate tone transfer curve CMR, it sends the generic tone transfer curve CMR to the printer.

If a print job that arrives at the printer is requesting a generic tone transfer curve CMR, the printer compares the requested characteristics with the device-specific tone transfer curve CMRs that it has available. If a match exists, the print server or printer uses the selected device-specific tone transfer curve CMR when it processes the print job. If the printer cannot find a good match for the generic tone transfer curve CMR, it ignores the request and uses its default tone transfer curve CMR.

The Color Management Object Content Architecture (CMOCA) defines several generic tone transfer curve CMRs with different appearance values. You can use the appearance values to specify how to print your job regarding the reported dot gain of the printer.

Generic tone transfer curves can be used to select these appearance values:

### **Dark**

The output is adjusted to show a dot gain of 33% for a 50% dot.

### **Accutone**

The output is adjusted to show a dot gain of 22% for a 50% dot.

### **Highlight Midtone**

The output is adjusted to show a dot gain of 14% for a 50% dot. This appearance might be used to emphasize the brightest part of an image.

### **Standard**

The output is adjusted enough to account for the effects of dot gain, effectively counteracting the dot gain.

If you use an AFP resource installer, it installs the generic tone transfer curve CMRs on your system automatically.

## CMR processing modes

CMR processing modes tell the print system how to apply a CMR to the print data it is associated with. You specify a CMR processing mode whenever you specify a CMR, although not all modes are valid for all CMR types.

### ***Audit processing mode***

CMRs with the audit processing mode refer to processing that is already applied to a resource. In most cases, audit CMRs describe input data and are similar to ICC input profiles.

The audit processing mode is used primarily with color conversion CMRs. In audit processing mode, those CMRs indicate which ICC profile must be applied to convert the data into the profile connection space (PCS).

For example, to take a photograph with a digital camera and then include the photograph in an AFP print job, you can use an AFP resource installer to:

1. Create a color conversion CMR by using the ICC profile of your camera.
2. Install your photograph in a resource library.
3. Associate the color conversion CMR with the data object, indicating the audit processing mode.

Then, you create a print job that includes the data object. When the print job is processing, the system uses the color conversion CMR to convert the colors in the image into the PCS. The colors can then be converted into the color space of the printer that is printing it.

### ***Instruction processing mode***

CMRs with the instruction processing mode refer to processing that is done to prepare the resource for a specific printer that uses a certain paper or another device. Generally, instruction CMRs refer to output data and are similar to ICC output profiles.

The instruction processing mode is used with color conversion, tone transfer curve, and halftone CMRs. In instruction processing mode, these CMRs indicate how the system must convert a resource so it prints correctly on the target printer. The manufacturer of your printer needs to provide ICC profiles or a variety of CMRs that you can use. Those ICC profiles and CMRs might be installed in the printer controller, included with the printer on a CD, or available for download from the manufacturer's website.

If you send a color AFP print job to a printer that supports AFP Color Management, color conversion and tone transfer curve CMRs in instruction processing mode can be associated with the job. When the printer processes the print job, it applies the CMRs in this order:

1. Color conversion CMRs in audit processing mode to convert the resources into the ICC profile connection space (PCS).
2. Color conversion and tone transfer curve CMRs in instruction processing mode to convert the resources into the color space of the printer.
3. Halftone CMR in instruction processing mode to convert the job pages from their digital format into the pattern of dots that the printer can produce.

In some cases, CMRs that are typically used as instruction CMRs can be used as audit CMRs. For example, if you send a very large print job to a high-speed printer, the images in the print job are converted into the color space of that printer by using a color conversion CMR with the instruction processing mode. However, if you must reprint part of the job on a different printer, the system must convert the print job into the color space of the second printer. In that case, the color conversion CMR of the first printer is used in the audit processing mode to move the images back into the PCS. Then, the system uses a color conversion CMR of the second printer in instruction mode to convert the images into its color space.

### ***Link processing mode***

CMRs with the link processing mode are used to link an input color space in the presentation data (sometimes defined by an audit CMR) to the output color space of the presentation device (sometimes

defined by an instruction CMR). Only link (LK) and device link (DL) CMRs can be used in link processing mode.

Whenever you install or uninstall audit or instruction color conversion CMRs in your resource library by using an AFP resource installer, the AFP resource installer automatically creates or deletes link (LK) CMRs for every combination of audit and instruction color conversion CMR.

When a print job calls for a specific audit-instruction combination, the print server checks the resource library for a link (LK) CMR for that combination. If the print server finds an appropriate link CMR, it sends the CMR to the printer with the print job. Your printer can use the link (LK) CMRs whenever a print job indicates that it uses a particular combination of audit and instruction CMRs.

If you do not use an AFP resource installer to install your resources, your color printer must either create link (LK) CMRs while it processes your print jobs or convert the colors in your jobs twice, first from the original color space to the PCS and then from the PCS to the color space of the printer.

## CMR creation and installation

Device manufacturers and groups that support AFP color standards create CMRs that you can use in your color printing systems. You can also create CMRs yourself, based on your needs.

The AFP Consortium, the group that defined the AFP Color Management Object Content Architecture (CMOCA), identified a set of color conversion CMRs that are most often used in audit processing mode. The set includes color conversion CMRs for common color spaces, such as:

- Adobe RGB (1998)
- sRGB
- SMPTE-C RGB
- SWOP CMYK

The standard CMRs are included with an AFP resource installer, although they are not installed by default. You can install the standard CMRs that you plan to use. In addition, an AFP resource installer automatically installs all the generic halftone and tone transfer curve CMRs in any resource library you create.

You can download device-specific CMRs from the printer manufacturer's website.

If you need more CMRs, you can create them by using wizards provided in the AFP resource installer. See the online help for details about the wizard.

If you use an AFP resource installer to create a CMR, the software automatically installs the CMR in a resource library. You can also use an AFP resource installer to install CMRs that you get from your printer manufacturer.

## Data objects

Presentation data objects contain a single type of data (such as GIF, JPEG, PNG, and TIFF images) and can be used in your print jobs. These data objects can be placed directly in a page or overlay or can be defined as resources and included in pages or overlays. Using a data object as a resource is more efficient when that object appears more than once in a print job; resources are downloaded to the printer once and referenced as needed.

Data objects can either be included inline with a print job or installed in a resource library by using software such as an AFP resource installer. If you install your data objects in a resource library, you can associate color conversion CMRs with them. For more information, see [“AFP resource installer” on page 241](#).

See [“Resource library management” on page 239](#) for more information about characteristics of resource libraries.

## Types of data objects

Image data objects can be stored in a number of different formats, including AFPC JPEG Subset, EPS, GIF, IOCA, PDF, PNG, and TIFF. These image types are device-independent so they can be used by different systems and still be interpreted consistently.

- AFPC JPEG Subset (JPEG)

AFPC (AFP Consortium) JPEG Subset files, formerly called JPEG File Interchange Format (JFIF) files, are bitmap image files that are compressed by using Joint Photographic Experts Group (JPEG) compression. As a result, AFPC JPEG Subset files are most commonly referred to as JPEG files. JPEG files most commonly use the file extension .jpg, but can also use .jpeg, .jpe, .jfif, and .jif.

JPEG compression deletes information that it considers unnecessary from images when it converts them. JPEG files vary from having small amounts of compression to having large amounts of compression. The more an image is compressed, the more information is lost. If the image is compressed only once, there usually is no noticeable effect on the image. However, if the image is compressed and decompressed repeatedly, the effects of deleting information become more noticeable.

JPEG compression is commonly used for photographs, especially photographs that are transmitted or displayed on web pages. The compression makes the files small enough to transmit on a network efficiently, but leaves enough information that the image is still visually appealing.

- Encapsulated PostScript (EPS)

EPS is a PostScript graphics file format that follows conventions that Adobe Systems defined. EPS files support embedded ICC profiles.

- Graphics Interchange Format (GIF)

GIF files are bitmap image files that are limited to a palette of 256 RGB colors. Because of the limited color range that it can contain, GIF is not a good format for reproducing photographs, but it is typically adequate for logos or charts. GIF images are widely used on the Internet because they are usually smaller than other image formats. GIF files use the file extension .gif.

- Image Object Content Architecture (IOCA)

IOCA is an architecture that provides a consistent way to represent images, including conventions and directions for processing and exchanging image information. The architecture defines image information independently of all data objects and environments in which it might exist and uses self-identifying terms; each field contains a description of itself along with its contents.

- Portable Document Format (PDF)

PDF is a standard file format that Adobe Systems developed.

PDF files can be used and stored on various operating systems and contain all the required image and font data. Design attributes in a PDF are kept in a single compressed package.

**Note:** Single-page and multiple-page PDF files can be used as data objects in AFP print jobs.

- Portable Network Graphics (PNG)

PNG files are bitmap image files that support indexed colors, palette-based images with 24-bit RGB or 32-bit RGBA colors, grayscale images, an optional alpha channel, and lossless compression. PNG is used for transferring images on the Internet, but not for print graphics. PNG files use the file extension .png.

- Tagged Image File Format (TIFF)

TIFF files are bitmap image files that include headers to provide more information about the image. TIFF files use the file extensions .tif or .tiff.

TIFF files support embedded ICC profiles. If an ICC profile is embedded in a file, the characteristics of the input color space are known whenever the file is used; however, the profiles increase the file size. When you save a file in the TIFF format, you can use various compression algorithms.

**Note:** Single-image and multiple-image TIFF files can be used as data objects in AFP print jobs.



Not all printers support all types of data objects.

The embedded ICC profiles in EPS, JPEG, and TIFF files contain the information that a printer uses to convert colors in the image from an input color space into the profile connection space (PCS). The input color space might be an industry-standard space or it can describe the color reproduction capabilities of a device, such as a scanner, digital camera, monitor, or printer.

## Data object creation and installation

You can use a wide variety of software applications to create or manipulate images to include in print jobs. If you want to store them in central resource repositories, you can use an AFP resource installer to install them.

### Data object creation

Most types of data objects are images of some kind. They might be photographs that are taken with a digital camera, charts or diagrams generated by a software tool, or digital drawings created by using graphics software. Regardless of how images are created, you generally need to manipulate them to include them in print jobs.

The changes include:

- Convert the image into a file type that is appropriate for printing. For example, the file types that many graphics applications (such as Adobe Illustrator, CorelDRAW, and Corel Paint Shop Pro) use to store images while you work on them are not appropriate for printing. To use images that you create from any of those programs, you can save or export those files as a different file type, such as EPS, JPEG, or TIFF.
- Make sure that your image files are associated with an appropriate color space or input profile. Follow the instructions that are provided with your graphics software to set up color management, including installing and using ICC profiles for digital cameras and monitors, and customizing color management settings. The instructions must also explain how to change the color profile that an image uses and how to save an image with an embedded profile.
- Follow the tips and best practices provided in the information for creating images and managing them as data object resources.

### Data object installation

You can use an AFP resource installer to install your images in a resource library. An AFP resource installer includes wizards that can guide you through the process of installing an image as a data object. When you install an EPS, JPEG, or TIFF image with an embedded ICC profile by using an AFP resource installer, you can choose how you want to handle the profile:

- Leave the profile in the file without creating a CMR.
- Leave the profile in the file, but also copy the profile and create a CMR from the copy. Associate the new CMR with the data object.
- Remove the profile from the file (to reduce the file size) and make the profile into a CMR. Associate the new CMR with the data object.

## Resource library management

If you store CMRs and data objects in central resource libraries, you must understand some of the characteristics of resource libraries to make sure that your resources are available when and where you need them.

Resource libraries that an AFP resource installer creates use a *resource access table* (RAT) as the index of the resource library. The index is stored as a file in the library that it refers to. You must store CMRs in resource libraries that use a RAT. You must also store data objects in resource libraries that use a RAT.

When you use an AFP resource installer to create a resource library, it creates a RAT and stores it in the library. When you install a CMR or data object, the AFP resource installer updates the RAT with information about the resource. When a print server looks in a resource library for a resource, it first looks in the RAT to see whether the resource is listed.

The print server relies on the RAT; if it is incorrect, the print server cannot find resources in the resource library. As a result, you must always use an AFP resource installer to manage your resource libraries, including to:

- Add CMRs and data objects to a resource library.

Do not copy CMRs or data objects directly into the resource libraries that an AFP resource installer uses. If you copy CMRs or data objects into these resource libraries, the RAT is not updated so the print server cannot use it to find the CMRs or data objects.

- Modify properties of data objects and CMRs listed in the RAT.

Do not directly edit the RAT or any of the files in a resource library. Do not replace an existing version of a CMR or data object with a new version by copying the new version directly into the resource library; use an AFP resource installer to update the resource.

- Install CMRs or data objects in a different resource library or replicate a resource library in a different location.

Do not copy CMRs or data objects from a resource library and store them in another location.

For more information about completing these tasks, see the AFP resource installer help.

## Tips and best practices

These general guidelines about creating and managing images and other color resources can improve the performance of your AFP color printing system.

### Tips for images

To optimize the performance of your AFP color printing system, you must follow some guidelines for creating and including images in print jobs.

When you want to use color images in your print jobs:

- Get the original electronic versions of images instead of scanning existing documents.

Almost unnoticeable specks of color in the background of images that are scanned can greatly increase the size of the image. If you must scan an image, use an image editing tool to clean up the background as much as possible.

- Save all images in the same standard color space so you need only one input profile for all of them.

Adobe RGB (1998) is the recommended color space for images that are to be printed.

- Flatten multi-layer images (such as the ones you can create in graphics tools like Adobe Illustrator and Corel Paint Shop Pro) before you include them in print jobs.

Unflattened images are extremely large and more difficult to work with. Save a copy of the original image for future editing, but flatten the version that you include in your print job.

### Tips for resources

To optimize the performance of your AFP color printing system, you must follow some guidelines for managing color resources.

You can use an AFP resource installer to:

- Install all the CMRs for your printer in a resource library.
- Install the data objects that you use frequently in a resource library.
- Mark the CMRs and data objects that are reused regularly as non-private, capturable resources so they can be saved on the printer and used for other print jobs without being downloaded every time.

**Note:** This option is not advisable for secure resources, such as signature files.

- Install CMRs and data objects in resource libraries that the print server can access, so they need to be stored in one place only and can be used by all print servers.
- Associate audit color conversion CMRs with data objects that require color management, so the embedded profiles can be removed from the image files.

## AFP resource installer

---

An AFP resource installer is a key element of an AFP color management system when resources are stored in central libraries. You can use it to create, install, and manage color management resources (CMRs) and data objects for use in your system.

An AFP resource installer is a Java™ application that you install on a Windows workstation. You can use it to install and work with fonts in addition to CMRs and data objects.

