Advanced Function Presentation

IBM

Programming Guide and Line Data Reference

Advanced Function Presentation

Programming Guide and Line Data Reference

Preface

This book is a reference for printing line-mode and mixed-mode data in an AFP environment. It describes the presentation of line-mode data streams using the Page Definition (PageDef) print control object. Line-mode data streams that adhere to the specifications in this document are accepted for printing by IBM Print Services Facility (PSF) products in most system environments. These products include PSF for OS/390, PSF/VM, PSF/VSE, Infoprint Manager for AIX, Infoprint Manager for Windows NT and Windows 2000, and the AFP capability of AS/400.

This book also defines structured fields and objects that are used exclusively for processing line data. Because many of these objects are either defined directly by the Mixed Object Document Content Architecture (MO:DCA) or based on MO:DCA definitions, readers should also familiarize themselves with the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Portions of the content in this book relating to color management were developed by the current members (the "Members") of the AFP Color Consortium™ ("AFPCC™"). The AFPCC began in 2004 with a goal of adding color management support to AFP. It is an industry-wide collaboration whose members include both AFP application providers and AFP printer manufacturers. A list of the current Members of the AFPCC can be found at *http://www.afpcolor.org*. The Members have entered into a separate agreement by which they have identified those Members who have made contributions to, and therefore have ownership rights in, such portions of the content. Such Members shall be referred to as "Contributing Members."

This book is intended for programmers who write applications that generate line-mode and mixed-mode data streams, or data stream objects, for printing across IBM system environments.

Contents

Figures

Tables

Chapter 1. Introduction

Programmers can develop applications for Advanced Function Presentation™ (AFP™) hardware and software, generating either traditional unformatted line data, fully composed Mixed Object Document Content Architecture™ (MO:DCA™) data (also called AFP data), or a combination of both. This book contains examples and suggestions for writing such applications. IBM's AFP print servers support MO:DCA data streams in the following environments:

- Advanced Interactive Executive (AIX®)
- Application System/400® (AS/400®), iSeries™, and System i5

  Note: Unless otherwise stated, references to AS/400 also apply to iSeries and System i5™.
- Operating System/2® (OS/2®)
-
- System/390® (S/390®) and zSeries® environments, including:
  - OS/390® and z/OS®
  - Virtual Machine (VM) and z/VM®
  - Virtual Storage Extended (VSE) and z/VSE™

  Note: Unless otherwise stated, references to S/390 also apply to the corresponding zSeries environments.
- Microsoft® Windows®

The print data streams can include text, images, graphics, and bar code in MO:DCA format. MO:DCA defines the data stream used by applications to describe documents and object envelopes for interchange with other applications and application services. Documents defined in the MO:DCA format may be archived in a data base, then later retrieved, viewed, and printed in local or distributed systems environments.

Print Services Facility™ (PSF) software in the S/390 environments accepts data in traditional line-printer format and generates page-mode output from the line data, using information contained in a Page Definition (PageDef) resource object. The line data mapped by the Page Definition may or may not include additional MO:DCA structured fields. A file that includes a combination of line data and MO:DCA structured fields is called a *mixed mode* file. Only certain MO:DCA structured fields can be intermixed with line data. Detailed information on coding those structured fields appears in Chapter 4, "Mixed Documents: Adding MO:DCA Structured Fields to Line Data," on page 43.

PSF in the AIX and Windows environments accept non-MO:DCA data streams that can be formatted using Page Definition resource objects. These data streams can be in any of the following formats:

- Traditional line printer format, also called 1403 format
- Unformatted ASCII files without escape sequences
- DBCS (double-byte character set) ASCII files generated for an IBM® 5577 or 5587

PSF in the AS/400 environment accepts line data or mixed data, either created on an S/390 platform and networked to AS/400 or created natively on AS/400. Such data is placed on the printer spool using a *Printer File*, which may also specify the Page Definition and Form Definition (Formdef) to be used for formatting and printing the data.

1

Related Architectures

Mixed-mode data streams can include line data, MO:DCA structured fields, and objects of the following types:
- Bar Code Object Content Architecture™ (BCOCA™)
- Color Management Object Content Architecture™ (CMOCA™)
- Font Object Content Architecture (FOCA)
- Graphics Object Content Architecture (GOCA)
- Image Object Content Architecture (IOCA)
- Presentation Text Object Content Architecture (PTOCA)
- Non-OCA paginated presentation objects such as Encapsulated PostScript (EPS)

A related architecture, but not a user programming language, is the Intelligent Printer Data Stream™ (IPDS™) architecture. This is the data stream architecture used by print server products to manage IPDS printers.

System Model

AFP print servers provide support for interpreting line data, mixed-mode data, and MO:DCA data, for resolving resource references, and for building printer data streams for driving IPDS printers. Figure 1 shows the general relationship between the AFP data streams, the print server products, and IPDS printers.
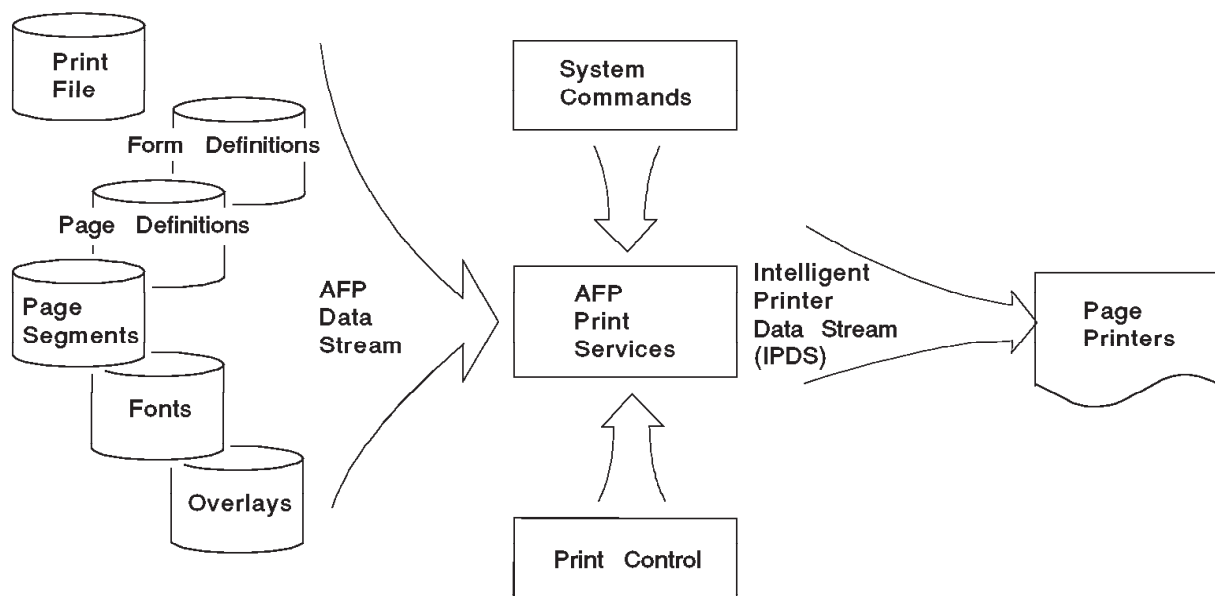
Figure 1. AFP System Printing Relationships

Each print server product has its own books that describe how to submit print jobs in its system environment. See these books for information on setting up jobs for printing. Refer to *Advanced Function Presentation Printer Information*, G544-3290, for individual characteristics and presentation considerations for various supported printers.

Supported Environments

IBM systems provide common printer support and print services in the following environments:
- AIX
- AS/400 and System i5
- OS/2
- OS/390 and z/OS
- VM and z/VM
- VSE and z/VSE
- Windows

Appendix D, "System Support Information," on page 183 describes the specific AFP functions available in each system environment.

Chapter 2. Line Data and MO:DCA (AFP) Data

The Advanced Function Presentation (AFP) products have been developed to be consistent with a set of architectures that define the format of documents and the nature of the commands sent from the host software to the supported printers. The Mixed Object Document Content Architecture (MO:DCA) defines a device-independent data stream format for interchanging documents among AFP products. Data to be printed can include text, graphics, images, and bar codes.

The *objects* used for Advanced Function Presentation include:

- Font objects, which consist of *font character sets* containing the patterns for letters, numbers, and special characters, and *code pages* which associate a hexadecimal value with each character in the font character set
- Resource objects, including overlays and page segments, which in turn can include text, graphics, image, and bar code
- Print control objects, which include Page Definitions used to format line data and Form Definitions used to control physical aspects of the print environment.

These objects can exist in resource libraries external to the data to be printed, or can be included inline with the data that will use them.

AFP objects can be obtained in a number of different ways. For example, a wide variety of fonts are available from IBM and other sources, and Page Definitions, Form Definitions, and overlays can be built using any of several tools available from IBM and other AFP vendors.

In addition, MO:DCA documents can be generated using the Document Composition Facility (DCF), the AFP Print Driver for Windows and OS/2, the AFP Toolbox, and other document composition products. In line data environments, users can add MO:DCA structured fields directly to their line data. Chapter 4, "Mixed Documents: Adding MO:DCA Structured Fields to Line Data" provides information on how to do this.

Print Services Facility (PSF) software, available on S/390 (OS/390,VM, and VSE), AS/400, OS/2, AIX, and Windows systems, accepts MO:DCA documents and resources and in turn generates Intelligent Printer Data Stream (IPDS) commands to drive the printers it controls. PSF can also accept other forms of data as input. One of the most widely used of these is called *line data*.

Line Data

Line data, meaning application output to be printed that is not already in MO:DCA format, is supported by PSF and formatted by Page Definition resource objects in all system environments except OS/2. The nature of line data is slightly different in the system environments where it appears.

S/390 Environments

System/390 applications written in programming languages such as Assembler, COBOL, FORTRAN, PL/I, RPG, or others have historically produced output files to be printed on line-mode printers such as the 1403, 3211, or 3800-1. These line

data files consist of individual print records, each of which corresponds to one line of data on an impact printer. The application program either formats line data records or leaves them unformatted.

Formatted line data records contain information exactly as it will be printed, because line printers have little or no capability of formatting print output records. Unformatted line data records contain only the fields of data to be printed. With unformatted line data records, the data is not formatted into lines, columns, paragraphs or other structures that determine how the records will appear on paper. AFP print server products support printing of both formatted and unformatted line data records using the Page Definition (also called PageDef) resource object.

Figure 2 illustrates the difference between formatted and unformatted line data records.
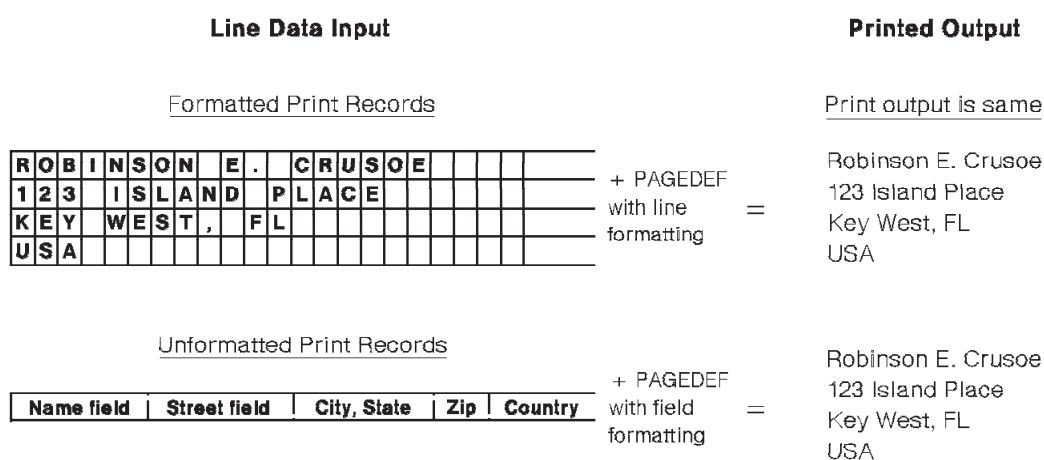
Figure 2. Formatted and Unformatted Line Data Records

Traditional impact printers allowed only one variation on the line-by-line format of output. The first character of the line, preceding the actual data characters, could optionally be a *carriage control byte*. This byte indicated whether the data line should be printed using single, double or triple spacing, whether spacing should be suppressed (creating an overstrike), or whether the line should be placed at a predefined position on the page that was associated with a value of 1 through 12, known as the *channel code*. This page position value was defined in an auxiliary object called a forms control buffer, or FCB. The FCB defined the number of lines per page, whether lines were spaced at 6 or 8 per inch, and which line, if any, was associated with the position values of 1 through 12. The 3800 Model 1 added a spacing value of 12 lines per inch to the FCB; and the 3800 Model 3 added 10 lines per inch.

The carriage control character could be represented in either of two coding schemes:

- *American National Standards Institute (ANSI) carriage control* is a standard representation used with printers from many different vendors. Table 1 on page 7 lists the ANSI codes and their functions.
- *Machine code control characters* were defined by IBM. They correspond to the channel command words issued by the operating system to accomplish the desired function. Table 2 on page 7 lists the IBM machine code values and their functions.

ANSI and machine codes may not be intermixed within a single data set.

Line spacing is handled differently by ANSI and machine code carriage controls. ANSI conventions cause spacing to take place first, followed by printing the line, while with machine codes, the line is printed first, and then the spacing action is performed.

Note that if a spacing control action moves the print position past the last line of the current page, processing continues at the first print position of a new page. That is, the spacing action is not carried over to the new page.

Table 1. ANSI Carriage Control Characters

Control Character Value (in hexadecimal)	Function
X'40' (blank)	Space 1 line, then print (single spacing)
X'F0' (zero)	Space 2 lines, then print (double spacing)
X'60' (dash)	Space 3 lines, then print (triple spacing)
X'4E' (plus sign)	Suppress spacing, then print (overstrike previous line)
X'F1'	Print the data at line position defined as Channel 1 (by convention, the first line on a new page)
X'F2'	Print the data at the line position defined as Channel 2 in the FCB
X'F3'	Print the data at the line position defined as Channel 3 in the FCB
X'F4'	Print the data at the line position defined as Channel 4 in the FCB
X'F5'	Print the data at the line position defined as Channel 5 in the FCB
X'F6'	Print the data at the line position defined as Channel 6 in the FCB
X'F7'	Print the data at the line position defined as Channel 7 in the FCB
X'F8'	Print the data at the line position defined as Channel 8 in the FCB
X'F9'	Print the data at the line position defined as Channel 9 in the FCB
X'C1'	Print the data at the line position defined as Channel 10 in the FCB
X'C2'	Print the data at the line position defined as Channel 11 in the FCB
X'C3'	Print the data at the line position defined as Channel 12 in the FCB
Note: When ANSI carriage controls are used, only the values that appear in this table are considered valid by PSF. PSF treats any other ANSI carriage control value as invalid and prints any data on the line using single spacing.	

Table 2. Machine Code Control Characters

Control Character Value (in hexadecimal)	Function
X'03'	No operation
X'09'	Print and space 1 line (single spacing)
X'11'	Print and space 2 lines (double spacing)
X'19'	Print and space 3 lines (triple spacing)
X'01'	Print without spacing (overstrike next line)
X'89'	Print the data, then skip to the line position defined as Channel 1 (by convention, the first line on a new page)
X'91'	Print the data, then skip to the line position defined as Channel 2

Table 2. Machine Code Control Characters (continued)

Control Character Value (in hexadecimal)	Function
X'99'	Print the data, then skip to the line position defined as Channel 3
X'A1'	Print the data, then skip to the line position defined as Channel 4
X'A9'	Print the data, then skip to the line position defined as Channel 5
X'B1'	Print the data, then skip to the line position defined as Channel 6
X'B9'	Print the data, then skip to the line position defined as Channel 7
X'C1'	Print the data, then skip to the line position defined as Channel 8
X'C9'	Print the data, then skip to the line position defined as Channel 9
X'D1'	Print the data, then skip to the line position defined as Channel 10
X'D9'	Print the data, then skip to the line position defined as Channel 11
X'E1'	Print the data, then skip to the line position defined as Channel 12
X'0B'	Space 1 line without printing
X'13'	Space 2 lines without printing
X'1B'	Space 3 lines without printing
X'8B'	Skip to Channel 1 immediate (by convention, the first line on a new page)
X'93'	Skip to the Channel 2 FCB position immediate
X'9B'	Skip to the Channel 3 FCB position immediate
X'A3'	Skip to the Channel 4 FCB position immediate
X'AB'	Skip to the Channel 5 FCB position immediate
X'B3'	Skip to the Channel 6 FCB position immediate
X'BB'	Skip to the Channel 7 FCB position immediate
X'C3'	Skip to the Channel 8 FCB position immediate
X'CB'	Skip to the Channel 9 FCB position immediate
X'D3'	Skip to the Channel 10 FCB position immediate
X'DB'	Skip to the Channel 11 FCB position immediate
X'E3'	Skip to the Channel 12 FCB position immediate
Note: PSF ignores the following hexadecimal machine-code carriage control characters and does not print lines containing them: X'02' through X'07', X'0A', X'12', X'23', X'43', X'63', X'6B', X'73', X'7B', X'EB', X'F3', and X'FB'. PSF treats any other carriage control value as invalid and prints any data on the line using single spacing.	

One other modification to printer line data was introduced with the 3800 Model 1. This modification allows an additional byte to appear at the beginning of a line to indicate which one of up to four different character arrangement tables loaded in the printer is used to print the line. This byte, the *table reference character* (TRC), contains a value of X'F0', X'F1', X'F2', or X'F3', corresponding to the relative position of the desired character arrangement table in the list of table names specified in the data set's job control language. If carriage control bytes are used with the data, the table reference character follows the carriage control byte but precedes the data bytes. If carriage control is not used, the table reference character is the first byte of the data record. As with carriage control, if table reference

characters are used, every data record must contain a TRC byte. More information on table reference characters can be found in the application programming guides for PSF/MVS, PSF/VM, and PSF/VSE.

AIX and Windows Environments

Data in an AIX or Windows environment can be in *stream* format, with each record or line to be printed delimited by a *line separator*; or it can be in *record-based* format. In record-based format, the *line separator* is not required, as the length of each record is contained in a two-byte prefix to the record. Either of these formats is considered line data and can be mapped for printing by PSF if a Page Definition is used.

Note: The *line separator* is described in the "Line Data Summary."

AS/400 Environment

In AS/400 print environments, line data and mixed data is written to the system spool using a *Printer File*. This file may also reference the Page Definition and Form Definition to be used for formatting and printing the data. The data with Page Definition and Form Definition is processed by PSF for AS/400.

PSF/2 Environment

PSF/2 does not include Page Definition support. PSF/2 can accept data in MO:DCA format, or it can print IPDS data streams sent from another AFP system.

Table 3. Platform Support of Data formats

Platform	Record-based	Stream
S/390	X	X (Note 1)
AIX, Windows	X (Note 2)	X
AS/400	X	X (Note 1)
Notes: 1. Only supported when input data is XML. 2. Only supported when the length of each record is contained in a two byte prefix to the record or when each record is the same size.		

Line Data Summary

To print line data, PSF must know the dimensions of the page, the exact position on that page where each record must be printed, and the fonts to be used. This information is provided for line data records in an AFP resource object called the Page Definition, or PageDef. The Page Definition is described in Chapter 3, "Using a Page Definition to Print Data."

Figure 3 on page 10, and Figure 4 on page 12 summarize the valid forms of line data.

Note: In Figure 3 on page 10 and Figure 4 on page 12, the stream formats are terminated with a *line separator*. A *line separator* is normally a Line Feed character or a combined Carriage Return character and Line Feed character pair. Windows platforms typically use the Carriage Return and Line Feed pair as the *line separator*. The *line separator* code points vary based on the data encoding and platform. The supported *line separators* are:

MO:DCA and Line Data

- EBCDIC data: Line Feed (X'25').
- ASCII and UTF-8 data: Line Feed (X'0A') or Carriage Return (X'0D') and Line Feed (X'0A') pair.
- UTF-16BE: Line Feed (X'000A') or Carriage Return (X'000D') and Line Feed (X'000A') pair.
- UTF-16LE: Line Feed (X'0A00') or Carriage Return (X'0D00') and Line Feed (X'0A00') pair. Note that when the input data is UTF-16LE (little-endian byte order), the program that processes the line data needs to convert the data to big-endian byte order. Big-endian byte order is the only byte order supported in AFP environments.