You can use an AFP resource installer to:

- Create CMRs from existing data, including ICC profiles.

You can use a wizard to guide you through the process.

- Install CMRs, fonts, and data objects in resource libraries on the local system or on any system that you can access with FTP.
- Associate CMRs with data objects, so data objects can be reproduced accurately on different printers.

In some cases, you can reduce the file size of your images by removing the embedded color profile from the file and by using an associated CMR.

- Mark resources as capturable.

Capturable resources can be captured and saved in the printer for use with other print jobs, which can help improve system performance. The print server queries the printer before it sends any resources; if the printer already has the resource, the print server does not have to send it.

- Mark resources as private.

Private resources cannot be captured in the printer and must be downloaded with every print job that uses them. For example, you can mark signature files used for company checks as private for security reasons.

When you use an AFP resource installer to install a color conversion CMR, the software automatically creates link (LK) CMRs between the new color conversion CMR and the existing color conversion CMRs. When a print file references the new CMR, the print server automatically downloads the link CMRs that match the target device type and model and sends them to the printer with the print job. If one of those link CMRs is appropriate, the printer can use it instead of spending extra time creating a link CMR.

To let a print server use resources installed by an AFP resource installer, you must add the path to the resource libraries to the AFP resource path in the server.



---

## Appendix G. Accessibility

Accessible publications for this product are offered through [IBM Documentation \(www.ibm.com/docs/en/zos\)](http://www.ibm.com/docs/en/zos).

If you experience difficulty with the accessibility of any z/OS information, send a detailed message to the [Contact the z/OS team web page \(www.ibm.com/systems/campaignmail/z/zos/contact\\_z\)](http://www.ibm.com/systems/campaignmail/z/zos/contact_z) or use the following mailing address.

IBM Corporation  
Attention: MHVRCFS Reader Comments  
Department H6MA, Building 707  
2455 South Road  
Poughkeepsie, NY 12601-5400  
United States

---

### Accessibility features

Accessibility features help users who have physical disabilities such as restricted mobility or limited vision use software products successfully. The accessibility features in z/OS can help users do the following tasks:

- Run assistive technology such as screen readers and screen magnifier software.
- Operate specific or equivalent features by using the keyboard.
- Customize display attributes such as color, contrast, and font size.

---

### Consult assistive technologies

Assistive technology products such as screen readers function with the user interfaces found in z/OS. Consult the product information for the specific assistive technology product that is used to access z/OS interfaces.

---

### Keyboard navigation of the user interface

You can access z/OS user interfaces with TSO/E or ISPF. The following information describes how to use TSO/E and ISPF, including the use of keyboard shortcuts and function keys (PF keys). Each guide includes the default settings for the PF keys.

- *z/OS TSO/E Primer*
- *z/OS TSO/E User's Guide*
- *z/OS ISPF User's Guide Vol I*

---

### Dotted decimal syntax diagrams

Syntax diagrams are provided in dotted decimal format for users who access IBM Documentation with a screen reader. In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), they can appear on the same line because they are considered a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that the screen reader is set to read out punctuation. All the syntax elements that have the same dotted decimal number (for example, all the syntax elements that have the number 3.1)

are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, it is preceded by the backslash (\) character. The \* symbol is placed next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element \*FILE with dotted decimal number 3 is given the format 3 \\* FILE. Format 3\* FILE indicates that syntax element FILE repeats. Format 3\* \\* FILE indicates that syntax element \* FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol to provide information about the syntax elements. For example, the lines 5.1\*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, it indicates a reference that is defined elsewhere. The string that follows the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you must refer to separate syntax fragment OP1.

The following symbols are used next to the dotted decimal numbers.

#### **? indicates an optional syntax element**

The question mark (?) symbol indicates an optional syntax element. A dotted decimal number followed by the question mark symbol (?) indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element, (for example 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that the syntax elements NOTIFY and UPDATE are optional. That is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.

#### **! indicates a default syntax element**

The exclamation mark (!) symbol indicates a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicate that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the dotted decimal number can specify the ! symbol. For example, if you hear the lines 2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In the example, if you include the FILE keyword, but do not specify an option, the default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, the default FILE (KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP applies only to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

#### **\* indicates an optional syntax element that is repeatable**

The asterisk or glyph (\*) symbol indicates a syntax element that can be repeated zero or more times. A dotted decimal number followed by the \* symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1\* data area, you know that you can include one data area, more than one data area, or no data area. If you hear the lines 3\* , 3 HOST, 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

#### **Notes:**

1. If a dotted decimal number has an asterisk (\*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you can write HOST STATE, but you cannot write HOST HOST.
3. The \* symbol is equivalent to a loopback line in a railroad syntax diagram.

**+ indicates a syntax element that must be included**

The plus (+) symbol indicates a syntax element that must be included at least once. A dotted decimal number followed by the + symbol indicates that the syntax element must be included one or more times. That is, it must be included at least once and can be repeated. For example, if you hear the line 6.1+ data area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. Similar to the \* symbol, the + symbol can repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the \* symbol, is equivalent to a loopback line in a railroad syntax diagram.





## Notices

---

This information was developed for products and services that are offered in the USA or elsewhere.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing  
IBM Corporation  
North Castle Drive, MD-NC119  
Armonk, NY 10504-1785  
United States of America*

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

*Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan*

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

This information could include missing, incorrect, or broken hyperlinks. Hyperlinks are maintained in only the HTML plug-in output for IBM Documentation. Use of hyperlinks in other output formats of this information is at your own risk.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*IBM Corporation  
Site Counsel  
2455 South Road*

Poughkeepsie, NY 12601-5400  
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

## Terms and conditions for product documentation

---

Permissions for the use of these publications are granted subject to the following terms and conditions.

### **Applicability**

These terms and conditions are in addition to any terms of use for the IBM website.

### **Personal use**

You may reproduce these publications for your personal, noncommercial use provided that all proprietary notices are preserved. You may not distribute, display or make derivative work of these publications, or any portion thereof, without the express consent of IBM.

### **Commercial use**

You may reproduce, distribute and display these publications solely within your enterprise provided that all proprietary notices are preserved. You may not make derivative works of these publications, or

reproduce, distribute or display these publications or any portion thereof outside your enterprise, without the express consent of IBM.

## Rights

Except as expressly granted in this permission, no other permissions, licenses or rights are granted, either express or implied, to the publications or any information, data, software or other intellectual property contained therein.

IBM reserves the right to withdraw the permissions granted herein whenever, in its discretion, the use of the publications is detrimental to its interest or, as determined by IBM, the above instructions are not being properly followed.

You may not download, export or re-export this information except in full compliance with all applicable laws and regulations, including all United States export laws and regulations.

IBM MAKES NO GUARANTEE ABOUT THE CONTENT OF THESE PUBLICATIONS. THE PUBLICATIONS ARE PROVIDED "AS-IS" AND WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, AND FITNESS FOR A PARTICULAR PURPOSE.

## IBM Online Privacy Statement

---

IBM Software products, including software as a service solutions, ("Software Offerings") may use cookies or other technologies to collect product usage information, to help improve the end user experience, to tailor interactions with the end user, or for other purposes. In many cases no personally identifiable information is collected by the Software Offerings. Some of our Software Offerings can help enable you to collect personally identifiable information. If this Software Offering uses cookies to collect personally identifiable information, specific information about this offering's use of cookies is set forth below.

Depending upon the configurations deployed, this Software Offering may use session cookies that collect each user's name, email address, phone number, or other personally identifiable information for purposes of enhanced user usability and single sign-on configuration. These cookies can be disabled, but disabling them will also eliminate the functionality they enable.

If the configurations deployed for this Software Offering provide you as customer the ability to collect personally identifiable information from end users via cookies and other technologies, you should seek your own legal advice about any laws applicable to such data collection, including any requirements for notice and consent.

For more information about the use of various technologies, including cookies, for these purposes, see IBM's Privacy Policy at [ibm.com/privacy](http://ibm.com/privacy) and IBM's Online Privacy Statement at [ibm.com/privacy/details](http://ibm.com/privacy/details) in the section entitled "Cookies, Web Beacons and Other Technologies," and the "IBM Software Products and Software-as-a-Service Privacy Statement" at [ibm.com/software/info/product-privacy](http://ibm.com/software/info/product-privacy).

## Policy for unsupported hardware

---

Various z/OS elements, such as DFSMSdfp, JES2, JES3, and MVS, contain code that supports specific hardware servers or devices. In some cases, this device-related element support remains in the product even after the hardware devices pass their announced End of Service date. z/OS may continue to service element code; however, it will not provide service related to unsupported hardware devices. Software problems related to these devices will not be accepted for service, and current service activity will cease if a problem is determined to be associated with out-of-support devices. In such cases, fixes will not be issued.

## Minimum supported hardware

---

The minimum supported hardware for z/OS releases identified in z/OS announcements can subsequently change when service for particular servers or devices is withdrawn. Likewise, the levels of other software products supported on a particular release of z/OS are subject to the service support lifecycle of those

products. Therefore, z/OS and its product publications (for example, panels, samples, messages, and product documentation) can include references to hardware and software that is no longer supported.

- For information about software support lifecycle, see: [IBM Lifecycle Support for z/OS \(www.ibm.com/software/support/systemsz/lifecycle\)](http://www.ibm.com/software/support/systemsz/lifecycle)
- For information about currently-supported IBM hardware, contact your IBM representative.

## Trademarks

---

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available at [Copyright and Trademark information \(www.ibm.com/legal/copytrade.shtml\)](http://www.ibm.com/legal/copytrade.shtml).

Adobe and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

# Glossary

---

This glossary defines technical terms and abbreviations used in PSF for z/OS documentation.

These cross-references are used in this glossary:

**See**

Refers to preferred synonyms or to defined terms for acronyms and abbreviations.

**See also**

Refers to related terms that have similar, but not synonymous, meanings, or to contrasted terms that have opposite or substantively different meanings.

## A

**abend**

See [abnormal end of task](#).

**abnormal end of task (abend)**

The termination of a task, job, or subsystem because of an error condition that recovery facilities cannot resolve during processing

**above the bar storage**

Storage above the 2 GB bar. The system programmer can specify that all inline resources are stored in this area during job processing. This is useful when there might not be enough virtual storage in the region area (above the 16 MB line and below the 2 GB bar).

**access method**

A technique for moving data between main storage and input/output devices.

**active environment group**

A collection of mapping structured fields, positioning controls, and data descriptors that define the environment for a page. These structured fields form an internal object in a composed text page, page definition, or overlay.

**addressable point**

For page printers, any defined position or picture element in a presentation surface or physical medium that can be referenced. See also [picture element](#) and [print position](#).

**Advanced Function Presentation (AFP)**

A set of licensed programs, together with user applications, that use the all-points-addressable concept to print data on a wide variety of printers or to display data on a variety of display devices. AFP includes creating, formatting, archiving, retrieving, viewing, distributing, and printing information.

**AFP**

See [Advanced Function Presentation](#).

**AFP Font Collection**

An IBM licensed product that includes a set of utilities, and a single font source for all AFP operating systems.

**AFP resource installer**

An application that runs on a Windows workstation and installs and manages fonts, data objects, and color management resources (CMRs) in resource libraries. It also creates CMRs and associates CMRs with data objects.

**AFP Statistics (AFPSTATS) report**

Contains summary data about the resources used to print a document. The AFPSTATS report is used to indicate in which libraries PSF found a resource, diagnose some resource selection problems, obtain statistical data about how a print file is printed, and diagnose some print file printing performance problems.

**AFPSTATS report**

See [AFP Statistics report](#).

**AFPSTATS repository**

A data set where AFP Statistics (AFPSTATS) reports are written.

**AFP Toolbox**

A product that assists application programmers in formatting printed output. Without requiring knowledge of the AFP data stream, AFP Toolbox provides access to sophisticated AFP functions through a callable C, C++, or COBOL interface.

**all-points addressability (APA)**

The capability to address, reference, and position text, overlays, and images at any defined position or picture element on the printable area of the paper. This capability depends on the ability of the hardware to address and to display each picture element.

**all-points addressable (APA)**

Pertaining to addressing, referencing, and positioning text, overlays, and images at any defined position or picture element on the printable area of the paper.

**American Standard Code for Information Interchange (ASCII)**

A standard code used for information exchange among data processing systems, data communication systems, and associated equipment. ASCII uses a coded character set consisting of 7-bit coded characters. See also [Extended Binary Coded Decimal Interchange Code](#).

**APA**

See [all-points addressability](#) or [all-points addressable](#).

**APAR**

See [authorized program analysis report](#).

**application program**

A program used to communicate with stations in a network, enabling users to perform application-oriented activities.

**ASCII**

See [American Standard Code for Information Interchange](#).

**authorized program analysis report (APAR)**

A request for correction of a defect in a supported release of an IBM-supplied program.

**auxiliary data set**

In AFP printing, a data set that contains job header, data set header, job trailer, or message data. See also [print data set](#).

**auxiliary resource**

Fonts, page segments, overlays, page definitions, or form definitions associated with auxiliary data sets.

**B****bar code**

An array of elements, such as bars, spaces, and two-dimensional modules, that encode data in a particular symbology. The elements are arranged in a predetermined pattern following unambiguous rules defined by the symbology.

**Bar Code Object Content Architecture (BCOCA)**

An architected collection of constructs used to interchange and present bar code data.

**baseline**

A conceptual line with respect to which successive characters are aligned.

**BCOCA**

See [Bar Code Object Content Architecture](#).

**big endian**

Pertaining to the order in which binary data is stored or transmitted with the most significant byte placed first. See also [little endian](#).

**bin**

An enclosure on a printer that contains source or destination media, including paper, foils, labels, card stock, or microfilm. See also [cassette](#) and [stacker](#).

**bounded-box font**

A font in bounded-box format. See also [unbounded-box font](#).

**bounded-character box**

A character box that does not contain blank space on any sides of the character. See also [unbounded-character box](#).

**BTS**

See [burster-trimmer-stacker](#).

**burst**

To separate continuous-forms paper into separate sheets.

**burster-trimmer-stacker (BTS)**

An optional printer feature that separates continuous forms into separate sheets, trims the carrier strip from both edges of the paper, and stacks the sheets. The BTS also identifies jobs by offsetting the stacking.

**C****carriage control character**

A character that is used to specify a write, space, or skip operation. See also [control character](#).

**cassette**

In cut-sheet printers, a removable container for a supply of paper. See also [bin](#).

**CFS**

See [continuous-forms stacker](#).

**channel-attached**

Pertaining to the attachment of devices directly by input/output channels to a host processor. See also [SNA-attached](#) and [TCP/IP-attached](#).

**channel code**

A number from 1 to 12 that identifies a position in the forms control buffer or a page definition.