		D	A	T	A		

Simple data line

CC			D	A	T	A	

Data line with carriage control byte

TRC			D	A	T	A	

Data line with table reference character

CC	TRC			D	A	T	A	

Data line with carriage control byte and table reference character

	D	A	T	A		LS

Data line in stream format with line separator

CC		D	A	T	A		LS

Data line in stream format with carriage control byte and line separator

TRC		D	A	T	A		LS

Data line in stream format with table reference character and line separator

CC	TRC		D	A	T	A		LS

Data line in stream format with carriage control byte, table reference character, and line separator

Note: The data portion and line separators of the valid records above can be encoded using Unicode Standard encoding UTF-16 or UTF-8

Figure 3. Valid Line Data Records (Part 1 of 2)

BOM	D	A	T	A

Unicode data line with Byte Order Mark

CC	BOM	D	A	T	A

Unicode data line with carriage control byte and Byte Order Mark

TRC	BOM	D	A	T	A

Unicode data line with table reference character and Byte Order Mark

CC	TRC	BOM	D	A	T	A

Unicode data line with carriage control byte, table reference character, and Byte Order Mark

BOM	D	A	T	A	LS

Unicode data line in stream format with Byte Order Mark and line separator

CC	BOM	D	A	T	A	LS

Unicode data line in stream format with carriage control byte, Byte Order Mark, and line separator

TRC	BOM	D	A	T	A	LS

Unicode data line in stream format with table reference character, Byte Order Mark, and line separator

CC	TRC	BOM	D	A	T	A	LS

Unicode data line in stream format with carriage control byte, table reference character, Byte Order Mark, and line separator

Note: For a description of the BOM (Byte Order Mark) see "Unicode Line Data" on page 12. The BOM is allowed only on the first record and applies to all records in the print file.

Figure 3. Valid Line Data Records (Part 2 of 2)

Record-Format Line Data

Another form of line data that is supported by PSF and formatted by a Page Definition resource is *record-format* line data. With this format, each line data record contains a 1 to 250-byte record identifier, which selects the Record Descriptor (RCD) in a record-format Data Map that is used to format the line data. A carriage control (CC) byte is optional but is ignored if specified; table reference characters (TRCs) are not supported. Many functions used in the Line Descriptor (LND) to format traditional line data are also used in the RCD to format record-format line data. Others, such as header and trailer processing, are unique to RCDs.

Figure 4 on page 12 summarizes the valid forms of record-format line data.

Record ID	D	A	T	A		

Record format line data

CC	Record ID	D	A	T	A	

Record format line data with carriage control byte

Record ID	D	A	T	A	LS

Record format line data in stream format with line separator

CC	Record ID	D	A	T	A	LS

Record format line data in stream format with carriage control byte and line separator

Note: The data portion and line separators of the valid records above can be encoded using Unicode Standard encodings UTF-16 or UTF-8

BOM	Record ID	D	A	T	A	

Unicode Record format line data with Byte Order Mark

CC	BOM	Record ID	D	A	T	A

Unicode Record format line data with carriage control byte and Byte Order Mark

BOM	Record ID	D	A	T	A	LS

Unicode Record format line data in stream format with Byte Order Mark and line separator

CC	BOM	Record ID	D	A	T	A	LS

Unicode Record format line data in stream format with carriage control byte, Byte Order Mark, and line separator.

Note: For a description of the BOM (Byte Order Mark) see "Unicode Line Data." The BOM is allowed only on the first record and applies to all records in the print file.

Figure 4. Valid Record-Format Line Data

Unicode Line Data

The data portion of the valid line data formats shown in Figure 3 on page 10 and in Figure 4 can be encoded using Unicode Standard encodings UTF-16 or UTF-8. The Unicode Standard recommends that a byte order mark (BOM) be the first sequence of bytes in the data. This is to accommodate platforms, such as Windows, that use the little-endian byte order. It also serves as a signature to identify Unicode text. The Byte Order Marks supported are:
- UTF-8: X'EFBBBF'
- UTF-16BE (big-endian byte order): X'FEFF'
- UTF-16LE (little-endian byte order): X'FFFE'

The Byte Order Mark is optional. If used, the BOM is only on the first line or record of line data in the print file. It is recommended that switching the encoding in the page definition be avoided. If the font selected indicates the data is UTF-16 and the BOM is not used, big-endian byte order is assumed. When the BOM is not used, switching the encoding in the page definition should not pose any problems.

Unicode encoding is subject to these restrictions in an AFP environment:

- Shift-out/Shift-in (SOSI) controls are not used in Unicode to signify a shift into and out-of DBCS processing. Therefore, it is not possible to switch processing between Unicode encoding and single-byte (SBCS) encoding within a line data field or record using SOSI as described in "Processing Line Data with Shift-Out/Shift-In (SOSI) Controls" on page 34. That is, when a line data field is processed with a Page Definition, either the whole field is treated as Unicode-encoded, or none of it is treated as Unicode-encoded.
- If the Byte Order Mark used in UTF-16 data indicates the data is in little-endian byte order, programs that process the UTF-16 data will need to convert little-endian to big-endian byte order.
- If carriage control bytes (CC) or table reference character bytes (TRC) are used with UTF-16 encoded data, the CC and TRC bytes remain 1 byte fields and are not encoded in UTF-16.

XML Data

XML data may be formatted using a Page Definition resource, however this is subject to the following restrictions:

- Carriage Control (CC) and Table Reference Characters (TRC) are not supported.
- The data is encoded using one of the following:
  - EBCDIC (Single-byte only)
  - ASCII (Single-byte only)
  - UTF-8
  - UTF-16 (including surrogates; see "Unicode Line Data" on page 12 for information about byte order)

  Application Note: When using FOCA fonts (fonts mapped with an MCF in the AEG) to process XML data, the following can occur:
  - If the data is encoded in ASCII or UTF-8 and the font specifies an encoding scheme of Unicode Presentation, the processor of the XML data may convert the data to UTF-16BE.
  - If the data is encoded in UTF-16 and the font specifies an encoding scheme of PC Data SBCS, the processor of the XML data may convert the data to ASCII. This conversion might result in unprintable code points.

- MO:DCA data cannot be mixed with XML data.

For a description of XML Data, refer to the XML specification, *Extensible Markup Language (XML) 1.0*, which can be found at the World Wide Web Consortium web site, `http://www.w3.org/`.

MO:DCA Data Summary

In contrast to line data, fully composed page data, or MO:DCA (AFP) data, contains control information such as position, orientation, and font selection intermixed with the data to be printed. PSF accepts a MO:DCA document and generates the corresponding IPDS commands needed to drive a printer. An external Page Definition resource is not needed with MO:DCA data because all the formatting information is already included in the document.

MO:DCA data is fully described in the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Combining Line Data with MO:DCA Structured Fields

It is possible to intermix line data records and MO:DCA structured fields in a single file and send the mixed data to PSF for printing. This permits the addition of image, graphics, and bar code objects to existing line data output. Applications can be written to generate line data or other objects as needed to produce the desired final print product.

Note: MO:DCA structured fields cannot be combined with XML data.

Line data and MO:DCA records cannot be mixed haphazardly. Chapter 4, "Mixed Documents: Adding MO:DCA Structured Fields to Line Data," on page 43 provides guidelines on the valid combinations.

The Function of the Page Definition

Any print file that contains line data, whether alone or in combination with MO:DCA structured fields, requires a Page Definition for printing using PSF. The Page Definition is necessary to establish the environment for each page and to position each line of print.

A number of Page Definitions mapping common page layouts are provided with the PSF software products. Users can create their own Page Definitions with any of the following software products:
* Page Printer Formatting Aid/370, IBM Program Product 5688-190
* Page Printer Formatting Aid/6000, IBM Program Product 5765-140, feature 1970
* AFP PrintSuite for AS/400, IBM Program Product 5798-AF2
* Print Management Facility/MVS, IBM Program Product 5665-307
* Print Management Facility/VM, IBM Program Product 5664-310
* Elixir Application Builder for AFP from Elixir Technologies Corporation
* ISIS OverView for AFP from Isis Information Systems, Incorporated

Chapter 3. Using a Page Definition to Print Data

One of the major enhancements provided by AFP to existing line-data applications is the ability to print application-generated output in different formats without making any application changes. This function is called *outboard formatting*, and is provided by the Page Definition resource object.

Page Definitions are supported by PSF for OS/390, PSF/VM, PSF/VSE, Infoprint® Manager for AIX, Infoprint Manager for Windows, PSF for AS/400, Infoprint Server for z/OS, and Infoprint Server for iSeries. PSF uses Page Definitions to map the line data produced by application programs. Page Definitions are *not* used when printing fully composed MO:DCA documents, as formatting information is already included internally in these documents.

Page Definition names are provided in job control statements. Any print file can be associated with a Page Definition by using the appropriate parameters in the job control statements for the applicable operating system. See the reference publications for your print server and operating system for complete information.

Common Examples of Page Definition Use

Many users want to take advantage of AFP capabilities that provide multiple-up printing or rotated printing without any need to change the application that generates the output. Line data can be printed in any desired format by creating a Page Definition that describes that format. PSF software includes many Page Definitions that address common user needs, such as printing two-up output on 8.5 by 11 inch paper.

One example of multiple-up printing is provided by IBM-supplied Page Definition W12883. This Page Definition prints two pages of 64 lines each, side by side in landscape mode on letter-sized paper. The data is printed at 8.2 lines per inch, so a 15-pitch or smaller font must be used to prevent the lines from overlapping.

Another example applies to users of continuous-forms impact printers who install a cut-sheet laser printer and begin to use letter-size sheets of paper, rather than the larger forms found on impact printers. The impact printers always printed in the ACROSS direction on paper whose width exceeded its length. But ACROSS printing on cut-sheet paper is generally considered to mean printing parallel to the narrow edge, not the wide edge. A Page Definition that prints in the DOWN direction, or in the landscape orientation, can be used to get the same result with letter-size paper on a cut-sheet printer as with larger forms on an impact printer.

Using More than One Page Definition

When a line-data file (such as a SYSOUT file produced by a System/370™ application program) is printed, only one Page Definition can be used to map the output format of that file. However, multiple copies of the file can be printed, each one using a different Page Definition, if the appropriate job control statements are used. The actual syntax varies depending on the operating system. An example for OS/390 is shown in Figure 5 on page 16. By using a job stream similar to the one shown in the figure, multiple copies of a line-mode data set can be generated, each one in a different format.

This example produces three different collated copies of the entire output file, each one formatted using a different Page Definition. The same approach can be used with Form Definitions. Each OUTPUT statement includes a different Form Definition name to invoke various options such as overlays, paper source, simplex, or duplex.

Although only one Page Definition and Form Definition can be used when printing a single file, the internal structure of Page Definitions and Form Definitions includes multiple sets of formatting rules. These sets of rules are called *Data Maps* (also called *Page Formats*) in the Page Definition, and *Medium Maps* (also called *copy groups*) in the Form Definition. The *Invoke Data Map* and *Invoke Medium Map* structured fields can be written in the output by an application program and used to switch between maps as printing proceeds. This makes it possible for subsets of the file to be presented in different formats. Examples will be provided later in this chapter and in Chapter 4, "Mixed Documents: Adding MO:DCA Structured Fields to Line Data," on page 43.

When a Page Definition containing more than one Data Map, or a Form Definition containing more than one Medium Map is used, the one which appears physically first in the resource object is selected as the default.

```
//PRINTJOB  JOB ...//STEP1     EXEC  PGM=IEBGENER
//OUT1      OUTPUT PAGEDEF=PD1
//OUT2      OUTPUT PAGEDEF=PD2
//OUT3      OUTPUT PAGEDEF=PD3
   .
   .
   .
//SYSUT2    DD  SYSOUT=A,OUTPUT=(*.OUT1,*.OUT2,*.OUT3)
```

Figure 5. Printing a Data Set in OS/390 Multiple Times with Different Page Definitions

Page Definition Structure

A Page Definition is required to compose line data into pages for printing on page printers. A Page Definition consists of one or more *Data Maps* that define the page environment and provide instructions for mapping each line of data to the page.

A Page Definition object can be referenced from a library defined to PSF or can be included inline at the beginning of a print file in some system environments. The structured fields in the Page Definition conform to the MO:DCA architecture rules for structured fields. These rules are summarized in Chapter 5, "Structured Fields in a Page Definition and in Line Data," on page 71 of this publication and are formally defined in the *Mixed Object Document Content Architecture Reference*, SC31-6802.

A Page Definition optionally can contain one or more Conditional Processing (CCP) structured fields. Conditional processing permits the application programmer to define tests on selected data fields in the input line records and to specify actions to take when the conditions of the test are satisfied. Figure 6 on page 17 shows the structure of a Page Definition.

*   = optional
s = can appear more than once
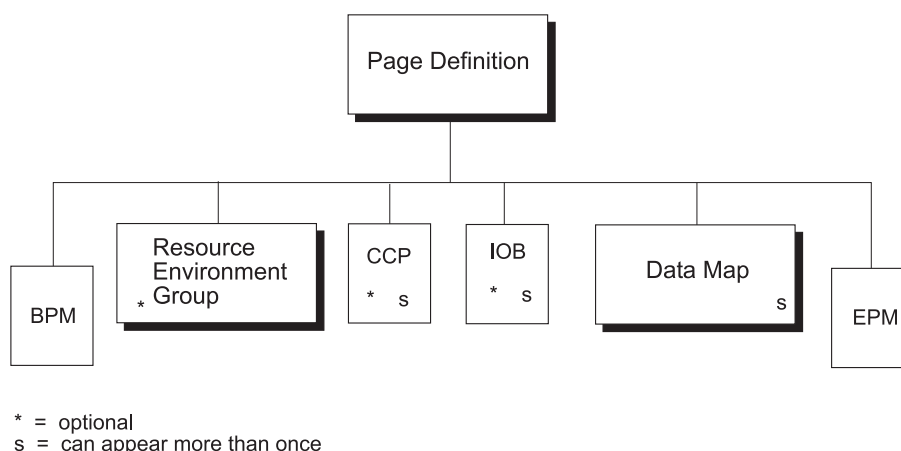
Figure 6. Page Definition Structure

The structured fields and objects that compose a Page Definition are as follows. (Chapter 5, "Structured Fields in a Page Definition and in Line Data" describes the structured fields.)

BPM (Begin Page Map)
> Begins a Page Definition resource object. An optional token name may be specified to identify the object.

Resource Environment Group
> The Resource Environment Group (REG) identifies complex resources that need to be loaded in the presentation device before the pages that follow are processed.

CCP (Conditional Processing Control)
> The CCP structured field is optional but can occur multiple times in the Page Definition. This structured field appears at the beginning of the Page Definition, outside any of the Data Map definitions, since it can be used by any Data Map to control switching among Data Maps in the Page Definition and Medium Maps in the Form Definition. The CCP defines the condition to be tested, the data value to be used to compare against the application data, the action to be taken based on the result of the test, and when the action is to be taken.

> A single CCP can make multiple tests, and Page Definitions can contain multiple conditions to form complex testing sequences. These multiple conditions are reflected in multiple CCPs.

> Each CCP in a Page Definition object has a unique identifier. The LND structured fields of a Data Map use this identifier to invoke conditional processing. Each LND using conditional processing specifies the length and position of the field in the application data record to be tested. Different LNDs can invoke the same CCP multiple times in the same Data Map definition.

> See "Conditional Processing Control (CCP)" on page 83 for details about the CCP structured field.

IOB (Include Object)
> The IOB structured field is optional but can occur multiple times in the Page Definition. The IOB appears at the beginning of the Page Definition, following the CCP structured fields. The IOB is processed when it is

referenced by an LND or RCD. The reference consists of an ID that is specified on the LND or RCD and that must match the ID on an IOB.

Data Map Object
Provides specific line definitions and mapping instructions for composing line data into a presentation page format. A single Page Definition object may specify multiple Data Maps. Different Data Maps in the Page Definition may be selected by using the Invoke Data Map structured field or by using conditional processing.

EPM (End Page Map)
Ends a Page Definition resource object. Any name specified in the EPM must match the name specified in the BPM.

Resource Environment Group

Figure 7 shows the structure of a Resource Environment Group (REG) in the Page Definition.

```
                    ┌──────────────┐
                    │ Resource     │
                    │ Environment  │
                    │ Group        │
                    └──────────────┘
         ┌──────────┬──────┬──────┬──────────┐
      ┌─────┐    ┌─────┐ ┌─────┐ ┌─────┐  ┌─────┐
      │ BSG │    │ MDR │ │ MPO │ │ PPO │  │ ESG │
      │     │    │ *  s│ │ *  s│ │ *  s│  │     │
      └─────┘    └─────┘ └─────┘ └─────┘  └─────┘
```

* = optional
s = can appear more than once
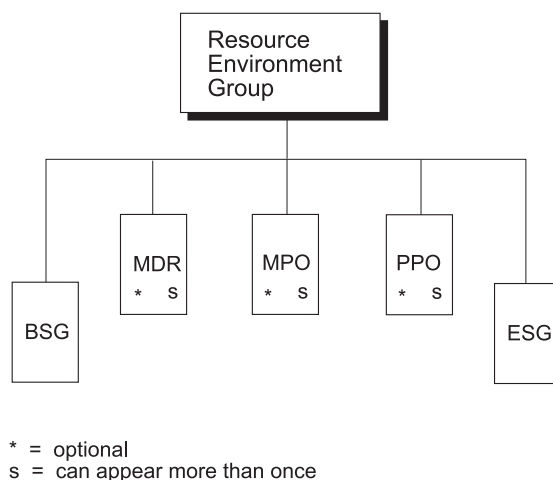
Figure 7. Resource Environment Group Structure for a Page Definition

A Resource Environment Group (REG), when specified in a Page Definition, is associated with a print file. The REG is used to identify complex resources, such as high-resolution color images, that need to be downloaded to the presentation device before the pages that follow are processed. The scope of a REG in the Page Definition is the line-format data in the print file. When a print file contains multiple line-format data and mixed data documents, the REG applies only to the line-format data documents in the print file. For a definition of line-format data, see Figure 33 on page 167. Line-format data may be bounded by explicit BDT/EDT pairs or by implicit BDT/EDT pairs.

Architecture Note: To get the optimum performance benefit from the REG in the Page Definition, the print file should contain only line-format data, and only large, complex objects should be mapped in the REG. This will allow the line-format data to be treated as a single document, and the REG will cause all mapped objects to be preloaded to the printer at the start of that document.
The REG in the Page Definition is not applied to MO:DCA-P documents in the print file. The mapping of resources in a REG is optional. Resources mapped in a REG must still be mapped in the AEG for the Data Map that uses the resources. The structured fields that compose a Resource Environment Group are as follows.

BSG (Begin Resource Environment Group)

Begins a Resource Environment Group. A token name may be specified to identify the REG.

MDR (Map Data Resource)

The MDR structured field is optional but can occur multiple times in a REG. The MDR specifies a resource that is required for presentation. The resource is identified with a file name, the identifier of a begin structured field for the resource, or any other identifier associated with the resource. The MDR may additionally specify a local or internal identifier for the resource object. Such a local identifier may be embedded one or more times within an object's data.

MPO (Map Page Overlay)

The MPO specifies overlay resources required for presentation. It is optional and can occur multiple times in a REG.

PPO (Preprocess Presentation Object)

The PPO structured field is optional but can occur multiple times in a REG. The PPO specifies presentation parameters for a data object that has been mapped as a resource. These parameters allow the presentation device to preprocess and cache the object so that it is in presentation-ready format when it is included with a subsequent include structured field in the document. Such preprocessing may involve a rasterization or RIP of the object, but is not limited to that. The resource is identified with a file name, the identifier of a begin structured field for the resource, or any other identifier associated with the resource. The referenced resource and all required secondary resources must previously have been mapped with an MDR or an MPO in this environment group.