**character**

1. Any symbol that can be entered on a keyboard, printed, or displayed. For example, letters, numbers, and punctuation marks are all characters.
2. In a computer system, a member of a set of elements that is used for the representation, organization, or control of data. See also [control character](#), [glyph](#), and [graphic character](#).
3. In bar codes, a single group of bars and spaces that represent an individual number, letter, punctuation mark, or other symbol.

**character box**

The area that completely contains the character pattern.

**character data**

Data in the form of letters and special characters, such as punctuation marks. See also [numeric data](#).

**character identifier**

The standard identifier for a character, regardless of its style. For example, all uppercase A's have the same character identifier. See also [graphic character identifier](#).

**character rotation**

The alignment of a character with respect to its character baseline, measured in degrees in a clockwise direction. See also [rotation](#) and [orientation](#).

**character set**

A defined set of characters that can be recognized by a configured hardware or software system. A character set can be defined by alphabet, language, script, or any combination of these items. See also [font character set](#).

**checkpoint**

A place in a program at which a check is made, or at which a recording of data is made to allow the program to be restarted in case of interruption.

**client**

A software program or computer that requests access to data, services, programs, and resources from a server. See also [server](#) and [host](#).

**CMR**

See [color management resource](#).

**coded font**

A font file that associates a code page and a font character set. For double-byte fonts, a coded font associates multiple pairs of code pages and font character sets.

**coded font section**

A font character set and code page pair. A single-byte coded font consists of only one coded font section; a double-byte coded font can consist of more than one.

**code page**

A particular assignment of code points to graphic characters. Within a given code page, a code point can only represent one character. A code page also identifies how undefined code points are handled. See also [coded font](#) and [extended code page](#).

**code point**

A unique bit pattern that represents a character in a code page.

**color management resource (CMR)**

An object that provides color management in presentation environments.

**color mapping table**

A MO:DCA object that is used to map color values specified in a source color space to color values specified in a target color space. This object is loaded into printers that support the color mapping table.

**color selection**

The ability to specify a color other than black to print data in more than one color. Some printers support selection of several colors, depending upon the color of the ribbon installed in the printer. Other printers support the selection of black or *color of media*, which can cause white lettering on a background that has been shaded black.

**command**

A request from a terminal or automated operator for the performance of an operation or service, or a request in a batch-processing job or print file for the operation or execution of a particular program.

**communication**

See [data communication](#).

**compatibility font**

An AFP raster font designed to emulate the uniformly spaced and fixed-pitch fonts used with line printers.

**complex text**

Unicode-encoded text that cannot be rendered in the traditional one-code-point to one-glyph fashion, such as bidirectional Arabic text or combined Hindi characters.

**composed text**

Text that has been formatted and that contains text-control information to direct the presentation of the text.

**computing system RPQ**

A customer request for a price quotation on alterations or additions to the functional capabilities of a computing system, hardware product, or device. The RPQ can be used in conjunction with programming RPQs to solve unique data processing problems. See also [programming request for price quotation](#).

**concatenate**

1. To link together.



2. To join two character strings.

**concatenated data set**

A group of logically connected data sets that are treated as one data set for the duration of a job step. See also [data set](#), [partitioned data set](#), and [library](#).

**conditional processing**

A page definition function that allows input data records to partially control their own formatting.

**console**

A display station from which an operator can control and observe the system operation.

**continuous forms**

A series of connected forms that feed continuously through a printing device. The connection between the forms is perforated so that the user can tear them apart. Before printing, the forms are folded in a stack, with the folds along the perforations. See also [cut-sheet paper](#).

**continuous-forms stacker (CFS)**

In continuous-forms printers, an output assembly that refolds and stacks continuous forms after printing.

**control character**

1. A character that represents a command that is sent to an output device, such as a printer or monitor. Examples are line-feed, shift-in, shift-out, carriage return, font change, and end of transmission. See also [carriage control character](#).
2. A character whose occurrence in a particular context initiates, modifies, or stops a control function.

**copy group**

An internal object in a form definition or a print data set that controls such items as modifications to a form, page placement, and overlays. See also [internal copy group](#).

**core interchange font**

A uniformly spaced typographic font with specialized characters for different languages.

**current print position**

The picture element that defines the character reference point or the upper-left corner of an image.

**cut-sheet paper**

Paper that is cut into uniform-size sheets before it is loaded into the printer. See also [continuous forms](#).

**D****data check**

A synchronous or asynchronous indication of a condition caused by erroneous data or incorrect positioning of data. Some data checks can be suppressed.

**data communication**

Transfer of data among functional units by means of data transmission protocols.

**data control block (DCB)**

A control block used by access method routines in storing and retrieving data.

**Data Facility Storage Management Subsystem (DFSMS)**

An operating environment that helps automate and centralize the management of storage. To manage storage, the storage management subsystem (SMS) provides the storage administrator with control over data class, storage class, management class, storage group, and automatic class selection (ACS) routine definitions.

**data map**

An internal object in a page definition that specifies fonts, page segments, fixed text, page size, and the placement and orientation of text.

**data object**

An object that is either specified within a page or overlay or is identified as a resource by using the Map Data Resource (MDR) structured field and later included in a page or overlay. Examples include:

PDF single-page and multiple-page objects, Encapsulated PostScript (EPS) objects, and IOCA images. See also [data object resource](#) and [resource](#).

**data object resource**

An object container resource or IOCA image resource that is either printer resident or downloaded. Data object resources can be:

- Used to prepare for the presentation of a data object, such as with a resident color profile resource object
- Included in a page or overlay through the Include Object (IOB) structured field; for example, PDF single-page and multiple-page objects, Encapsulated PostScript (EPS) objects, and IOCA images
- Called from within a data object; for example, PDF resource objects

**data set**

The major unit of data storage and retrieval, consisting of a collection of data in one of several prescribed arrangements and described by control information to which the system has access. See also [file](#), [concatenated data set](#), [partitioned data set](#), and [sequential data set](#).

**data set header**

A page in printed output that separates multiple data sets or multiple copies of a data set within a print job. See also [job header](#).

**DCB**

See [data control block](#).

**DCF**

See [Document Composition Facility](#).

**default**

Pertaining to an attribute, value, or option that is assumed when none is explicitly specified.

**deferred-printing mode**

A printing mode that spools output through JES to a data set instead of printing it immediately. Output is controlled by using JCL statements. See also [direct-printing mode](#).

**DFSMS**

See [Data Facility Storage Management Subsystem](#)

**direct-printing mode**

A printing mode that gives PSF exclusive use of a channel-attached printer. Output is printed immediately and is not spooled through JES. See also [deferred-printing mode](#).

**disabled mechanism**

A function of a printer that is temporarily out of operation or is not supported. In such a case, the device manager, such as PSF, might allow jobs to print with alternative options. See also [enabled](#).

**distributed print function (DPF)**

A component of a Windows print server that can be installed and used to print jobs from PSF.

**document**

1. A machine-readable collection of one or more objects that represent a composition, a work, or a collection of data.
2. Data that has already been composed into pages and that contains a Begin Document and an End Document structured field.

**Document Composition Facility (DCF)**

An IBM licensed program used to format input to a printer.

**double-byte coded font**

A font in which the characters are defined by 2 bytes. The first byte defines the coded font section; the second byte defines the code point in the code page specified for that section. See also [single-byte coded font](#).

**download**

To transfer data from a computer to a connected device, such as a workstation or a printer. Typically, users download from a large computer to a diskette or fixed disk on a smaller computer or from a system unit to an adapter.

**DPF**

See [distributed print function](#).

**duplex**

Pertaining to printing on both sides of a sheet of paper. See also [normal duplex](#), [simplex](#), and [tumble duplex](#).

**E****EBCDIC**

See [Extended Binary Coded Decimal Interchange Code](#).

**electronic form**

A collection of constant data that is electronically composed in the host processor and can be merged with variable data on a page during printing.

**enabled**

1. Pertaining to a state of the processing unit that allows the occurrence of certain types of interruptions.
2. A condition of the printer (physically selected) in which the printer is available to the host processor for typical work. The printer is online when in an enabled condition. See also [disabled mechanism](#).

**ERP**

See [error-recovery procedure](#).

**error-recovery procedure (ERP)**

A procedure designed to help isolate and, where possible, to recover from errors in equipment. The procedure is often used in conjunction with programs that record information about machine malfunctions.

**exception**

A condition or event that cannot be handled by a normal process.

**exception highlighting**

The markings placed on the printed page to indicate the location of a data-stream error.

**Extended Binary Coded Decimal Interchange Code (EBCDIC)**

A coded character set of 256 eight-bit characters developed for the representation of textual data. EBCDIC is not compatible with ASCII character coding. See also [American Standard Code for Information Interchange](#).

**extended code page**

A code page that is stored in a partitioned data set (PDS or PDSE) in a font resource library or in a UNIX file in a font path library. Extended code pages might contain Unicode values that a printer uses to print EBCDIC or ASCII encoded text strings with TrueType and OpenType fonts.

**F****FCB**

See [forms control buffer](#).

**file**

1. A collection of related data that is stored and retrieved by an assigned name. A file can include information that starts a program (program-file object), contains text or graphics (data-file object), or processes a series of commands (batch file).
2. See also [data set](#), [partitioned data set](#), [sequential data set](#), and [library](#).

## **FOCA**

See [Font Object Content Architecture](#).

## **font**

1. A family or assortment of characters of a given size and style, for example, 9-point Bodoni modern. A font has a unique name and might have a registry number.
2. A particular type style (for example, Bodoni or Times Roman) that contains definitions of character sets, marker sets, and pattern sets. See also [coded font](#) and [double-byte coded font](#).

## **font character set**

1. Part of an AFP font that contains the raster patterns, identifiers, and descriptions of characters. See also [character set](#).
2. A Font Object Content Architecture (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 architecture that defines the content of digital font resources by means of a set of parameter definitions.

## **font section**

A subdivision of a double-byte font character set. The section consists of a maximum of 256 characters. See also [coded font section](#).

## **form**

1. A physical piece of paper or other medium on which data is printed. See also [medium](#), [page](#), and [sheet](#).
2. A display screen, printed document, or file with defined spaces for information to be inserted.

## **format**

The shape, size, printing requirements, and general makeup of a printed document or presentation display.

## **formatted print records**

Traditional line data made up of records that are formatted for printing on line printers. PSF uses a page definition to print formatted records on page printers.

## **form definition**

An AFP resource object used by PSF that defines the characteristics of the form or printed media, including: overlays to be used, duplex printing, text suppression, the position of composed-text data on the form, and the number and modifications of a page.

## **forms control buffer (FCB)**

A buffer for controlling the vertical format of printed output. The FCB is a line-printer control that is similar to the punched-paper, carriage-control tape used on IBM 1403 printers. For AFP page printers, the forms control buffer is replaced by the page definition. See also [page definition](#).

## **forms flash**

In AFP support on the 3800 Printing Subsystem, a means of printing an overlay by using a negative plate projected on a form.

## **G**

### **global resource identifier (GRID)**

An 8-byte identifier that identifies a coded font resource. A GRID contains these fields, in the order listed:

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, 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 (FW), in 1440ths of an inch.

## **glyph**

1. A graphic symbol whose appearance conveys information, for example, the vertical and horizontal arrows on cursor keys that indicate the directions in which they control cursor movement.
2. An image, typically of a character, in a font. See also [character](#) and [graphic character](#).

## **GOCA**

See [Graphics Object Content Architecture](#).

## **graphical user interface (GUI)**

A type of computer interface that presents a visual metaphor of a real-world scene, often of a desktop, by combining high-resolution graphics, pointing devices, menu bars and other menus, overlapping windows, icons and the object-action relationship. See also [programming interface for customers](#).

## **graphic character**

1. A visual representation of a character, other than a control character, that is typically produced by writing, printing, or displaying. See also [glyph](#).
2. A member of a set of symbols that represent data. Graphic characters can be letters, digits, punctuation marks, or other symbols.

## **graphic character identifier**

The unique name for a graphic character in a font or in a graphic character set. See also [character identifier](#).

## **Graphics Object Content Architecture (GOCA)**

An architecture that provides a collection of graphics values and control structures used to interchange and present graphics data.

## **GRID**

See [global resource identifier](#).

## **GUI**

See [graphical user interface](#).

# **H**

## **hardcopy**

A printed copy of machine output in a visually readable form, such as printed reports, documents, and summaries. See also [softcopy](#).

## **hard resource**

A resource declared in the appropriate Map structured field and loaded in the printer the first time it is referenced. It can be reused during the job without being reloaded to the printer. See also [soft resource](#).

## **hardware default font**

The font used by the printer if no other font is specified.

## **hexadecimal**

Pertaining to a numbering system that has a base of 16.

## **HFS**

See [hierarchical file system](#).

## **hierarchical file system (HFS)**

A system for organizing files in a hierarchy, as in a UNIX system.

## **host**

1. A computer that is connected to a network and provides an access point to that network. The host can be a client, a server, or both a client and server simultaneously. See also [client](#) and [server](#).
2. In TCP/IP, any system that has at least one Internet Protocol address associated with it.

## **host font**

See [host resource](#).

**host resource**

A resource found either in a system library, in a user library, or inline in the print data set.

**host system**

See [host](#).

**I****image**

1. A pattern of toned and untoned pels that form a picture. See also [impression](#).
2. An electronic representation of an original document or picture produced by a scanning device or created from software.

**image data**

1. A pattern of bits with 0 and 1 values that define the pels in an image. A 1-bit is a toned pel.
2. Digital data derived from electrical signals that represent a visual image.
3. Rectangular arrays of raster information that define an image.

**Image Object Content Architecture (IOCA)**

An architecture that provides a collection of constructs used to interchange and present images, such as printing image data on a page, page segment, or overlay.

**impact printer**

A printer in which printing is the result of mechanically striking the printing medium. See also [nonimpact printer](#).

**impression**

The transfer of an image to a sheet of paper. Multiple impressions can be printed on each side of a sheet. Printer speed is often measured in impressions per minute (ipm).

**Infoprint Fonts for z/OS**

The outline version of the IBM Expanded Core Fonts.

**inline**

Pertaining to spooled input data that is read into a job by a reader. See also [inline resource](#).

**inline direction**

The direction in which successive characters are added to a line of text.

**inline resource**

A resource contained in a print file or a print data set.

**input/output (I/O)**

Pertaining to a device, process, channel, or communication path involved in data input, data output, or both.

**installation exit**

The means specifically described in an IBM software product's documentation by which an IBM software product can be modified by a customer's system programmers to change or extend the functions of the IBM software product. Such modifications consist of exit routines written to replace one or more existing modules of an IBM software product, or to add one or more modules or subroutines to an IBM software product.

**Intelligent Printer Data Stream (IPDS)**

An all-points-addressable data stream that lets users position text, images, graphics, and bar codes at any defined point on a printed page. IPDS is the strategic AFP printer data stream generated by PSF.

**interface**

A shared boundary between independent systems. An interface can be a hardware component used to link two devices, a convention that supports communication between software systems, or a method for a user to communicate with the operating system, such as a keyboard.

**intermediate device**

A device that operates on the data stream and is situated between a printer and a presentation services program in the host. Examples include devices that capture and store resources and devices that spool the data stream.

**internal copy group**

A copy group in a print data set instead of in a form definition. See also [copy group](#).

**internal medium map**

See [internal copy group](#).

**internal object**

A structured field that can be included as part of a resource or a print job (data set or file), but that cannot be accessed separately.

**I/O**

See [input/output](#).

**IOCA**

See [Image Object Content Architecture](#).

**IPDS**

See [Intelligent Printer Data Stream](#).

**J****JCL**

See [job control language](#).

**JES**

See [Job Entry Subsystem](#).

**JES2**

An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for processing, processes their output, and purges them from the system. In an installation with more than one processor, each JES2 processor independently controls its job input, scheduling, and output processing. See also [Job Entry Subsystem](#) and [JES3](#).

**JES3**

An MVS subsystem that receives jobs into the system, converts them to internal format, selects them for processing, processes their output, and purges them from the system. In complexes that have several loosely coupled processing units, the JES3 program manages processors so that the global processor exercises centralized control over the local processors and distributes jobs to them by using a common job queue. See also [Job Entry Subsystem](#) and [JES2](#).

**job control language (JCL)**

A command language that identifies a job to an operating system and describes the job's requirements.

**Job Entry Subsystem (JES)**

An IBM licensed program that receives jobs into the system and processes all output data that is produced by jobs. See also [JES2](#) and [JES3](#).

**job header**

A page in printed output that indicates the beginning of a user job. A user job can contain one or more data sets, or one or more copies of a print job. See also [data set header](#).

**job trailer**

A page in the printed output that indicates the end of a user job.

**K****Kanji**

A graphic character set consisting of symbols used in Japanese ideographic alphabets. Each character is represented by 2 bytes.

## L

### landscape page presentation

The position of a printed sheet that has its long edges as the top and bottom and its short edges as the sides. See also [portrait page presentation](#).

### library

1. A system object that serves as a directory to other objects. A library groups related objects, and allows the user to find objects by name.
2. A data file that contains copies of a number of individual files and control information that allows them to be accessed individually.
3. A partitioned data set or a series of concatenated partitioned data sets.

### library member

A named collection of records or statements in a library. See also [resource object](#).

### line data

Data prepared for printing on a line printer without any data placement or presentation information. Line data can contain carriage-control characters and table-reference characters (TRC) for spacing and font selections. See also [record format line data](#) and [traditional line data](#).

### line descriptor

Specifications that describe how traditional line data records are formatted into individual print lines. Line descriptors are interpreted by PSF when formatting printed output.

### line merging

The process of printing two or more records of traditional line data at the same location on the page. Line merging is used with traditional line data to mix different fonts on the same line, to underscore or overstrike, and, on impact printers, to create darker print.

### line printer

A device that prints a line of characters as a unit. See also [page printer](#).

### lines per inch (lpi)

1. The number of characters that can be printed vertically within an inch.
2. A unit of measurement for specifying the placement of the baseline.

### little endian

Pertaining to the order in which binary data is stored or transmitted with the least significant byte placed first. See also [big endian](#).

### logical page

The defined presentation space on the physical form. All the text and images in the print data must fit within the boundaries of the logical page, which has specified characteristics, such as size, shape, orientation, and offset. See also [form](#) and [physical page](#).

### logical page origin

The point on the logical page from which positions of images, graphics, page overlays, and text with 0-degree inline direction are measured.

### logical unit (LU, L-unit)

1. A unit of linear measurement. For example, in Mixed Object Document Content Architecture (MO:DCA) and AFP data streams, these measurements are used:  
 $1 \text{ L-unit} = 1/1440 \text{ inch}$   
 $1 \text{ L-unit} = 1/240 \text{ inch}$
2. An access point through which a user or application program accesses the SNA network to communicate with another user or application program. An LU can support at least two sessions, one with an SSCP and one with another LU, and might be capable of supporting many sessions with other LUs.

### lpi

See [lines per inch](#).



## **LU**

See [logical unit](#).

## **L-unit**

See [logical unit](#).

## **M**

### **magnetic ink character recognition (MICR)**

The identification of characters through the use of magnetic ink.

### **mandatory print labeling (MPL)**

A class, defined to RACF, that causes PSF to automatically label separator pages and data pages and to enforce the user printable area.

### **marking**

A method of updating certain structured fields to identify a resource as printer-resident.

### **media origin**

The reference point from which the logical page origin is positioned by the medium map. This point is represented by  $X_m=0$ ,  $Y_m=0$  in the  $X_m$ ,  $Y_m$  coordinate system. The media origin is defined relative to the upper-left corner of the form. See also [logical page origin](#). See also [logical page origin](#).

### **medium**

1. The material on which computer information is stored. Examples of media are diskettes, CDs, DVDs, and tape.
2. The physical material, such as paper, on which data is printed. See also [form](#), [page](#), and [sheet](#).

### **medium map**

See [copy group](#).

### **medium overlay**

An electronic overlay that is called by the medium map of a form definition for printing at a fixed position on the form. See also [page overlay](#).

### **member name**

The name under which a file is stored in a library. For example, X1BITR is the member name of a font in the font library.

### **message data set**

1. In PSF, a virtual data set built by the library access system interface (LASI) subcomponent in memory to store error messages for printing at the end of the document.
2. A data set on disk storage that contains queues of messages awaiting transmission to particular terminal operators or to the host system.

## **MICR**

See [magnetic ink character recognition](#).

### **microfilm device**

An output device that presents a hardcopy on microfilm.

### **microfilm setup resource**

A setup file that contains information used to present AFP data on microfilm. See also [object container](#).

### **migration**

The movement of data when software is upgraded or the data is transferred to a different hardware server or model.

### **Mixed Object Document Content Architecture (MO:DCA)**

An architected, device-independent data stream for interchanging documents.

### **Mixed Object Document Content Architecture for Presentation (MO:DCA-P)**

The subset of MO:DCA that defines presentation documents. PSF supports MO:DCA Presentation Interchange Set data streams.

**mixed-pitch font**

A font that simulates a proportionally spaced or typographic font. The characters are in a limited set of pitches (for example, 10 pitch, 12 pitch, and 15 pitch).

**MO:DCA**

See [Mixed Object Document Content Architecture](#).

**MO:DCA AFP/Archive (MO:DCA AFP/A)**

An AFP document architecture interchange set that is used for long-term preservation and retrieval. This subset ensures page independence and eliminates images without clearly specified resolution, device default fonts, and external resources.

**MO:DCA AFP/A**

See [MO:DCA AFP/Archive](#).

**MO:DCA AFP/A, IS/3**

An AFP document architecture interchange set that complies with the rules and restrictions of both the AFP/Archive and IS/3 interchange sets.

**MO:DCA data**

Print data that has been composed into pages. Text-formatting programs (such as DCF) can produce composed text data consisting entirely of structured fields. ACIF or AFP Download Plus can transform line data or XML data to MO:DCA data.

**MO:DCA GA**

See [MO:DCA Graphic Arts Function Set](#).

**MO:DCA Graphic Arts Function Set (MO:DCA GA)**

An extension of MO:DCA IS/3 that provides support for PDF presentation object containers.

**MO:DCA IS/1**

See [MO:DCA Presentation Interchange Set 1](#).

**MO:DCA IS/3**

See [MO:DCA Presentation Interchange Set 3](#).

**MO:DCA-P**

See [Mixed Object Document Content Architecture for Presentation](#).

**MO:DCA Presentation Interchange Set 1 (MO:DCA IS/1)**

A subset of MO:DCA that defines an interchange format for presentation documents.

**MO:DCA Presentation Interchange Set 3 (MO:DCA IS/3)**

A subset of MO:DCA that defines an interchange format for presentation documents. The MO:DCA IS/3 data stream includes structured fields that are not found in MO:DCA IS/1.

**monospaced font**

A font in which the spacing of the characters does not vary. See [uniformly spaced font](#). See also [proportionally spaced font](#).

**MPL**

See [mandatory print labeling](#).

**multiple up**

The printing of more than one page on a single surface of a sheet of paper.

**Multiple Virtual Storage (MVS)**

An IBM operating system that accesses multiple address spaces in virtual storage.

**MVS**

See [Multiple Virtual Storage](#).

**N****nonimpact printer**

A printer in which printing is not the result of mechanical impacts, for example, a thermal printer, an electrostatic printer, and a photographic printer. See also [impact printer](#).

**normal duplex**

Pertaining to printing on both sides of the paper such that the top of one side is at the same end as the top of the other side. Normal duplex printing is used for forms that are bound on the long edge of the paper, regardless of whether the printing is portrait or landscape. See also [duplex](#) and [tumble duplex](#).

**numeric data**

Data represented by numerals. See also [character data](#).

**N\_UP**

The partitioning of a side of a sheet into a fixed number of equal size partitions. For example, N\_UP 4 divides each side of the sheet into four equal partitions. In enhanced N\_UP printing, the sheet can be divided into 8 partitions, each of which can be anywhere on a single side of the sheet.

**O****object**

In AFP architecture, a collection of structured fields, bounded by a begin-object function and an end-object function. The object can contain other structured fields containing data elements of a particular type. Examples of objects are text, fonts, graphics, images, and bar codes.

**object container**

A MO:DCA structure that carries object data, which might or might not be defined by a presentation architecture.

**offset stacking**

A function that allows the printed output pages to be offset for easy separation of the print jobs.

**OGL**

See [Overlay Generation Language](#).

**OpenType font**

An extension of the TrueType font format that adds support for PostScript outlines and more support for international character sets and advanced typographic control.

**option**

A specification in a statement that can influence the running of the statement.

**orientation**

In printing, the number of degrees an object is rotated relative to a reference; for example, the orientation of an overlay relative to the logical page origin, or the orientation of printing on a page relative to the page coordinates. Orientation typically applies to blocks of information, whereas character rotation applies to individual characters. See also [character rotation](#).

**origin**

1. A position from which the placement and orientation of an element is specified.
2. The point in a coordinate system where the axes intersect. Examples of origins are the addressable position in an X m ,Ym coordinate system where both coordinate values are zero and the character reference point in a character coordinate system.

**outline font**

A font whose graphic character shapes are defined by mathematical equations rather than by raster patterns. See also [raster font](#).

**overlay**

1. A resource object that contains predefined presentation data, such as text, image, graphics, and bar code data, that can be merged with variable data on a page or form while printing. See also [page overlay](#) and [medium overlay](#).
2. The final representation of a collection of predefined presentation data on a physical medium.

**Overlay Generation Language (OGL)**

An IBM licensed program used for designing objects (such as lines, boxes, shadings, and irregular shapes) for electronic overlays.

## P

### page

1. A collection of data that can be printed on one side of a sheet of paper or a form.
2. A data stream object delimited by a Begin Page structured field and an End Page structured field. A page can contain presentation data such as text, image, graphics, and bar code data. See also [logical page](#) and [physical page](#).

### page definition

An AFP resource object used by PSF that defines the rules for transforming line data and XML data into MO:DCA data and text controls, such as width of margins and text orientation.

### page format

See [data map](#).

### page mode

The mode of operation in which a page printer can accept an entire page of data from a host processor to be printed on an all-points-addressable output medium. A page of data can consist of text, images, overlays, and page segments.

### page origin

See [logical page origin](#).

### page overlay

An electronic overlay that can be called for printing and positioned at any point on the page by an Invoke Page Overlay structured field in the print data. See also [medium overlay](#).

### page position

A control in the copy group to assign the upper-left boundary point of the logical page on a sheet for a data set. The page position is determined from the media origin.

### page printer

1. In AFP support, any of a class of printers that accepts composed pages, constructed of composed text and images, among other things. See also [line printer](#).
2. A device that prints one page at a time.

### Page Printer Formatting Aid (PPFA)

An IBM licensed program with which to create and store form definitions and page definitions, which are resource objects used for print-job management. These stored objects are used to format printed output.

### page segment

An AFP resource object containing text, image, graphics, or bar code data that can be positioned on any addressable point on a page or an electronic overlay.

### parameter

A value or reference passed to a function, command, or program that serves as input or controls actions. The value is supplied by a user or by another program or process.

### partition

In basic N\_UP printing, the division of the medium presentation space into a specified number of equal-sized areas in a manner determined by the current physical medium.

### partitioned data set (PDS)

A data set in direct-access storage that is divided into partitions, called members, each of which can contain a program, part of a program, or data. See also [sequential data set](#).

### PDS

See [partitioned data set](#).

### pel

See [picture element](#).

### physical medium

A physical entity on which information is presented; for example, a sheet of paper, a roll of paper, microfilm, an envelope, label, or display screen.

**physical page**

A single surface (front or back) of a form. See also [form](#), [logical page](#), and [page](#).

**picture element (pel, pixel)**

1. An element of a raster pattern about which a toned area on the photoconductor might appear. When used with a number, *pel* indicates resolution. Examples include 240-pel and 300-pel.
2. The smallest printable or displayable unit that can be displayed. A common measurement of device resolution is picture elements per inch. Typical monitors display between 72 and 96 pixels per inch. Characters and graphics are created by turning pixels on or off.

**pitch**

A unit of measurement for the width of type (or a printed character), based on the number of characters that can be set (or printed) in one linear inch; for example, 10-pitch has 10 characters per inch. Uniformly spaced fonts are measured in pitch. See also [point](#).

**pixel**

See [picture element](#).

**point**

A unit of measurement used mainly for describing type sizes. Each pica has 12 points, and an inch has approximately 72 points. See also [pitch](#).

**point size**

The height of a font in points.

**portrait page presentation**

The position of a printed sheet that has its short edges as the top and bottom and its long edges as the sides. See also [landscape page presentation](#).

**PostScript**

A page description language developed by Adobe Systems, Incorporated that describes how text and graphics are presented on printers and display devices.

**PPFA**

See [Page Printer Formatting Aid](#).

**preprinted form**

A sheet of paper containing a preprinted design of constant data on which variable data can be printed.

**presentation text**

See [composed text](#).

**Presentation Text Object Content Architecture (PTOCA)**

An architecture that provides a collection of constructs used to interchange and present presentation text data, such as printing text data on a page, page segment, or overlay.

**printable area**

The area on a sheet of paper where print can be placed.