Note: Preprocessing is not supported for objects that are included with structures that are outside the document. Examples of such objects are medium overlays and PMC overlays, both of which are included with structures in the Form Definition.

ESG (End Resource Environment Group)

Ends a Resource Environment Group.

Data Map Structure

Figure 8 shows the structure of a Data Map, also called a Page Format.
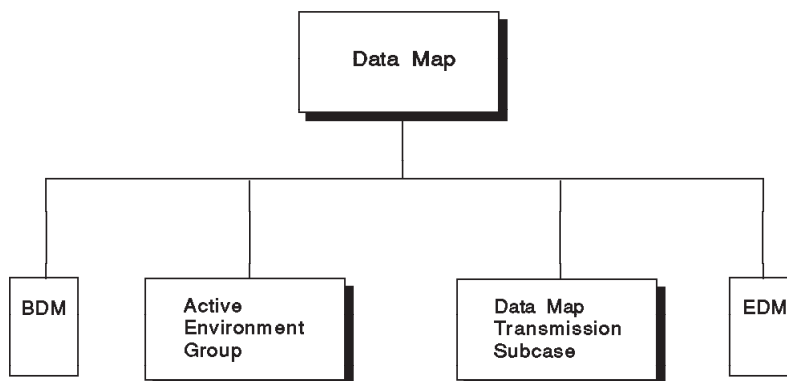
Figure 8. Data Map Structure for a Page Definition

Each Page Definition must include at least one Data Map. Structured fields in the Data Map accomplish the page layout functions similar to those provided by FCBs used with non-AFP printers, but many additional functions are available.

Each Data Map provides instructions for mapping line data to a page. The number of Data Maps that can be included in a Page Definition is limited only by practical considerations such as whether the total size of the Page Definition will be so large that it might not fit in PSF's program storage. Each Data Map in the Page Definition can contain entirely different information about how a page should appear, so different Data Maps can be used from one page to the next with output produced by a line-data application.

The Data Maps in a Page Definition can select two types of line data processing:
- Traditional line data containing optional CCs and TRCs are processed using LNDs in the Data Map Transmission Subcase.
- Record-format line data containing record IDs and optional CCs are processed using RCDs in the Data Map Transmission Subcase.

All Data Maps in the Page Definition must specify the same line data processing.

The application can select which Data Map to use by writing an Invoke Data Map structured field in the output file or by using conditional processing in the Page Definition to select a Data Map based on the value of a field in the application data stream. Examples of using an IDM can be found in Chapter 4, "Mixed Documents: Adding MO:DCA Structured Fields to Line Data," on page 43. Examples of conditional processing appear at the end of this chapter.

The Data Map consists of two parts: the Active Environment Group and the Data Map Transmission Subcase. Bracketing them are the Begin Data Map and End Data Map structured fields. The format of these structured fields is as follows:

BDM (Begin Data Map)
> Begins a Data Map. A one-to-eight character token name is required to identify the Data Map. A one-byte code indicates whether the Data Map contains LNDs or RCDs. For the latter, the BDM may contain a triplet that specifies the page margins and a triplet that specifies page count controls.

EDM (End Data Map)
> Ends a Data Map. Any name specified in the EDM must match the name specified in the BDM.

The Active Environment Group establishes the page environment, including page size, and may contain the names of resources, such as fonts and page segments, that are to be mapped. The Data Map Transmission Subcase specifies the position, orientation, color and font selection for text, the identification of data fields to be suppressed, any "fixed text" for the page, and any conditional processing tests and actions. It may also specify objects to be included on the page.

Active Environment Group Structure

Figure 9 on page 21 shows the structure of an Active Environment Group (AEG) in the Data Map.

Active Environment Group

PEC
*

MCF
*   s

MDR
*   s

MPO
* † s

MPS
*   s

PGD

OBD (text)
*

OBP (text)
*

PTD
*

BAG

EAG

* = optional
s = can appear more than once
† = required for every IPO specified in a page
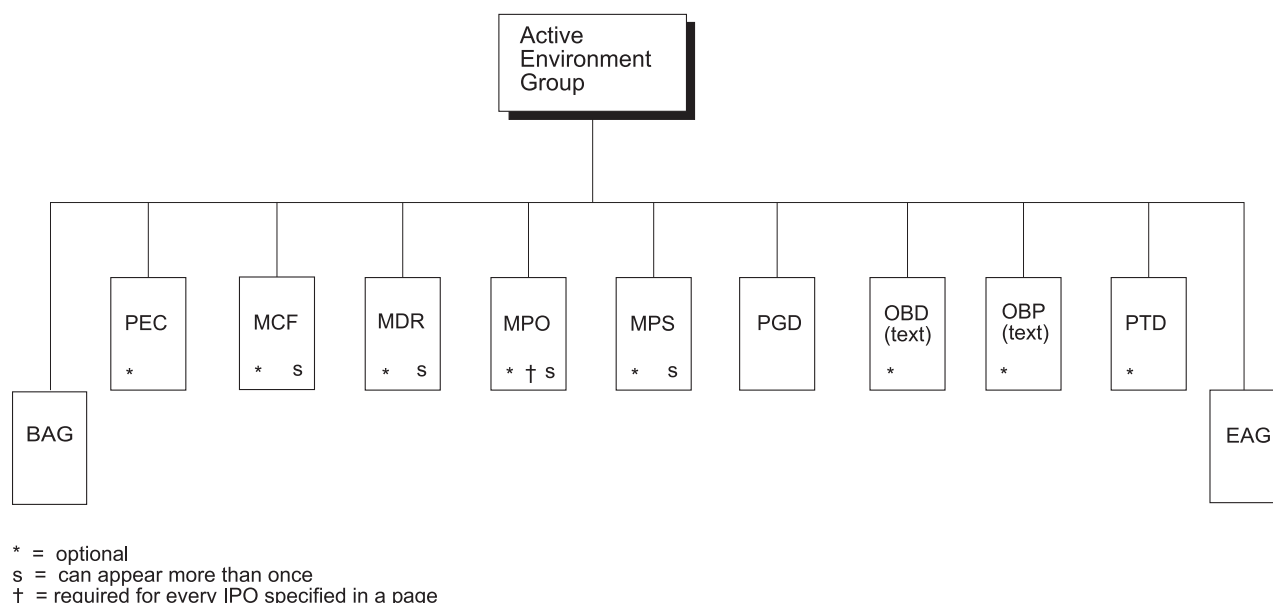
Figure 9. Data Map Active Environment Group Structure for a Page Definition

The Active Environment Group contains structured fields which describe the features and characteristics of the entire page. Page size, names of fonts, page segments, page overlays, and identifiers of objects to be used are all part of the AEG. Because a page may consist entirely of text or entirely of image (as in a page segment), most of these fields are optional. The only required structured field in the AEG is the Page Descriptor, which contains the size of the page. If an application attempts to place data outside the page boundaries, a positioning data-check error will be generated by the printer and reported by PSF.

Font Lists
The Map Coded Font (MCF) and Map Data Resource (MDR) structured fields may be used in the AEG to list fonts to be used on the page. If either are used, each font is assigned a local identifier which is used in the body of the page to select the font for a given line or field.

- Record-format processing

  When the Page Definition specifies record-format processing, font specifications external to the Page Definition are ignored.

  Each font that is used must be mapped with an MCF or MDR in the AEG, and the MCF or MDR should specify the encoding scheme with an Encoding Scheme (X'50') triplet. The values supported in the ESidCP field of the Encoding Scheme ID triplet when printing page numbers with record-format processing are:
  - X'6100'—EBCDIC Presentation SBCS
  - X'6200'—EBCDIC Presentation DBCS
  - X'2100'—PC Data SBCS (ASCII)
  - X'8200'—Unicode Presentation

  The values supported in the ESidUD field of the Encoding Scheme ID triplet when printing page numbers with record-format processing are:
  - X'7200'—UTF-16, including surrogates; byte order is big-endian (UTF-16BE)
  - X'7807'—UTF-8

  The code points used for printing page numbers are:
  - X'F0'–X'F9' for EBCDIC Presentation SBCS

> – X'42F0'–X'42F9' for EBCDIC Presentation DBCS
> – X'30'–X'39' for PC Data SBCS (ASCII) or UTF-8
> – X'0030'–X'0039' for Unicode Presentation or UTF-16

- XML processing

  When the Page Definition specifies XML Data processing, font specifications external to the PageDef are ignored.

  Each font that is used must be mapped with an MCF or MDR in the AEG, and the MCF or MDR must specify the Encoding Scheme ID (X'50') triplet. The values supported in the ESidCP field of the Encoding Scheme ID triplet when formatting XML data with a Page Definition are:
  - X'6100'—EBCDIC Presentation SBCS
  - X'2100'—PC Data SBCS (ASCII)
  - X'8200'—EBCDIC Presentation

  The values supported in the ESidUD field of the Encoding Scheme ID triplet when formatting XML data with a Page Definition are:
  - X'7200'—UTF-16, including surrogates; byte order is big-endian (UTF-16BE)
  - X'7807'—UTF-8

  The code points used for printing page numbers are:
  - X'F0'–X'F9' for EBCDIC Presentation SBCS
  - X'30'–X'39' for PC Data SBCS (ASCII) or UTF-8
  - X'0030'–X'0039' for Unicode Presentation or UTF-16

Table Reference Characters

If the data to be printed contains 3800-style table reference characters, font information is required to map each table reference character to the name of the font to be used when the data is printed. This information can be provided either by font character-set names in job control statements accompanying the data to be printed, or by the fonts mapped in the AEG in the Page Definition. When no fonts are mapped in the AEG but font character-set names are specified in the job control, the first character set specified corresponds to TRC 0, the second to TRC 1, and so forth.