**print data set**

A data set created by an application program that contains the actual information to be printed and, optionally, some of the data that controls the format of the printing. The types of print data sets are composed text, line format, XML data, and mixed format. See also [auxiliary data set](#) and [print file](#).

**print direction**

A combination of the inline direction and the baseline direction.

**Printer Control Language (PCL)**

The Hewlett Packard page description language that is used in laser and ink-jet printers.

**print file**

A file that is created for the purpose of printing data. A print file includes information to be printed and, optionally, some of the data that controls the format of the printing. See also [print data set](#).

**print job**

One or more documents submitted in the same job to be printed on the same printer.

**print labeling**

A controlled method of placing identification labels on each page of PSF printed output.

**print position**

Any location on a medium where a character can be printed.

**print quality**

The measure of the quality of printed output relative to existing standards and in comparison with jobs printed previously.

**Print Services Facility (PSF)**

An IBM licensed program that manages and controls the input data stream and output data stream required by supported page printers.

**processor**

In a computer, the part that interprets and processes instructions. Two typical components of a processor are a control unit and an arithmetic logic unit.

**programming interface for customers**

Any product method that lets a customer-written program obtain the services of the product (for example, CSECT names, data areas or control blocks, data sets or files, exits, macros, parameter lists, and programming languages). Not all products have programming interfaces for customers; some products provide their services through graphical user interfaces, while others provide their services only to other products. See also [graphical user interface](#).

**programming request for price quotation (PRPQ)**

A customer request for a price quotation on alterations or additions to the functional capabilities of system control programming or licensed programs. The PRPQ can be used in conjunction with computing system RPQs to solve unique data processing problems. See also [computing system RPQ](#).

**program temporary fix (PTF)**

For System i®, System p, and IBM Z products, a package containing individual or multiple fixes that is made available to all licensed customers. A PTF resolves defects and might provide enhancements.

**proportionally spaced font**

A font in which the character increment for each graphic character varies. Proportionally spaced fonts provide the appearance of even spacing between presented characters and eliminate excess blank space around narrow characters, such as the letter i. See [mixed-pitch font](#) and [monospaced font](#).

**protocol**

A set of rules controlling the communication and transfer of data between two or more devices or systems in a communications network.

**PRPQ**

See [programming request for price quotation](#).

**PSF**

See [Print Services Facility](#).

**PSF direct**

A function of an AIX or Windows print server that enables another PSF program to print remotely.

**PTF**

See [program temporary fix](#).

**PTOCA**

See [Presentation Test Object Content Architecture](#).

**R****RACF**

See [Resource Access Control Facility](#).

**raster font**

A font in which the characters are defined directly by the raster bit map. See also [outline font](#).

**raster pattern**

A series of picture elements (pels) arranged in scan lines to form an image. The toned or untoned status of each pel creates an image. A digitized raster pattern is an array of bits. The on or off status of each bit determines the toned or untoned status of each pel.

**RAT**

See [resource access table](#).

**RDW**

See [record descriptor word](#).

**record descriptor**

Specifications that describe how record format line data records are formatted into individual print lines. Record descriptors are interpreted by PSF when formatting printed output.

**record descriptor word (RDW)**

Data preceding a variable record or a structured field that specifies the length of the entire record including the RDW.

**record format line data**

A form of line data where each record is preceded by a 10-byte identifier. See also [line data](#).

**region area**

Storage above the 16 MB line and below the 2 GB bar. By default, inline resources are stored in the region area during job processing. See also [above the bar storage](#).

**repositioning**

A process in which PSF, following an indication from the printer of a potentially recoverable error, locates the proper spool record for recomposing one or more pages for printing.

**request for price quotation (RPQ)**

A customer request for a price quotation on alterations or additions to the functional capabilities of a hardware product for a computing system or a device. See [computing system RPQ](#) and [programming request for price quotation](#).

**resident resource**

A resource, such as a font, symbol set, page segment, or overlay, that resides in a printer or an intermediary device, such as a personal computer.

**resolution**

A measure of the sharpness of an image, expressed as the number of lines per unit of length or the number of points per unit of area discernible in that image.

**resource**

A collection of printing instructions used, in addition to the print data set, to produce the printed output. Resources include coded fonts, font character sets, code pages, page segments, overlays, form definitions, and page definitions.

**Resource Access Control Facility (RACF)**

An IBM licensed program that provides for access control by identifying users to the system, verifying users of the system, authorizing access to protected resources, logging unauthorized attempts to enter the system, and logging accesses to protected resources.

**resource access table (RAT)**

An array of data that is used to map a resource name specified in the MO:DCA data stream to information used to find and process the resource on a given system.

**resource name**

The name under which an AFP resource object is stored, the first 2 characters of which indicate the resource type.

**resource object**

In AFP, 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 (or file) of a library and can be called for by PSF when needed. The different resource objects include: coded font, font character set, code page, page segment, overlay, form definition, and page definition. See also [library member](#).

**rotation**

The number of degrees a graphic character is turned relative to the page coordinates. See [character rotation](#). See also [orientation](#).

**routine**

1. A set of statements in a program that causes the system to perform an operation or a series of related operations.
2. A program or sequence of instructions called by a program. Typically, a routine has a general purpose and is frequently used.

**RPQ**

See [request for price quotation](#).

**rule**

A solid or patterned line of any weight (line width) that extends horizontally across a row or page, or vertically down a column or page.

**S****Scalable Vector Graphics (SVG)**

A vector graphics format that produces graphics that cannot be produced using AFP GOCA.

**segment**

A collection of composed text and images, prepared before formatting and included in a document when it is printed. See also [page segment](#).

**sequence number**

A 2-byte field in the structured field introducer that identifies the position of the structured field in the data set.

**sequential data set**

A data set whose records are organized on the basis of their successive physical positions, such as on magnetic tape. See also [partitioned data set](#).

**server**

A software program or a computer that provides services to other software programs or other computers. The program or computer making the request of the server is typically called the client. See also [client](#) and [host](#).

**service program**

See [utility program](#).

**sheet**

A division of the physical medium; multiple sheets can exist on a physical medium. For example, a roll of paper might be divided by a printer into rectangular pieces of paper, each representing a sheet. Envelopes are an example of a physical medium that comprises only one sheet. The IPDS architecture defines four types of sheets: cut-sheets, continuous forms, envelopes, and computer output on microfilm. Each type of sheet has a top edge. A sheet has two sides, a front side and a back side. See also [form](#).

**shift-out, shift-in (SOSI)**

Special EBCDIC or ASCII characters that exist in the data stream to indicate the switches between double-byte fonts and single-byte fonts.

**simplex**

Pertaining to printing on only one side of the paper. See also [duplex](#), [normal duplex](#), and [tumble duplex](#).

**single-byte coded font**

A font in which the characters are defined by a 1-byte code point. A single-byte coded font has only one coded font section. See also [double-byte coded font](#).

**skip**

1. A move of the current print position to another location.



2. To ignore one or more instructions in a sequence of instructions.
3. To pass over one or more positions on a data medium; for example, to perform one or more line feed operations.

**SMF**

See [System Management Facilities](#).

**SNA**

See [Systems Network Architecture](#).

**SNA-attached**

Pertaining to a device that is linked to the host system through VTAM® or ACF/VTAM and uses an SNA protocol to transfer data. It does not need to be physically connected to the host; some printers are attached to a control unit, a communication controller, or both, and they can transfer data over telecommunication lines. For example, an IBM 3825 Page Printer attached to a communication controller that uses the LU 6.2 communication protocol to transfer data to a communication controller is considered an SNA-attached printer. See also [channel-attached](#) and [TCP/IP-attached](#).

**softcopy**

One or more files that can be electronically distributed, manipulated, and printed by a user. See also [hardcopy](#).

**soft resource**

A resource that is not declared in a Map structured field but is sent to the printer inline with data. It cannot be reused during the job without being reloaded to the printer. See also [hard resource](#).

**SOSI**

See [shift-out, shift-in](#).

**stacker**

An enclosure in a printer in which printed media is stacked.

**startup procedure**

A program used to start an application and to specify initialization parameters, libraries that contain system resources, and routing-control information.

**storage**

1. A functional unit in which data can be placed and retained, and from which it can be retrieved. See also [virtual storage](#).
2. The location of saved information.

**structured field**

1. A self-identifying string of bytes and its data or parameters.
2. A mechanism that permits variable length data to be encoded for transmission in the data stream.

**subgroup**

A set of modifications in 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 traditional line data can be placed. In the page definition, multiple subpages can be placed on a physical page as specified in the print data.

**suppression**

A method used to prevent presentation of specified data. In AFP support, a page- and form-definition function that is used to identify fields in a print record that are not printed on selected pages of a document. See also [text suppression](#).

**SVG**

See [Scalable Vector Graphics](#).

**symbol set**

A type of font that resides in a printer but has fewer attributes than can be specified for resident coded fonts. See also [character set](#).

**SYSIN**

See [system input stream](#).

**SYSOUT**

See [system output stream](#).

**system input stream (SYSIN)**

A data definition (DD) statement used to begin an in-stream data set. See also [system output stream](#).

**system library**

A collection of data sets or files in which one or more system resources are stored. See also [user library](#).

**System Management Facilities (SMF)**

A component of z/OS that collects and records a variety of system and job-related information. Examples of information collected by SMF are statistics, accounting information, and performance data.

**system output stream (SYSOUT)**

A data definition (DD) statement used to identify a data set as a system output data set. See also [system input stream](#).

**system path library**

A path or set of paths for system UNIX files that contain font objects. See also [user path library](#).

**Systems Network Architecture (SNA)**

The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through the networks and controlling the configuration and operation of networks. The layered structure of SNA allows the ultimate origins and destinations of information (the users) to be independent of and unaffected by the specific SNA network services and facilities that are used for information exchange.

**T****table reference character (TRC)**

A numeric character corresponding to the order in which font character sets have been specified. The TRC is used to select a font character set during printing.

**TCP/IP-attached**

Pertaining to a device that is linked to an operating system through an Internet Protocol network and receives data from the system by using an application-layer protocol for IPDS printers. Some TCP/IP-attached printers require the i-data 7913 IPDS Printer LAN Attachment. See also [channel-attached](#) and [SNA-attached](#).

**text**

A sequence of characters that can be read by a person and encoded into formats such as ASCII that can be interpreted by a computer.

**text control**

Structured field data that control the format, placement, and appearance of text.

**text control sequence**

A text control and its associated data.

**text orientation**

A description of the appearance of text as a combination of print direction and character rotation.

**text suppression**

The intentional omission of portions of text in copy groups specified in the form definition.

**throughput**

1. The measure of the amount of work performed by a device, such as a computer or printer, over a period of time, for example, the number of jobs per day.
2. In data communications, the total traffic between stations over a period of time.

**trace**

1. A record of the processing of a computer program or transaction. The information collected from a trace can be used to assess problems and performance.
2. A Db2® for z/OS facility that provides the ability to collect monitoring, auditing, performance, accounting, statistics, and serviceability (global) data.

**traditional line data**

A form of line data that is prepared for printing on a line printer. See also [line data](#).

**transmission**

The sending of data from one place for reception elsewhere.

**tray**

See [bin](#).

**TRC**

See [table reference character](#).

**TrueType font**

A font format based on scalable outline technology in which the graphic character shapes are based on quadratic curves. The font is described with a set of tables contained in a TrueType font file.

**tumble duplex**

Pertaining to printing on both sides of the paper such that the top of one side is at the same end as the bottom of the other side. Tumble duplex printing is used for forms that are bound on the short edge of the paper, regardless of whether the printing is portrait or landscape. See also [duplex](#), [normal duplex](#), and [simplex](#).

**typeface**

All characters of a single type family or style, weight class, width class, and posture, regardless of size. An example is Helvetica bold condensed italic, in any point size. See also [font](#).

**type size**

A measurement in pitch or points of the height and width of a graphic character in a font. For example, the vertical height (point size) of a given typeface, such as 10 point.

**typographic font**

See [proportionally spaced font](#).

**U****UCS**

See [universal character set](#).

**unbounded-box font**

A font designed to use unbounded-character boxes. See also [bounded-box font](#).

**unbounded-character box**

A character box that can have blank space on any sides of the character shape. See also [bounded-character box](#).

**unformatted print records**

Traditional line data made up of fields of data that have not been formatted into print lines. PSF uses a page definition to format these records for printing on page printers.

**Unicode**

A character encoding standard that supports the interchange, processing, and display of text that is written in the common languages around the world, plus some classical and historical texts. For example, the text name for \$ is *dollar sign* and its numeric value is X'0024'. The Unicode standard has a 16-bit character set defined by ISO 10646.

**uniformly spaced font**

A font in which the character increment for each graphic character is the same. See also [monospaced font](#) and [proportionally spaced font](#).

**universal character set (UCS)**

A printer feature that permits the use of a variety of character arrays. See [font](#).

**UNIX file**

An object that exists in a hierarchical file system. Examples of UNIX files are a DFSMS Hierarchical File System (HFS), a Network File System (NFS), a temporary file system (TFS), and the z/OS File System (zFS).

**UPA**

See [user printable area](#).

**user library**

A private print-resource library owned by an individual user, accessed only when the name is specified by the owner in a JCL statement.

**user path library**

A private font library owned by an individual user, accessed only when the path name is specified by the owner in a JCL statement.

**user printable area (UPA)**

The area within the valid printable area (VPA) where user-generated data can print without causing an exception condition. See also [valid printable area](#).

**utility program**

A computer program in general support of computer processes; for example, a diagnostic program, a trace program, or a sort program.

**V****valid printable area (VPA)**

The intersection of the current logical page or current overlay with the physical page in which printing is allowed. See also [user printable area](#).

**value**

In programming, the alphabetic or numeric contents of a variable, parameter, special register, field, or storage location.

**virtual storage**

The storage space that can be regarded as addressable main storage by the user of a computer system in which virtual addresses are mapped into real addresses. The size of virtual storage is limited by the addressing scheme of the computer system and by the amount of auxiliary storage available, not by the actual number of main storage locations. See also [storage](#).

**Virtual Telecommunications Access Method (VTAM)**

An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

**VPA**

See [valid printable area](#).

**VTAM**

See [Virtual Telecommunications Access Method](#).

**X****XML data**

Data identified with the Extensible Markup Language (XML), which is a standard metalanguage for defining markup languages that is based on Standard Generalized Markup Language (SGML). For printing on page printers, a page definition is required to provide the data placement and presentation information. The XML data processed by PSF can be encoded in EBCDIC, ASCII, UTF-8 or UTF-16.

**Z****zFS**

See [z/OS File System](#).

**z/OS**

An IBM mainframe operating system that uses 64-bit real storage.

**z/OS File System (zFS)**

A type of file system that resides in a Virtual Storage Access Method (VSAM) linear data set (LDS). zFS contains files and directories that can be used by z/OS UNIX System Services to provide data access over IP networks.

**z/OS Font Collection**

A base element of z/OS V2R2 or later that contains a comprehensive set of fonts, including AFP outline fonts, AFP raster fonts, and WorldType fonts (TrueType and OpenType fonts). The recommended source of AFP fonts for printing with PSF.



## Bibliography

This bibliography lists the titles of publications containing additional information about PSF, AFP, the z/OS operating system, and related products.

The titles and order numbers might change from time to time. To verify the current title or order number, consult your IBM marketing representative.

You can obtain many of the publications listed in this bibliography from the AFP Consortium Publications ([afpcinc.org/publications](http://afpcinc.org/publications)), the IBM Documentation ([www.ibm.com/docs/en](http://www.ibm.com/docs/en)), and the IBM Publications Center ([www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss](http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss)).

## Advanced Function Presentation (AFP)

The AFP publications in Table 53 on page 277 are available from [AFP Consortium Publications](http://afpcinc.org/publications) ([afpcinc.org/publications](http://afpcinc.org/publications)).

Table 53. AFP publications available from the AFP Consortium	
Publication	Order Number
<i>Advanced Function Presentation: Programming Guide and Line Data Reference</i>	S544-3884
<i>AFP Consortium: AFP Color Management Architecture (ACMA)</i>	AFPC
<i>Bar Code Object Content Architecture Reference</i>	AFPC-0005
<i>Color Management Object Content Architecture Reference</i>	AFPC-0006
<i>Font Object Content Architecture Reference</i>	AFPC-0007
<i>Graphics Object Content Architecture for AFP Reference</i>	AFPC-0008
<i>Guide to Advanced Function Presentation</i>	G544-3876
<i>IBM AFP Fonts: Font Summary for AFP Font Collection</i>	S544-5633
<i>IBM AFP Fonts: Type Transformer User's Guide</i>	G544-3796
<i>IBM Infoprint Fonts: Font Summary</i>	G544-5846
<i>IBM Infoprint Fonts: Introduction to Type Transformer and Utilities for Windows</i>	G544-5853
<i>Image Object Content Architecture Reference</i>	AFPC-0003
<i>Intelligent Printer Data Stream Reference</i>	AFPC-0001
<i>Mixed Object Document Content Architecture Reference</i>	AFPC-0004
<i>Overlay Generation Language/370 User's Guide and Reference</i>	S544-3702
<i>Presentation Text Object Content Architecture Reference</i>	AFPC-0009
<i>Using OpenType Fonts in an AFP System</i>	G544-5876

The AFP publications in Table 54 on page 277 are available from the IBM Documentation ([www.ibm.com/docs/en](http://www.ibm.com/docs/en)) and the IBM Publications Center ([www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss](http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss)). For best results, search the IBM Publications Center by order number.

Table 54. AFP publications available from the IBM Documentation and the IBM Publications Center	
Publication	Order Number
<i>z/OS Font Collection</i>	GA32-1048

The AFP publications in Table 55 on page 278 are available from the IBM Publications Center ([www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss](http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss)). For best results, search the IBM Publications Center by order number.

Table 55. AFP publications available from the IBM Publications Center	
Publication	Order Number
<i>AFP Application Programming Interface: Programming Guide and Reference</i>	S544-3872
<i>AFP Toolbox User's Guide</i>	S544-5292
<i>Guide to Advanced Function Presentation</i>	G544-3876
<i>IBM AFP Fonts: Font Summary for AFP Font Collection</i>	S544-5633
<i>IBM AFP Fonts: Type Transformer User's Guide</i>	G544-3796
<i>IBM Infoprint Fonts: Font Summary</i>	G544-5846
<i>IBM Infoprint Fonts: Introduction to Type Transformer and Utilities for Windows</i>	G544-5853

## Text Processing

The AFP publications in Table 56 on page 278 are available from the IBM Publications Center ([www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss](http://www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss)). For best results, search the IBM Publications Center by order number.

Table 56. Text processing publications available from the IBM Publications Center	
Publication	Order number
<i>Document Composition Facility: Bar Code User's Guide</i>	S544-3115
<i>Document Composition Facility: SCRIPT/VS Text Programmer's Guide</i>	SH35-0069
<i>Document Composition Facility: SCRIPT/VS User's Guide</i>	S544-3191
<i>Publishing Systems BookMaster User's Guide</i>	SC34-5009



---

# Index

## Numerics

2-up printing, page definitions for [186](#), [190](#)  
240- and 300-pel raster fonts [20](#)  
31-bit storage [49](#)  
64-bit storage  
    storing inline resources in [144](#)

## A

above-bar storage  
    storing inline resources in [144](#)  
absolute colorimetric [230](#)  
access method, PSF as [9](#)  
accessibility  
    contact IBM [243](#)  
    features [243](#)  
ACIF, using [80](#)  
ADDRESS parameter [108](#)  
Advanced Function Presentation (AFP)  
    architecture [1](#)  
    color management [231](#)  
    files, uploading to z/OS [164](#)  
    fonts [19](#)  
    introducing [1](#)  
    parameters in JCL, specifying [83](#)  
    Printer Driver for Windows [78](#)  
    printer, selecting [111](#)  
    printing  
        direction and character rotation [3](#)  
        how PSF manages [7](#)  
        overview of [2](#)  
    resource installer [241](#)  
    resource management [239](#)  
    sending data to microfilm devices [83](#)  
    statistics about a print file, obtaining [167](#)  
    structured fields in line data [74](#)  
    using JCL for [81](#)  
AFP Conversion and Indexing Facility, using [80](#)  
AFP Reblocking Program [161](#)  
AFP Toolbox, using [78](#)  
AFPPARMS parameter [144](#)  
AFPSTATS  
    parameter [85](#)  
    report  
        hardcopy sample [216](#)  
        hardcopy, sections and generating [171](#)  
        requesting [167](#)  
        softcopy record details [197](#)  
        softcopy sample [214](#)  
        softcopy, format and records [168](#)  
    repository [167](#)  
AFRREBLK  
    command [164](#)  
    profile, setting up [162](#)  
all-points addressability [5](#)  
alternative paper source, selecting [53](#)

APSRCF30 [20](#)  
APSRMARK utility [48](#)  
architecture, AFP [1](#)  
assistive technologies [243](#)  
audit processing mode, CMR [236](#)

## B

Bar Code Object Content Architecture  
    definition of [1](#)  
    resources [39](#)  
bar codes  
    description of [39](#)  
    prefix for [41](#)  
    printing [41](#)  
    producing with  
        BCOCA [41](#)  
        DCF [41](#)  
        fonts [24](#)  
        resources, using [41](#)  
        symbolologies for [39](#)  
baseline direction for text [3](#)  
basic N\_UP printing [56](#)  
BCOCA  
    definition of [1](#)  
    resources [39](#)  
bin selection [96](#)  
BIN subcommand, specifying [61](#)  
bins  
    selecting different output [62](#)  
    specifying [117](#)  
blocking data-check errors [90](#)  
BookMaster, creating MO:DCA-P data with [78](#)  
bounded-box fonts [20](#)  
box formatting errors [152](#)  
BTS feature  
    example of using [139](#)  
    parameter for [86](#)  
BUILDING parameter [108](#)  
BURST parameter [86](#)  
burster-trimmer-stacker feature  
    example of using [139](#)  
    parameter for [86](#)  
bursting output [139](#)

## C

calibration, color [228](#)  
capturing fonts [25](#)  
carriage control characters, parameter for [89](#)  
carriage control characters, specifying [132](#)  
character rotation for text [3](#), [64](#)  
CHARS  
    parameter [86](#)  
    using with SOSI process [135](#)  
checkpoints  
    pages between [87](#)

- checkpoints (*continued*)
  - seconds between [87](#)
  - specifying [153](#)
- CKPTPAGE parameter [87](#)
- CKPTSEC parameter [87](#)
- CLASS parameter [87](#)
- CMOCA [1](#)
- CMYK color space [226](#)
- code pages
  - description of [20](#)
  - extended [141](#)
- coded fonts, description of [20](#)
- color conversion CMRs [233](#)
- Color Management Object Content Architecture [1](#)
- color management resources (CMRs)
  - audit processing mode [236](#)
  - color conversion [233](#)
  - comparison to other AFP resources [232](#)
  - creating [237](#)
  - description of [28](#)
  - device link (DL) [233](#)
  - generic halftone [234](#)
  - generic tone transfer curve [235](#)
  - halftone [234](#)
  - indexed [235](#)
  - installing [237](#)
  - instruction processing mode [236](#)
  - link (LK) [233](#)
  - link color conversion [233](#)
  - link processing mode [236](#)
  - passthrough [233](#)
  - processing modes [236](#)
  - specifying in object container libraries [150](#)
  - tone transfer curve [235](#)
  - types [232](#)
- color mapping tables (CMTs)
  - creating [156](#)
  - description of [155](#)
  - inline [150](#)
  - parameter for [88](#)
  - resources
    - examples for using [149](#)
    - from user libraries [149](#)
    - in object containers [28](#)
    - included inline [150](#)
  - sample source file [157](#)
  - source groups in [155](#)
  - target groups in [156](#)
  - using [155](#)
- Color Mapping Tool [156](#)
- color printing
  - concepts [226](#)
  - file size [229](#)
  - solutions [225](#)
- color spaces
  - color mapping tables [155](#)
  - color printing [226](#)
  - grayscale printing [229](#)
- COLORMAP parameter [88](#)
- column formatting errors [152](#)
- command, AFRREBLK [164](#)
- communication between PSF and printers [9](#)
- complex text [141](#)
- components of PSF printing [7](#)

- composite characters [5](#)
- COMSETUP parameter [88](#)
- conditional processing [67](#)
- constant forms, specifying [57](#)
- contact
  - z/OS [243](#)
- CONTROL parameter [89](#)
- controls
  - for printing with PSF [81](#)
  - in copy groups [52](#)
  - in page formats [63](#)
  - subgroups containing [61](#)
- conversion program, font [20](#)
- conversion, measurement unit [6](#)
- copies
  - defining the number of [116](#)
  - defining the number of transmissions for [114](#)
  - large number for each page [90](#)
  - printing multiple [137](#)
  - small number for each page [89](#)
  - specified in copy group subgroups [61](#)
- COPIES parameter
  - group values [89](#)
  - syntax for [89](#)
  - with FORMDEF parameter [114](#)
  - without FORMDEF parameter [116](#)
- copy groups
  - description of [51](#)
  - multiple, using [128](#)
  - printing controls specified in [52](#)
  - selecting in form definitions [52](#)
  - using internal [51](#)
- COPYCNT parameter [90](#)
- creating FOCA fonts [24](#)

## D

- data
  - formatting with PSF [10](#)
  - line [71](#)
  - MO:DCA-P [78](#)
  - printing different types of [71](#)
  - XML [76](#)
- data maps [63](#)
- data object resources
  - search order for [14](#)
  - using [27](#)
- data objects
  - creating [239](#)
  - in color printing [237](#)
  - installing [239](#)
- data page labeling, parameter for [92](#)
- data sets
  - reblocking with AFRREBLK [164](#)
  - specifying error disposition for [102](#)
  - transmissions, defining number of [114](#)
  - transmitting to IBM i systems [147](#)
- data streams
  - definition of [2](#)
  - types of PSF [7](#)
- data suppression, specified in
  - copy group subgroups [61](#)
  - page definitions [126](#)
- data-check errors, blocking [90](#)

- DATAACK parameter [90](#)
- DBCS simulation fonts [20](#)
- DCF, creating MO:DCA-P data with [78](#)
- DD statement
  - assigning OUTPUT statements to [83](#)
  - parameters used on OUTPUT statement [83](#)
- defaults
  - determining printer [81](#)
  - form, blank [221](#)
  - form, example [81](#)
- deferred-printing mode [9](#)
- DEPT parameter [108](#)
- DEST parameter [91](#)
- device link (DL) CMRs [233](#)
- diagnosing
  - incorrect printer output [173](#)
  - print file problems [167](#)
- direct-printing mode
  - creating JCL for [82](#)
  - PSF as access method [9](#)
- direction specified in page definitions, print [64](#)
- direction-rotation combinations of text [3](#)
- disabled mechanisms, printers that support [53](#)
- distributing output, parameters for [108](#)
- Document Composition Facility, creating MO:DCA-P data with [78](#)
- dot gain [228](#)
- double-byte fonts
  - description of [20](#)
  - with single-byte fonts in data [135](#)
- DPAGELBL parameter [92](#)
- DPF, storing resources with [48](#)
- DUMMY value in
  - FORMDEF [113](#)
  - PAGEDEF [122](#)
- DUPLEX parameter [93](#)
- duplex printing
  - example of specifying [117](#)
  - specifying in copy groups [54](#)
- duplex-page offsets
  - example of specifying [147](#)
  - specifying in copy groups [56](#)

## E

- electronic forms, using overlays for [30](#)
- encapsulated PostScript resources
  - description of [28](#)
  - using [27](#)
- encoding schemes for line data [73](#)
- enhanced N\_UP printing [56](#)
- EPS resources
  - description of [28](#)
  - using [27](#)
- errors
  - blocking data-check [90](#)
  - fidelity resolution [152](#)
  - inhibiting recovery of [146](#)
  - printer output, correcting [173](#)
  - printing messages for [139](#)
  - specifying data set disposition for [102](#)
- examples, printing [111](#)
- extended code pages
  - description of [20](#)

- extended code pages (*continued*)
  - using [141](#)

## F

- FCB parameter [93](#)
- feedback xxi
- fidelity resolution problems [152](#)
- fields, page format options for formatting [66](#)
- file size, color printing [229](#)
- files, uploading AFP [164](#)
- finishing output
  - examples of [151](#)
  - form definition for [182](#)
  - specifying in copy groups [59](#)
- FLASH parameter
  - syntax [93](#)
  - used with default form definition [116](#)
- FOCA [1](#)
- FOCA fonts
  - creating [24](#)
  - description of [20](#)
  - obtaining [23](#)
  - referencing [23](#)
- font character sets, description of [20](#)
- font conversion program [20](#)
- Font Object Content Architecture [1](#)
- font path library objects [22](#)
- fonts
  - 240-pel raster [20](#)
  - 300-pel raster [20](#)
  - AFP [19](#)
  - bar codes produced with [24](#)
  - bounded-box format [20](#)
  - capturing [25](#)
  - character rotation [3](#)
  - conversion program for [20](#)
  - creating FOCA [24](#)
  - DBCS simulation [20](#)
  - description of [19](#)
  - direction [3](#)
  - double-byte [20](#)
  - examples of specifying [126](#)
  - FOCA [20](#)
  - inline [86](#)
  - library-member types [20](#)
  - line spacing for PSF monospaced [194](#)
  - mapping tables for [24](#)
  - microfilm device [223](#)
  - monospaced, line spacing for [194](#)
  - OpenType [22](#)
  - outline [20](#)
  - parameter for specifying [86](#)
  - printer-resident [24](#)
  - printing at different resolutions [152](#)
  - process mode, specifying for [135](#)
  - raster [20](#)
  - resident in the printer [24](#)
  - rotation [3](#)
  - selected in page formats [67](#)
  - selecting for MO:DCA-P data [79](#)
  - selecting with table reference characters [133](#)
  - single-byte [20](#)
  - SOSI codes, using to change [74](#)