In OS/390, VM, and VSE, the maximum number of characters allowed in the character-set name (CHARS) parameter was four. This presented no problem when 3800 compatibility-mode character sets were used, as none of them had names of more than four characters. But the typographic fonts available for page-mode printing have eight-character names (including a two-character font prefix), and as a result cannot be coded in the CHARS parameter. To associate a table reference character with an eight-character font name, a Page Definition must be built that explicitly makes that mapping. A Page Definition is also required if five or more fonts are to be used. Figure 10 on page 23 provides an example of PPFA source code that could be used to build a Page Definition that addresses both requirements. Here, six table reference characters are defined and each one is associated with a different font of the Sonoran Sans Serif family.

```
SETUNITS LINESP 6 LPI;
PAGEDEF TRCXMP
        WIDTH 8.2 IN HEIGHT 10.8 IN
        REPLACE YES;
  FONT ZERO A0758C;  /* EIGHT-POINT SANS SERIF BOLD   */
  FONT ONE  A0759C;  /* NINE-POINT SANS SERIF BOLD    */
  FONT TWO  A0750C;  /* TEN-POINT SANS SERIF BOLD     */
  FONT THREE A075BC; /* 12-POINT SANS SERIF BOLD      */
  FONT FOUR A0559C;  /* NINE-POINT SANS SERIF ROMAN   */
  FONT FIVE A0550C;  /* TEN-POINT SANS SERIF ROMAN    */

  PAGEFORMAT JSTRC;
    TRCREF 0 FONT ZERO;
    TRCREF 1 FONT ONE;
    TRCREF 2 FONT TWO;
    TRCREF 3 FONT THREE;
    TRCREF 4 FONT FOUR;
    TRCREF 5 FONT FIVE;

    PRINTLINE CHANNEL 1
            POSITION .1 IN .2 IN REPEAT 20;
    ENDSUBPAGE;
```

Figure 10. PPFA Code for Page Definition with Six TRCs to Select Typographic Fonts

The rules for coding Table Reference Characters are different for page mode printers and for the 3800 running in compatibility mode. Table 4 summarizes the differences.

Table 4. Use of TRCs in Page Mode and 3800 Compatibility Mode

Function	Compatibility Mode	Page Mode
Number of table reference characters supported for a single output file	4	127
Valid hexadecimal values for table reference characters	X'F0'–X'F3'	X'F0'–X'F3' or X'00'–X'7E' for 4 or fewer TRCs; X'00'–X'7E' for more than 4 TRCs
How table reference characters are associated with fonts	With character set names in job control language	Same as compatibility mode for 4 or fewer TRCs; with font names in the AEG of the Data Map for more than 4 TRCs

For compatibility with 3800-1 applications, AFP print servers accept TRC values of X'F0' through X'F3' when four or fewer TRCs are used. TRC values of X'00' through X'7E' are valid regardless of how many fonts are used.

The Line Descriptor structured fields in the Data Map contain a bit that indicates which type of TRC to use. PPFA and PMF set this bit to B'1' to indicate

compatibility TRCs when four or fewer TRCs are described in the Page Definition. These software programs set the bit to B'0' when more than four TRCs are used.

Note: Regardless of whether font character set names are specified in the job control or not, if fonts are mapped in the AEG:

- Table reference character 0 (X'00' or X'F0') selects the first font mapped in the Active Environment Group (AEG) of the Data Map; table reference character 1 (X'01' or X'F1') selects the second font mapped in the AEG; and so on. This historically positional selection of fonts mapped in the AEG precludes the use of a mixture of fonts mapped with MCFs and fonts mapped with MDRs. TRCs may be used when all fonts in the AEG are mapped using MCFs only or MDRs only.
- A table reference character higher than 127 selects the first font mapped in the AEG of the Data Map.
- A table reference character higher than the number of fonts mapped defaults to the first font mapped in the AEG of the Data Map.

Page Overlays

If the application uses the Include Page Overlay structured field to place overlays dynamically at any point on the page, a Map Page Overlay structured field must be included in the Active Environment Group containing the name of each overlay to be used.

Page Segments

A Map Page Segment structured field is not required in the Active Environment Group for each page segment to be used by the application, but printer throughput improves if MPS structured fields are included. Mapped page segments are loaded to the printer the first time they are included and are not reloaded on subsequent invocations. These are called *hard* page segments. When a page segment is not mapped in the Active Environment Group of the currently active Data Map, the page segment data is loaded to the printer every time the segment is included by an Include Page Segment structured field. Such segments are called *soft* page segments.

Data Objects

Data objects that are included with an IOB structured field, such as EPS objects and IOCA objects, can be mapped using the MDR structured field. An MDR for such objects is not required in the AEG of the Data Map, but may improve printer throughput if used. Mapped data objects are loaded to the printer the first time they are included, and are not reloaded on subsequent includes.

Color Management

A Color Management Resource (CMR) can be associated with a page, a data object included on the page with an Include Object (IOB) structured field, or a GOCA or BCOCA object generated by the page definition. Associating a CMR is accomplished by using the MDR structured field to reference the CMR as a resource in the AEG for the data map. The CMR reference will be identified as targeted to the page or data object. If a data object is included on a page with an Include Object (IOB) structured field or generated by the page definition, a CMR can be associated with this object by specifying the name of the CMR on the IOB, LND, RCD, or XMD as an external resource reference and then referencing the CMR with a MDR in the AEG of the data map. See the *Mixed Object Document Content Architecture Reference*, SC31-6802, and *Color Management Object Content Architecture Reference*, S550-0511, for more information on Color Management in an AFP environment.

Structured Fields

The structured fields that comprise the Active Environment Group in a Data Map are as follows: (See the *Mixed Object Document Content Architecture Reference*, SC31-6802, for a complete description of these structured fields.)

BAG (Begin Active Environment Group)
Begins an Active Environment Group. A token name may be specified to identify the AEG.

PEC (Presentation Environment Control)
The Presentation Environment Control structured field specifies parameters that affect the rendering of presentation data and the appearance that is to be assumed by the presentation device. The scope of the Presentation Environment Control structured field is the page generated using the data map that contains this structured field.

Note: The PEC structured field in the AEG for the data map is only used to specify the rendering intent for the page using the Rendering Intent triplet; all other PEC triplets are ignored.

MCF (Map Coded Font)
Identifies each font resource object used in the page and assigns each a 1-byte local identifier. The strategic format of the MCF structured field is called the MCF-2; the coexistence format is called the MCF-1.

MDR (Map Data Resource)
Identifies data object resources that are to be downloaded to the printer for subsequent use in the page.

MPO (Map Page Overlay)
Identifies overlay object resources used in the page. Each overlay referenced by an Include Page Overlay structured field in the page must be mapped in an MPO structured field.

MPS (Map Page Segment)
Identifies page segments used on the page that are to be downloaded to the printer.

PGD (Page Descriptor)
Specifies the units of measure for the page presentation space and the size of the page. This parameter is required.

OBD (Object Area Descriptor)
Specifies the units of measure for the text output area and the size of the area. The OBD is optional. If specified, the units of measure must be the same as those specified for the page in the PGD, and the output area size must be the same size as the page.

OBP (Object Area Position)
Specifies the origin and orientation of the text output area on the page, as well as the origin and orientation of the text presentation space on the output area. The OBP is optional. If specified, the origin of the output area and the origin of the text presentation space must be the same as the origin of the page, and the orientation of the output area and of the text presentation space must be 0°.

PTD (Presentation Text Descriptor)
Specifies the units of measure for the text presentation space and the size of the space. For composed page text objects enveloped with BPT and EPT structured fields, the PTD may also specify initial text conditions for the text object. The PTD is required in the AEG if the page contains

presentation text objects. If the PTD is specified, the text presentation space units of measure and size must match the page presentation space units and size specified in the PGD. This descriptor has two formats:

- PTD-1, formerly called CTD, specifies only the text presentation space units of measure and size.

- PTD-2 specifies the text presentation space units of measure, expands the fields containing the presentation space extents from two bytes to three bytes, and allows initial text conditions to be specified for composed page text objects enveloped with BPT and EPT. These initial text conditions are set whenever a BPT structured field starts a new text object, and may be specified on the PTD-2 using the PTOCA control sequences shown in Table 5.

Note that whenever a BPT is encountered in the data stream, AFP presentation servers set the following default page-level initial text conditions *before* the PTD initial conditions are set:

Parameter   **Value**
Initial (I,B) Presentation Position
          (0,0)
Text Orientation
          0°,90°
Coded Font Local ID
          X'FF' (default font)
Baseline Increment
          6 lpi
Inline Margin 0
Intercharacter Adjustment
          0
Text Color   X'FFFF' (default color)

EAG (End Active Environment Group)
    Ends the AEG. Any name specified in the EAG must match the name specified in the BAG.

Table 5. Initial Text Conditions in PTD-2

Initial Text Condition Parameter	Control Sequence
Baseline Increment	Set Baseline Increment (SBI)
Coded Font Local ID	Set Coded Font Local (SCFL)
Initial Baseline Coordinate	Absolute Move Baseline (AMB)
Initial Inline Coordinate	Absolute Move Inline (AMI)
Inline Margin	Set Inline Margin (SIM)
Intercharacter Adjustment	Set Intercharacter Adjustment (SIA)
Extended Text Color	Set Extended Text Color (SEC)
Text Color	Set Text Color (STC)
Text Orientation	Set Text Orientation (STO)

Data Map Transmission Subcase

A Data Map Transmission Subcase can contain LNDs, RCDs, or XMDs, but not a mixture.