- fonts (*continued*)
  - specifying in page formats [67](#)
  - TrueType [22](#)
  - unbounded-box format [20](#)
  - using
    - to produce bar codes [24](#)
    - with different printer resolutions [24](#)
- form controls in copy groups [52](#)
- form definitions
  - copy group printing controls specified in [52](#)
  - creating with PPFA [51](#)
  - description of [25](#)
  - DUMMY, specifying [113](#)
  - inline [113](#)
  - microfilm device [223](#)
  - modified-default [116](#)
  - names for
    - 3-hole punched paper [178](#)
    - 3800 printers [176](#)
    - compatibility with continuous-forms printers [180](#)
    - compatibility with cut-sheet printers [181](#)
    - finishing output [182](#)
    - N\_UP 2 printing [178](#)
    - N\_UP compatibility with continuous-forms printers [180](#)
    - non-3800 printers [177](#)
    - PCL4 and PPDS printers [179](#)
    - print quality on 64xx and 65xx printers [179](#)
    - printers other than 3800, PCL4, and PPDS [176](#)
    - printing envelopes on 4028 printers [177](#)
    - printing PSF reports [183](#)
    - rotating pages on paper [178](#)
    - special purpose jobs [181](#)
    - use with OGL [182](#)
  - naming convention [175](#)
  - overview of [52](#)
  - page overlay rotation in [36](#)
  - parameter for [94](#)
  - prefix for [26](#)
  - resources
    - from user libraries [113](#)
    - included inline [113](#)
  - selecting copy groups in [52](#)
  - specifying [112](#)
  - supplied with PSF [175](#)
- form, defaults
  - blank [221](#)
  - example [81](#)
- format resolution
  - order of PSF search [152](#)
  - specification errors [152](#)
- formatted print records, page format options for [65](#)
- formatting
  - changing within documents [128](#)
  - data with PSF [10](#)
  - fields with page format options [66](#)
  - line data [63](#)
  - MO:DCA-P data [79](#)
  - options for formatted print records [65](#)
- FORMDEF parameter [94](#)
- FORMLEN parameter [95](#)
- forms control buffers
  - converted to page definitions
    - supplied with PSF [69](#)
- forms control buffers (*continued*)
  - converted to page definitions (*continued*)
    - using [123](#)
  - parameter [93](#)
- forms flash
  - parameter for [93](#)
  - specified without FORMDEF parameter [116](#)
  - subgroup modification [62](#)
  - using [118](#)
- FORMS parameter [95](#)
- forms source, selecting [53](#)
- forms, specifying [96](#)

## G

- gamut
  - colors in [228](#)
  - definition of [227](#)
- generic halftone CMRs [234](#)
- GOCA
  - definition of [1](#)
  - pattern fill source group [155](#)
  - resources [42](#)
- graphics
  - description of [42](#)
  - prefix for [43](#)
  - printing [43](#)
  - resources, using [43](#)
- Graphics Object Content Architecture
  - definition of [1](#)
  - pattern fill source group [155](#)
  - resources [42](#)
- grayscale printing
  - concepts [229](#)
  - solutions [225](#)
- group values in COPIES parameter [89](#)

## H

- halftones
  - CMR [234](#)
  - grayscale printing [230](#)
  - overview [228](#)
- hard page segment [39](#)
- hard resources [15](#)
- hardcopy report, AFPSTATS [171](#)
- header page, sample [108](#)
- highlight color source group [155](#)
- horizontal adjustment value, specifying [61](#)
- horizontal page offsets [52](#)
- host resources [13](#)

## I

- IBM i system, transmitting data sets to [147](#)
- ICC [230](#)
- ICC profiles
  - color printing [226](#)
  - grayscale printing [229](#)
  - overview [230](#)
- IM image data object, printing with [45](#)
- Image Object Content Architecture
  - definition of [1](#)

## Image Object Content Architecture (*continued*)

- resources [43](#)
- image resources
  - description of [43](#)
  - printing
    - color tips [240](#)
    - with IOCA [45](#)
    - without IOCA [45](#)
  - using [27](#), [44](#)
- incorrect printer output [173](#)
- incorrect-character errors, blocking [90](#)
- indexed CMRs [235](#)
- inhibiting print job recovery [146](#)
- inline direction for text [3](#)
- inline resources
  - color mapping table [150](#)
  - fonts [86](#)
  - form definitions [113](#)
  - microfilm setup [148](#)
  - page definitions [122](#)
  - printing with [143](#)
  - storing [13](#)
  - storing above bar [144](#)
  - using [49](#)
- input paper source, selecting [53](#)
- instruction processing mode, CMR [236](#)
- internal copy groups [51](#)
- International Color Consortium [230](#)
- INTRAY parameter [96](#)
- IOCA
  - definition of [1](#)
  - resources [43](#)
- IOCA tile resource objects
  - description of [28](#)
  - using [27](#)

## J

### JCL

- default printer values [81](#)
  - defaults form
    - blank [221](#)
    - example [81](#)
  - for Color Mapping Tool, sample [157](#)
  - for direct-printing mode [82](#)
  - for microfilm devices [83](#)
  - OUTPUT statement assigned to DD statement [83](#)
  - parameters
    - for microfilm jobs [224](#)
    - for printing MO:DCA-P data [79](#)
    - ignored with XML data [78](#)
    - specifying for microfilm jobs [148](#)
  - PRINTDEV statement [82](#)
  - using for AFP [81](#)
- ### JCL parameters
- ADDRESS [108](#)
  - AFPPARMS [144](#)
  - AFPSTATS [85](#)
  - BUILDING [108](#)
  - BURST [86](#)
  - CHARS [86](#)
  - CKPTPAGE [87](#)
  - CKPTSEC [87](#)
  - CLASS [87](#)

## JCL parameters (*continued*)

- COLORMAP [88](#)
- COMSETUP [88](#)
- CONTROL [89](#)
- COPIES [89](#)
- COPYCNT [90](#)
- DATAACK [90](#)
- DEPT [108](#)
- DEST [91](#)
- DPAGELBL [92](#)
- DUPLEX [93](#)
- FCB [93](#)
- FLASH [93](#)
- FORMDEF [94](#)
- FORMLEN [95](#)
- FORMS [95](#)
- INTRAY [96](#)
- LINECT [96](#)
- NAME [108](#)
- NOTIFY [97](#)
- OFFSETXB [97](#)
- OFFSETXF [98](#)
- OFFSEYB [98](#)
- OFFSEYF [99](#)
- on OUTPUT or DD statement [83](#)
- OUTBIN [100](#)
- OVERLAYB [100](#)
- OVERLAYF [100](#)
- PAGEDEF [100](#)
- PIMSG [101](#)
- PRMODE [102](#)
- PRTERORR [102](#)
- PRTQUEUE [103](#)
- RESFMT [103](#)
- ROOM [108](#)
- SEGMENT [104](#)
- SUBSYS [104](#)
- SYSAREA [105](#)
- SYSOUT [105](#)
- TITLE [108](#)
- TRC [106](#)
- UCS [106](#)
- USERLIB [107](#)
- USERPATH [107](#)
- job segments, specifying [104](#)

## K

### keyboard

- navigation [243](#)
- PF keys [243](#)
- shortcut keys [243](#)

## L

### landscape page presentation [3](#)

### libraries

- multiple system page segment [39](#)
- storing resource [13](#)
- types of resource [15](#)
- library objects, font path [22](#)
- library-member types for FOCA fonts [20](#)
- line data

- line data (*continued*)
  - definition of [71](#)
  - encoding schemes for [73](#)
  - examples of printing [121](#)
  - formatting with page definitions [63](#)
  - line spacing for [124](#)
  - merging lines [135](#)
  - printing [71](#)
  - record format [73](#)
  - selecting fonts [126](#)
  - SOSI codes in [74](#)
  - structured fields in [74](#)
  - traditional [71](#)
  - using carriage control characters in [132](#)
  - using table reference characters in [132](#), [133](#)
- line merging applications, creating [135](#)
- line printer, description of [3](#)
- line screen frequency, halftone [228](#)
- line spacing for
  - line and XML data, specifying [124](#)
  - monospaced fonts [194](#)
  - pages [65](#)
  - PSF-supplied page definitions [192](#)
- line spacing parameter [89](#)
- LINECT parameter [96](#)
- link (LK) CMRs [233](#)
- link color conversion CMRs [233](#)
- link processing mode, CMR [236](#)
- logical page
  - definition of [64](#)
  - origin [32](#), [52](#)

## M

- mapping tables
  - color [28](#)
  - font [24](#)
- margin errors [152](#)
- marking resources [48](#)
- measurement unit conversion [6](#)
- media-relative colorimetric [230](#)
- medium maps
  - description of [51](#)
  - multiple, using [128](#)
  - printing controls specified in [52](#)
  - selecting in form definitions [52](#)
  - using internal [51](#)
- medium origin [52](#)
- medium overlays
  - positioning [31](#)
  - printing [119](#)
  - using [33](#)
- merging
  - data with overlay [34](#)
  - lines [135](#)
- messages, printing
  - example of specifying [139](#)
  - parameter for [101](#)
- messages, sending notification [146](#)
- microfilm devices
  - considerations for [223](#)
  - creating JCL for [83](#)
- microfilm setup resources
  - examples for specifying [148](#)

- microfilm setup resources (*continued*)
  - from user libraries [148](#)
  - in object containers [29](#)
  - included inline [148](#)
  - parameter for [88](#)
- mixing structured fields with line data [74](#)
- mixing, color [228](#)
- MO:DCA-P data
  - creating with programs [78](#)
  - definition of [1](#)
  - formatting [79](#)
  - JCL parameters for printing [79](#)
  - object containers [1](#)
  - OCAs in [1](#)
  - printing
    - example of [131](#)
    - overview [78](#)
    - selecting fonts for [79](#)
- monospaced fonts, line spacing for [194](#)
- multiple
  - copies
    - parameter for large [90](#)
    - parameter for small [89](#)
    - printing [137](#)
    - specified in copy group subgroups [61](#)
    - specified in JCL [116](#)
  - copy groups, using [128](#)
  - page formats, using [128](#)
  - pages on a sheet, printing [125](#)
  - resolution printers, printing on [152](#)
  - system page segment libraries [39](#)
- multiple-image and multiple-page resources [29](#)
- multiple-up printing
  - conditional processing for [68](#)
  - example of [124](#)
  - page definitions for [186](#), [190](#)

## N

- N\_UP printing
  - basic [56](#)
  - enhanced [56](#)
  - example of [125](#)
  - specifying in copy groups [56](#)
- NAME parameter [108](#)
- naming convention for form definitions [175](#)
- navigation
  - keyboard [243](#)
- normal duplex printing, specifying [54](#)
- notification messages, sending [146](#)
- NOTIFY parameter [97](#)
- number of copies
  - parameter for large [90](#)
  - parameter for small [89](#)
  - specified in copy group subgroups [61](#)
  - specified in JCL [116](#)
  - transmitted [114](#)

## O

- object container resources
  - color management resources [28](#)
  - color mapping table resources [28](#)

- object container resources (*continued*)
  - description of [26](#)
  - encapsulated PostScript (EPS) resources [28](#)
  - IOCA tile resource objects [28](#)
  - microfilm setup resources [29](#)
  - multiple-image and multiple-page resources [29](#)
  - PDF resource objects [30](#)
  - PDF single-page object resources [29](#)
  - resident color profile resource objects [30](#)
  - scalable vector graphics resource objects [30](#)
  - specifying in libraries [150](#)
  - SVG resource objects [30](#)
  - using [27](#)
- object containers, definition of [1](#)
- Object Content Architecture (OCA)
  - BCOCA resources [39](#)
  - description of [1](#)
  - GOCA resources [42](#)
  - IOCA resources [43](#)
  - PTOCA resources [46](#)
  - source group, standard [155](#)
- objects, TrueType and OpenType [22](#)
- obtaining AFP statistics [167](#)
- offset stacking
  - continuous forms [139](#)
  - specified in copy groups [60](#)
- offsets
  - duplex-page [56](#), [147](#)
  - horizontal and vertical page [52](#)
- OFFSETXB parameter [97](#)
- OFFSETXF parameter [98](#)
- OFFSETYB parameter [98](#)
- OFFSETYF parameter [99](#)
- OpenType fonts
  - obtaining [23](#)
  - referencing [23](#)
  - using [140](#)
- origin, medium or page [52](#)
- OUTBIN
  - parameter [100](#)
  - subcommand or keyword [62](#)
- outline fonts [20](#)
- output
  - bins [62](#)
  - bursting [139](#)
  - finishing [59](#)
  - offset stacking of [60](#)
  - parameters for distributing [108](#)
  - stacking [139](#)
  - writer, PSF as [9](#)
- OUTPUT statement
  - AFPPARMS parameter [144](#)
  - assigned to DD statement [83](#)
  - parameters used on DD statement [83](#)
- overlapping characters [5](#)
- Overlay Generation Language (OGL)
  - creating overlays with [30](#)
  - form definition for [182](#)
  - using with marked resources [48](#)
- OVERLAYB parameter [100](#)
- OVERLAYF parameter [100](#)
- overlays
  - description of [30](#)
  - identified in copy group subgroups [61](#)

- overlays (*continued*)
  - medium
    - example of printing [119](#)
    - positioning [31](#)
    - using [33](#)
  - merging data with [34](#)
  - page
    - example of printing [119](#)
    - positioning [32](#), [120](#)
    - rotation of [36](#)
    - using [33](#)
  - prefix for [33](#)
  - preprinted form overlays [33](#)
  - printing medium and page on same page [35](#)
  - printing with [118](#)
  - rotation of page [36](#)
  - specifying [33](#)
  - testing [35](#)
  - using with different printer resolutions [35](#)