Data Map Transmission Subcase with LNDs

Figure 11 shows the structure of a Data Map Transmission Subcase with LNDs. The principal function of the Data Map Transmission Subcase with LNDs is to map
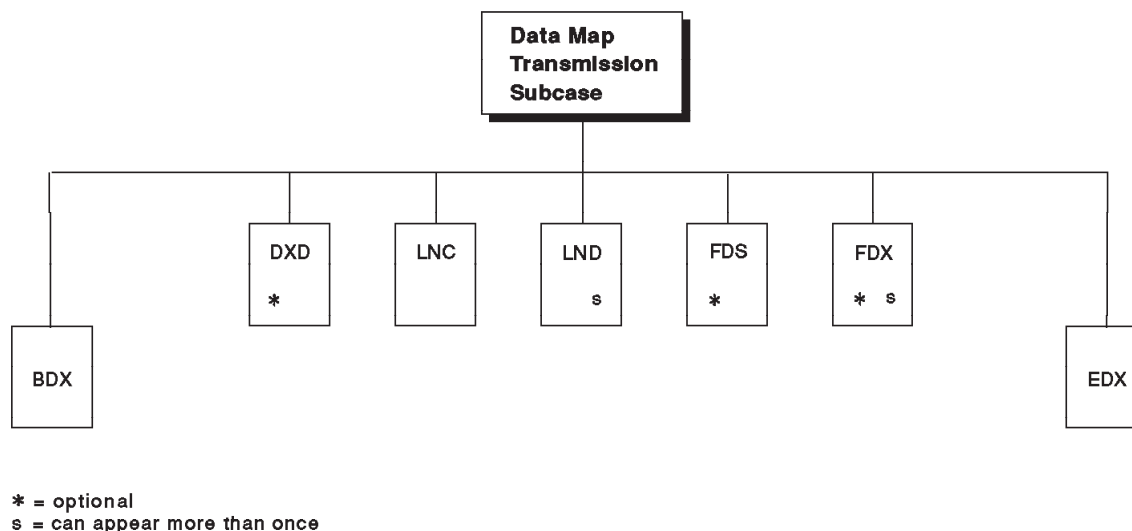
Figure 11. Data Map Transmission Subcase with LNDs

the lines of data to the page. Each line on a page is represented by a Line Descriptor structured field, which contains the horizontal and vertical position on the page where the line is to appear. Rotation and font information is also contained in the Line Descriptors, as is the association with any conditional processing controls used to test data on the current line. The Line Descriptor structured fields are generally used to map lines of text, but they can also be used to position page segments or page overlays or to present line data code points as a bar code. An Include Page Segment or Include Page Overlay structured field which contains a value of X'FFFFFF' in the X-axis positioning parameter, the Y-axis positioning parameter, or both indicates that the page segment or overlay is to be placed at the X-axis or Y-axis position specified by the current LND.

Functions that originated with older printers are also implemented in Line Descriptors. If carriage control skip-to-channel codes are used with the data, each channel code must be defined in at least one Line Descriptor (LND) in the Page Definition; this LND defines the page position associated with that channel code number. Carriage control characters that call for double spacing, triple spacing, or space suppression cause PSF to skip over Line Descriptors or reuse the same Line Descriptor in the Data Map, in a manner analogous to FCBs used with impact printers. Either ANSI or machine code carriage controls can be used, but whichever type is selected must be used for the entire print file. Part of the data cannot contain ANSI carriage controls and another part contain machine code carriage controls. In addition, if carriage control characters are used, every record in the print file must begin with a carriage-control byte, even if it is only "print with single spacing". The type of carriage control being used must be identified in the job control associated with the print file, just as in a non-AFP environment.

The following new functions are provided in Page Definitions that are not available in FCBs. These functions are triggered by information in the Line Descriptor structured field.

- Field formatting, which is the ability to separate specific fields in a line-data record and place them anywhere on a page. Field size, orientation, placement,

color, and font to be used are specified in the Line Descriptor. Fixed text may be specified in the Line Descriptor and added to data from application programs.

- Multiple-up printing, which is the ability to divide the page into sections, each with the appearance of a smaller page. This can be accomplished by defining *subpages*, which are subsets of the page, using Line Descriptors.
- Conditional processing, which is the ability to define tests on certain characters of the line data and perform actions based on the results of the tests.
- Resource object include, which is the ability to include an overlay or page segment and position it relative to the current line. :li.Bar code generation, which is the ability to select a field in a record and present it as a bar code.
- Specification of spot (highlight) colors or process colors for a record or field.
- Object include, which is the ability to include a data object relative to the current line, and change its position, size, and orientation.

Each Line Descriptor formats only one line-data record. The same record may be formatted multiple times on a page using the "reuse record" function in the Line Descriptor. Since Line Descriptors are contained in a Data Map Transmission Subcase whose scope is a page, they cannot be used to place a single record on more than one page.

The *text suppression* function in AFP is an implementation of the 3800-1 COPYMOD function. A combination of information in the Line Descriptor structured field in the Page Definition and the Medium Modification Control structured field in the Form Definition, it provides the same function as "spot carbons" with impact printers, where multi-part forms with carbon paper were often used. Some of these applications required that selected fields not be printed on certain copies of the output (for example, internal prices should not appear on customer copies). The same effect can be obtained with AFP printers by defining fields as *suppressible* in the Page Definition, and then suppressing these fields on certain copies in the Form Definition.

Data Map Transmission Subcase with RCDs

A Data Map Transmission Subcase with RCDs has the same structure as one with LNDs except that the LNDs are replaced with RCDs. A Data Map Transmission Subcase with RCDs is used to process record-format line data instead of traditional line data.

Each record in the data contains a 1 to 250-byte Record ID, and is processed by the RCD in the Data Map Transmission Subcase that contains a matching Record ID. If a CC byte is specified in the record, it must precede the Record ID and is not part of the compare. In addition to providing LND-like functions such as data position, orientation, font selection, coloring, bar code generation, and object includes, RCDs support additional functions like headers, trailers, page numbering, and graphics generation.

Data Map Transmission Subcase with XMDs

A Data Map Transmission Subcase with XMDs has the same structure as one with LNDs except that the LNDs are replaced with XMDs. A Data Map Transmission Subcase with XMDs is used to process XML data instead of traditional line data.

To process XML data, the processor must build a Qualified Tag by concatenating XML start tags. These Qualified Tags are then compared to Qualified Tags in the Data Map. The Qualified Tags in the Data Map are built by specifying a separate XML Name triplet on each XML Descriptor (XMD) for each XML Start tag that has to be traversed in order 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 that as the processor parses the XML, it must buffer the XML start tags traversed in order to have a current Qualified Tag. Each time an end tag is found, the last matching start tag is removed. For example, in the following XML hierarchy:

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

the Qualified Tag for the element <first> is {person name first}. The Qualified Tag for the element <last> is {person name last}. Notice that the tag for element <first> has been removed since its end was received prior to the start tag for element <last>. To process this "current" Qualified Tag, an XMD in the Data Map would also have a Qualified Tag made up from separate XML Name triplets for each XML start tag. This Qualified Tag for this XMD would match the current Qualified Tag and the XMD would be used to present the XML element content "John" on the page.

In addition to providing LND-like functions such as data position, orientation, font selection, coloring, bar code generation, and object includes, XMDs support additional functions like headers, trailers, page numbering, and graphics generation.

Data Map Transmission Subcase Structure

The structured fields that compose the Data Map Transmission Subcase are as follows. (See Chapter 5, "Structured Fields in a Page Definition and in Line Data," on page 71 for a formal description of these structured fields.)

BDX (Begin Data Map Transmission Subcase)
Begins the Data Map Transmission Subcase.

DXD (Data Map Transmission Subcase Descriptor)
Contains constant data.

LNC (Line Descriptor Count)
Specifies the number of Line Descriptor (LND) or Record Descriptor (RCD) structured fields in the Data Map Transmission Subcase.

LND (Line Descriptor)
Specifies how the current line data should be processed. The Data Map Transmission Subcase can contain more than one LND, and each LND points to the next LND used.

When the print file does not use carriage control characters, processing begins with the first LND structured field. When the data record contains a carriage control character that specifies a channel code, the first LND containing that channel code is selected to control processing. If there is no LND in the Data Map containing a channel code matching the channel code specified in the data record, an error is generated.

If an LND specifies that a conditional processing test should be performed on the current record, the LND specifies the field to be tested and the ID of the Conditional Processing Control (CCP) structured field that contains the test criteria and actions. Such LNDs do not place data on the page.

When the LND specifies that fixed text data should be printed, the data is located in the Fixed Data Text (FDX) structured field.

RCD (Record Descriptor)
Specifies how the record with matching record ID should be processed. The Data Map Transmission Subcase can contain more than one RCD.

With RCD processing, carriage controls in the data record are ignored. Processing begins with the first RCD that matches the Record ID of the first record. If a matching RCD is not found, an error is generated.

If conditional processing is to be performed on the current record, the RCD specifies the field to be tested and the ID of the CCP that contains the test criteria and actions. Such RCDs are called *conditional processing* RCDs and do not place data on the page.

When the RCD specifies that fixed text data should be printed, the data is located in the Fixed Data Text (FDX) structured field.

XMD (XML Descriptor)
Specifies how the data with matching start tags should be processed. The Data Map Transmission Subcase can contain more than one XMD.

With XMD processing, carriage controls and table reference characters in the data record are not allowed. Processing begins with the first XMD that matches the start tag. If a matching XMD is not found, the data is ignored and processing resumes with the next start tag.

If conditional processing is to be performed on the current element, the XMD specifies the field to be tested and the ID of the CCP that contains the test criteria and actions. Such XMDs are called *conditional processing* XMDs and do not place data on the page.

When the XMD specifies that fixed text data should be printed, the data is located in the Fixed Data Text (FDX) structured field.

FDS (Fixed Data Size)
If constant text is to be included in line format data, this structured field is required. The FDS specifies the number of bytes of the text that will be found in the following Fixed Data Text (FDX) structured fields. One FDS structured field is used for all FDX structured fields.

FDX (Fixed Data Text)
Must follow an FDS structured field and contains data that can be added to or used instead of line data. More than one FDX structured field is allowed.

EDX (End Data Map Transmission Subcase)
Ends the Data Map Transmission Subcase. Any name specified in the EDX must match the name specified in the BDX.

Field Formatting—LND Processing

A Page Definition may be used to break line-data records into fields that are formatted individually. This is done by building a chain of LND structured fields called a *reuse* chain.

The first LND used to process an input record is called the *base* LND. If this LND specifies flag byte bit 6=B'1' (reuse record), it is also the head of a reuse chain and points to the next LND in the chain with bytes 16–17. This next LND is used to select and process a field in the same record. If additional field processing is

required, the next LND also specifies flag byte bit 6=B'1' and points to another LND to select and process another field in the record, and so on. All LNDs in a reuse chain are called *reuse LNDs*. The last LND in a reuse chain specifies flag byte bit 6=B'0' and bytes 16–17=X'0000'. This LND terminates the reuse chain.