## P

- page definitions
  - converted from FCBs [123](#)
  - data suppression in [126](#)
  - description of [36](#)
  - DUMMY, specifying [122](#)
  - fonts, specifying in [126](#)
  - formatting data with [63](#)
  - inline, using [122](#)
  - names for
    - 10.12 x 14.33 B4 cut-sheet paper [188](#), [189](#)
    - 12 x 8.5 continuous-forms paper [187](#)
    - 12 x 8.5 cut-sheet paper [185](#)
    - 12 x 8.5 or 9.5 x 11 letter and continuous-forms paper [190](#)
    - 14.88 x 11 continuous-forms paper [188](#)
    - 14.88 x 11 cut-sheet paper [186](#)
    - 14.88 x 11 legal and continuous-forms paper [190](#)
    - 14.88 x 11 paper and FCBs [185](#)
    - 3-hole punched paper [191](#)
    - 3800 printers [185](#)
    - 4224, 4230, 4234, 4247, and 6400 printers [187](#)
    - 8.27 x 11.69 A4 cut-sheet paper [188](#), [189](#)
    - 8.5 x 11 cut-sheet letter paper [188](#)
    - 8.5 x 14 cut-sheet legal paper [189](#)
    - 9.5 x 11 continuous-forms paper [187](#)
    - 9.5 x 11 cut-sheet paper [186](#)
    - HP-CL4 and PPDS printers [188](#)
    - line spacing [192](#)
    - multiple-up printing [186](#), [190](#)
    - other printers [189](#)
    - printing PSF reports [192](#)
  - overview of [63](#)
  - page formats in [63](#)
  - page overlay rotation in [36](#)
  - parameter for [100](#)
  - prefix for [37](#)
  - relative print line positioning [65](#)
  - resources
    - converted from FCBs [123](#)
    - from user libraries [122](#)
    - included inline [122](#)
  - specifying [121](#)



- page definitions (*continued*)
  - supplied with PSF [185](#)
  - using PPFA to create [51](#)
- page formats
  - controls for
    - conditional processing [67](#)
    - fonts [67](#)
    - formatted print records [65](#)
    - formatting fields [66](#)
    - line spacing for a page [65](#)
    - page size [64](#)
    - print direction [64](#)
    - relative print line positioning [65](#)
  - description of [63](#)
  - multiple, using [128](#)
  - required information in [63](#)
  - type of data specified in [63](#)
- page offsets [52](#), [56](#)
- page origin [52](#)
- page overlays
  - positioning [32](#), [120](#)
  - printing [119](#)
  - rotation in page and form definitions [36](#)
  - using [33](#)
- page position [52](#)
- page presentation
  - controls for [59](#)
  - direction [3](#)
- page printer
  - description of [3](#)
  - selecting [111](#)
- Page Printer Formatting Aid, using [51](#)
- page segments
  - description of [37](#)
  - hard [39](#)
  - libraries, using multiple system [39](#)
  - prefix for [38](#)
  - printing [130](#)
  - soft [39](#)
  - testing [39](#)
  - using [38](#)
- page size specified in page definitions [64](#)
- page-printer defaults form [81](#), [221](#)
- PAGEDEF parameter [100](#)
- paper characteristics for color [231](#)
- paper destination, selecting [62](#)
- paper source
  - changing in a document [118](#)
  - example of specifying [117](#)
  - specifying in copy groups [53](#)
  - specifying in subgroups [61](#)
- paper tray, selecting [96](#)
- parameters, JCL [83](#)
- passthrough CMRs [233](#)
- pattern fill, GOCA [155](#)
- pattern, halftone [228](#)
- PCS [230](#)
- PDF resource objects
  - description of [30](#)
  - using [27](#)
- PDF single-page object resources
  - description of [29](#)
  - using [27](#)
- pels
  - (*continued*)
    - density and placement [6](#)
    - printing at different resolutions [152](#)
  - perceptual [230](#)
  - PIMSG parameter [101](#)
  - portrait page presentation [3](#)
  - PPFA, using [51](#)
  - predesigned forms, using
    - forms-flash unit [118](#)
    - OGI [30](#)
  - prefixes for resources [15](#)
  - preprinted form overlays
    - using [33](#)
  - Presentation Text Object Content Architecture
    - definition of [1](#)
    - resources [46](#)
  - print direction
    - for text [3](#), [64](#)
    - specifying [123](#)
  - print line positioning, relative [65](#)
  - print quality, selecting levels [60](#)
  - Print Services Facility (PSF)
    - access method [9](#)
    - APSRMARK utility [48](#)
    - combining different types of data [48](#)
    - communication with printers [9](#)
    - data streams [7](#)
    - deferred-printing mode [9](#)
    - direct-printing mode [9](#)
    - distributed print function (DPF) [48](#)
    - form definitions provided with [175](#)
    - formatting data with [10](#)
    - introduction to [7](#)
    - libraries [15](#)
    - managing AFP printing [7](#)
    - messages, printing [101](#)
    - output writer [9](#)
    - page definitions provided with [185](#)
    - printer defaults for [81](#)
    - printing components [7](#)
    - products [7](#)
    - resources
      - overview of [13](#)
      - supplied with [50](#)
  - print-complete messages, sending [146](#)
  - print-positioning errors, blocking [90](#)
  - PRINTDEV statement [82](#)
  - printer
    - calibration for color [228](#)
    - checkpoints, specifying [153](#)
    - defaults form, page [81](#), [221](#)
    - defaults, determining [81](#)
    - driver
      - for Windows, AFP [78](#)
      - PSF [2](#)
    - output, incorrect [173](#)
    - resident fonts [24](#)
    - resolutions, using FOCA fonts with different [24](#)
    - selection
      - example of [111](#)
      - parameter for [87](#)
    - support for disabled mechanisms [53](#)
- printers
  - communication with PSF [9](#)



- printers (*continued*)
  - line, description of [3](#)
  - page
    - description of [3](#)
    - selecting [111](#)
- printing
  - AFP [2](#)
  - bar codes [41](#)
  - color concepts [226](#)
  - components of PSF [7](#)
  - controls specified in copy groups [52](#)
  - data [71](#)
  - deferred mode [9](#)
  - direct mode
    - creating JCL for [82](#)
    - PSF in [9](#)
  - duplex [54](#)
  - error messages [101](#), [139](#)
  - examples [111](#)
  - graphics [43](#)
  - grayscale concepts [229](#)
  - how PSF manages AFP [7](#)
  - images [45](#)
  - inhibiting recovery of [146](#)
  - job segments [104](#)
  - line data [71](#), [121](#)
  - messages [101](#), [139](#)
  - mixed data [74](#)
  - MO:DCA-P data
    - example of [131](#)
    - overview [78](#)
  - multiple copies [137](#)
  - multiple pages on a sheet [125](#)
  - multiple-up [124](#)
  - N\_UP
    - example of [125](#)
    - specifying in copy groups [56](#)
  - notification sent when finished [146](#)
  - on microfilm devices [223](#)
  - on multiple-resolution printers [152](#)
  - overlays [30](#), [35](#), [118](#)
  - page segments [130](#)
  - performance problems, diagnosing [167](#)
  - record format line data [73](#)
  - resources [13](#)
  - simplex [54](#)
  - solutions, color and grayscale [225](#)
  - tasks [111](#)
  - text [47](#)
  - with inline resources [143](#)
  - with user library resources [142](#)
  - XML data [76](#), [121](#)
- PRMODE parameter [102](#)
- process mode
  - parameter [102](#)
  - specifying SOSI codes with [135](#)
- processing modes, CMR
  - audit [236](#)
  - instruction [236](#)
  - link [236](#)
- profile connection space [230](#)
- profile, AFRREBLK [162](#)
- PRERROR parameter [102](#)
- PRTQUEUE parameter [103](#)

- PTOCA
  - definition of [1](#)
  - resources [46](#)
- publications, related [277](#)

## Q

- Qualified Tag [76](#)
- quality levels, selecting print [60](#)

## R

- raster fonts [20](#)
- RATs [239](#)
- reblocking
  - data sets [164](#)
  - program for AFP files [161](#)
- record format line data
  - definition of [73](#)
  - encoding schemes for [73](#)
- recovery of print jobs, inhibiting [146](#)
- related publications [277](#)
- relative print line positioning [65](#)
- rendering intent [230](#)
- rendering intents
  - definition of [227](#)
- report, AFPSTATS [167](#)
- repository, AFPSTATS [167](#)
- reprocessing [69](#)
- RESFMT parameter [103](#)
- resident
  - fonts, printer [24](#)
  - resources [13](#)
- resident color profile resource objects
  - description of [30](#)
  - using [27](#)
- resolutions
  - parameter for specifying output [103](#)
  - printing at different [152](#)
  - using FOCA fonts with different [24](#)
  - using overlays on printers with different [35](#)
- resource access tables [239](#)
- resource installer, AFP [241](#)
- resource management, color [239](#)
- resources
  - AFP fonts [19](#)
  - bar codes [39](#)
  - color mapping table, using [149](#)
  - color printing tips [240](#)
  - definition of [13](#)
  - FOCA fonts [20](#)
  - fonts [19](#)
  - form definitions [25](#)
  - graphics (GOCA) [42](#)
  - hard [15](#)
  - host [13](#)
  - image (IOCA) [43](#)
  - inline
    - printing with [143](#)
    - storing above bar [144](#)
    - using [49](#)
  - installing [239](#)
  - libraries [15](#)

- resources (*continued*)
  - marking [48](#)
  - object containers [26](#)
  - OpenType fonts [22](#)
  - overlays [30](#)
  - page definitions [36](#)
  - page segments [37](#)
  - prefixes for [15](#)
  - printing at different resolutions [152](#)
  - problems, diagnosing [167](#)
  - resident [13](#)
  - searching for in print jobs [14](#)
  - soft [15](#)
  - statistics about [167](#)
  - storing [13](#)
  - storing with DPF [48](#)
  - supplied with PSF [50](#)
  - text (PTOCA) [46](#)
  - TrueType fonts [22](#)
  - types of PSF [8](#)
  - user library, printing [142](#)
- RGB color space [226](#)
- ROOM parameter [108](#)
- rotation of page overlays [36](#)
- rotation, halftone [228](#)
- rotations, character [3](#), [64](#)

## S

- saturation [230](#)
- scalable vector graphics resource objects
  - description of [30](#)
- search for specified resources [14](#)
- security labeling, parameter for [92](#)
- SEGMENT parameter [104](#)
- selecting
  - fonts [126](#)
  - output bins [62](#)
  - paper destination [62](#)
  - paper source [53](#)
  - paper tray [96](#)
  - print quality level [60](#)
  - printers [111](#)
- sending to IBM
  - reader comments [xxi](#)
- separator pages
  - parameters for [108](#)
  - sample [108](#)
- shift-out, shift-in (SOSI) codes
  - in line data [74](#)
  - parameter for specifying [102](#)
  - specifying [135](#)
  - using CHARS parameter with [135](#)
- shortcut keys [243](#)
- simplex printing, specifying [54](#)
- single-byte fonts
  - description of [20](#)
  - with double-byte fonts in data [135](#)
- size specified in page definitions, page [64](#)
- soft page segment [39](#)
- soft resources [15](#)
- softcopy report, AFPSTATS [168](#)
- source file, sample of color mapping table [157](#)
- source groups in color mapping tables [155](#)

- spacing, parameter for line [89](#)
- stacking output [139](#)
- stacking, offset
  - continuous forms [139](#)
  - specified in copy groups [60](#)
- standard OCA source group [155](#)
- stapling, example of [151](#)
- statistics, obtaining AFP [167](#)
- storage
  - above-bar
    - storing inline resources in [144](#)
  - region area [49](#)
- storing PSF resources [13](#)
- structured fields for
  - changing copy groups or page formats [128](#)
  - identifying image resources [43](#)
  - including page segments [38](#)
  - line data [74](#)
  - producing bar codes [41](#)
  - rotating page overlays [36](#)
  - specifying offset stacking [60](#)
- subgroup modifications
  - data suppression [61](#)
  - forms flash [62](#)
  - number of page copies [61](#)
  - output bins [62](#)
  - overlays [61](#)
  - paper source [61](#)
- subgroups in copy groups [61](#)
- subpages, placement of [186](#), [191](#)
- SUBSYS parameter [104](#)
- summary of changes [xxiii](#)
- suppression, data
  - specified in copy group subgroups [61](#)
  - specified in page definitions [126](#)
- SVG resource objects
  - description of [30](#)
- symbolologies for bar codes [39](#)
- syntax for JCL parameters [83](#)
- SYSAREA parameter [105](#)
- SYSOUT parameter [105](#)

## T

- table reference characters
  - parameter for [106](#)
  - rules for coding [134](#)
  - selecting fonts with [126](#), [133](#)
  - specifying [106](#), [132](#)
  - using in line data [133](#)
- target groups in color mapping tables [156](#)
- tasks, printing [111](#)
- testing
  - overlays [35](#)
  - page segments [39](#)
- text
  - description of [46](#)
  - printing [47](#)
  - resources, using [47](#)
- text direction and rotation [3](#), [64](#)
- TITLE parameter [108](#)
- tone transfer curves
  - CMR [235](#)
  - grayscale printing [230](#)

- tone transfer curves (*continued*)
  - overview [228](#)
- tools, application development
  - ACIF [80](#)
  - AFP Toolbox [78](#)
- traditional line data
  - definition of [71](#)
  - encoding schemes for [73](#)
  - multiple-up printing [124](#)
  - using table reference characters in [132](#), [133](#)
- transmissions, data set
  - defining number of [114](#)
  - to IBM i systems [147](#)
- tray selection [96](#)
- TRC parameter [106](#)
- troubleshooting printer output [173](#)
- TrueType fonts
  - obtaining [23](#)
  - referencing [23](#)
  - using [140](#)
- tumble duplex printing, specifying [54](#)
- Type Transformer [24](#)

## U

- UCS parameter [106](#)
- unblocking data-check errors [90](#)
- unbounded-box fonts [20](#)
- unformatted print records, field formatting options for [66](#)
- Unicode complex text [141](#)
- unit conversion [6](#)
- uploading AFP files to z/OS [164](#)
- user interface
  - ISPF [243](#)
  - TSO/E [243](#)
- user library resources
  - color mapping table [149](#)
  - form definitions [113](#)
  - microfilm setup [148](#)
  - page definitions [122](#)
  - printing with [142](#)
- USERLIB parameter [107](#)
- USERPATH parameter [107](#)

## V

- vertical page offsets [52](#)

## W

- Windows AFP Printer Driver [78](#)

## X

- XML data
  - element content [77](#)
  - encoding schemes and conversions [77](#)
  - examples of printing [121](#)
  - external entities [77](#)
  - formatting with page definitions [63](#)
  - JCL parameters ignored [78](#)
  - line spacing for [124](#)
  - printing [76](#)







Product Number: 5655-M32

S550-0435-06