Field Formatting—RCD Processing

Field formatting is also supported when RCDs are used to process record-format line data. The first RCD used to process an input record is called a *record* RCD. It is identified by RCDFlgs bit 6=B'0' and RCDFlgs bit 11=B'0'. If the FLDrcd parameter in a record RCD is non-zero, it specifies the RCD number of a *field* RCD that is to be used to process a field in this record. A field RCD is identified by RCDFlgs bit 6=B'1' and RCDFlgs bit 11=B'0'. Multiple field RCDs can be chained to a record RCD in this manner. The last field RCD in this chain must specify FLDrcd= X'0000'.

Field Formatting—XMD Processing

Field formatting is also supported when XMDs are used to process XML data. The first XMD used to process a start tag is called an *element* XMD. It is identified by XMDFlgs bit 6=B'0', XMDFlgs bit 10=B'0', and XMDFlgs bit 11=B'0'. If the FLDxmd parameter in an element XMD is non-zero, it specifies the XMD number of a *field* XMD that is to be used to process a field in this element data. A field XMD is identified by XMDFlgs bit 6=B'1' and XMDFlgs bit 11=B'0'. Multiple field XMDs can be chained to an element XMD in this manner. The last field XMD in this chain must specify FLDxmd=X'0000'.

Using Conditional Processing in a Page Definition

The conditional processing function allows a different Data Map in the current Page Definition, a different Medium Map in the current Form Definition, or both to be selected for use with the next page based on characteristics of the application data stream. This provides a way to change Data Maps or Medium Maps as necessary without having to make application programming changes. The new format can take effect either before or after a specified line or a specified subpage. With LND-based Data Maps, a *subpage* is a subset of the lines presented on a page. Subpages are defined in the Data Map by the user when coding a Page Definition, and are often used to create multiple-up Page Definitions. Subpages are ignored with RCD-based and XMD-based Data Maps, that is, each page is a single subpage.

Conditional processing is implemented by a combination of structured fields in the Page Definition. CCP structured fields specify the test to be performed and action to be taken, while LND, RCD, and XMD structured fields include the location and length of data fields to be tested and a pointer to the CCP. When the conditions of a test are satisfied, the actions that can be taken are switching to a new Data Map, switching to a new Medium Map, or both, either before or after the current line or subpage is printed. When the action takes effect, printing of the current page ends. If a new Data Map is selected, printing resumes on a new side of a sheet of paper. If a new Medium Map is selected, printing resumes on a new physical sheet. As a result, it is *not* possible to format part of a page with one Data Map and format another part of the same page with a different Data Map. Note that the Medium Map may specify the N-up function, which places multiple pages into partitions on a sheet side. When N-up is specified, switching to a new Data Map or a new Medium Map may cause printing to resume in a new partition instead of on a new

sheet-side or a new sheet. For more information on N-up printing, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Conditional processing can be used to format subsets of a single output file differently, or it can be used to force an eject to a new page or sheet based on some condition. Examples of these uses of conditional processing are shown below.

Using Different Formats for Different Subsets of Output

A common example of this is an application program requirement to print detail pages of a report in a different format from summary pages. Assuming that a known field in the application data stream can be tested to identify the detail records and the summary records, a Page Definition with two Data Maps can be constructed to provide the different formats without changes to the application program. Figure 12 assumes a file where each record has identifying information in bytes 2 through 5. Records with the characters DETL in these positions are to use Data Map PF1 and Medium Map CG1. Records with the characters SUMM in these positions are to use Data Map PF2 and Medium Map CG2. Figure 12 shows the PPFA/370 code that generates a Page Definition to test these positions and to print the detail pages in the ACROSS direction and the summary pages in the DOWN direction.

```
SETUNITS 1 IN 1 IN ;
LINESP 6 LPI ;
PAGEDEF CPSAM1 REPLACE YES ;
  PAGEFORMAT PF1
           WIDTH 8.2 HEIGHT 10.0
           DIRECTION ACROSS ;
    PRINTLINE REPEAT 1
             CHANNEL 1 ;
    CONDITION TEST1 START 2 LENGTH 4
             WHEN EQ 'SUMM'
                  BEFORE SUBPAGE
                  COPYGROUP CG2
                  PAGEFORMAT PF2 ;
    PRINTLINE REPEAT 59 ;
    ENDSUBPAGE ;

  PAGEFORMAT PF2
           WIDTH 10.0 HEIGHT 8.2
           DIRECTION DOWN;
    PRINTLINE REPEAT 1
             CHANNEL 1 ;
    CONDITION TEST2 START 2 LENGTH 4
             WHEN EQ 'DETL'
                  BEFORE SUBPAGE
                  COPYGROUP CG1
                  PAGEFORMAT PF1 ;
    PRINTLINE REPEAT 34 ;
    ENDSUBPAGE ;
```

Figure 12. PPFA Code for Page Definition with Conditional Processing

Conditionally Skipping to a New Page or a New Sheet

Another common use of conditional processing is to skip to a new page or a new sheet when a control break in the output data occurs. This control break might be the start of a new customer number, a new department, or some other change in the output that requires starting on a new page, or on a new sheet of paper.

Such cases include applications that print using multiple-up format, where having data for one department appear on the left-hand half of the page while having data for a different department appear on the right-hand half of the page is not desirable. This possibility can be avoided by having the application force a completely new page when the department number changes. In PPFA, this condition is coded with the NEWSIDE parameter.

Applications that print duplex output (using both front and back of the form) probably must force a new physical sheet at a control break in the data, to avoid having output for two different user destinations on the front and back of the same sheet. In PPFA, this condition is coded with the NEWFORM parameter. For output printed multiple-up on both sides of the sheet, the NEWFORM parameter forces a new page and a new sheet. Coding both is not necessary.

A new page or sheet can be forced when using a Page Definition with only one Data Map or a Form Definition with only one Medium Map. Conditional processing can be used to re-invoke the currently active Data Map or Medium Map when the condition is satisfied. This is what happens when NEWSIDE or NEWFORM is coded in PPFA. More than one Data Map or Medium Map is required only if subsets of the output are to be formatted or handled differently based on the defined condition. Note that if the Medium Map specifies the N-up function, the new "sheet" may actually be a new N-up partition on the sheet.

The example in Figure 13 shows PPFA source code to accomplish a skip to a new page when the department number in character positions 1 through 3 changes.

```
SETUNITS 1 IN 1 IN ;
LINESP 8 LPI ;
PAGEDEF NEWPG REPLACE YES
       WIDTH 10.5 HEIGHT 8.1
       DIRECTION DOWN ;
  PAGEFORMAT NEWPG ;
    PRINTLINE REPEAT 40
             CHANNEL 1
             POSITION .5     TOP ;
    CONDITION TEST1 START 1 LENGTH 3
             WHEN CHANGE
             BEFORE SUBPAGE
             NEWSIDE ;
    ENDSUBPAGE ;
```

Figure 13. PPFA Code for Page Definition to Skip to New Page

The example in Figure 14 on page 34 is similar; but in this case the skip is to a new sheet, or form, where printing of the output is resumed.

```
SETUNITS 1 IN 1 IN ;
LINESP 8 LPI ;
PAGEDEF NEWFM REPLACE YES
        WIDTH 10.5 HEIGHT 8.1
        DIRECTION DOWN ;
  PAGEFORMAT NEWFM ;
    PRINTLINE REPEAT 40
              CHANNEL 1
              POSITION .5      TOP ;
    CONDITION TEST1 START 1 LENGTH 3
              WHEN CHANGE
              BEFORE SUBPAGE
              NEWFORM ;
    ENDSUBPAGE ;
```

Figure 14. PPFA Code for Page Definition to Skip to New Sheet

Processing Line Data with Shift-Out/Shift-In (SOSI) Controls

Shift-out (SO-X'0E') and shift-in (SI-X'0F') controls are used to signal the beginning and end, respectively, of a string of double-byte code points that are to be rendered using characters from a double-byte font or rendered as a QR Code bar code symbol. SOSI processing is specified in the print request and applies to both fixed text fields and the input line data.

SOSI processing for text output is supported by two modes of font selection in the PageDef. Both modes may be intermixed in the same Page Map.

- *Record-based or field-based font selection.* In this mode, the font to be used following an SO can be uniquely selected for each record or field by specifying a non-zero shift-out font local ID in byte 26 of the LND or byte 34 of the RCD that is used to process the line data. The font used following an explicit SI is then always the primary font specified in byte 10 of the LND or byte 23 of the RCD, and use of this font must be enabled with flag bit 4 = B'1'. An error condition exists if flag bit 4 = B'0'. Note that an implicit SI is assumed at the start of every record. This selects the primary font specified in byte 10 of the LND or byte 23 of the RCD, if it is enabled with flag bit 4 = B'1'.

- *Page-based font selection.* In this mode, the font to be used following an SO is the same for all records and fields on the page. LND byte 26 or RCD byte 34 is set to X'00', and the font used following an SO is the font mapped to local ID X'02' in the AEG for the Data Map. The font used following an explicit SI is the font mapped to local ID X'01' in the AEG. Note that the font used following the implicit SI at the start of every record is still the primary font specified in byte 10 of the LND or byte 23 of the RCD, as long as it is enabled with flag bit 4 = B'1'. Both an SO font and an SI font must be mapped in the AEG with the proper local IDs. The presence of only one mapped font is an error condition. If no fonts are mapped in the AEG, a presentation system default may be used.

The SO and SI controls used to delimit the strings of double-byte code points are not valid printable characters nor are they valid QR Code bar code data characters. The line data processor must either remove or convert the SO and SI characters to blanks according to the selection of SOSI processing mode in the print request.

For text output, the SOSI processing modes are described as follows:

SOSI mode   Action

SOSI1   Specifies that each shift-out, shift-in control is to be converted to a blank and a Set Coded Font Local text control.
- Each SO (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 font mapped to local ID X'02'.
- Each SI (X'0F') is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the font mapped to local ID X'01', followed by a blank (X'40').

SOSI2   Specifies that each shift-out, shift-in control is to be converted to a Set Coded Font Local text control.
- Each SO (X'0E') is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the font mapped to local ID X'02'.
- Each SI (X'0F') is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the font mapped to local ID X'01'.

SOSI3   Specifies that the shift-in control is to be converted to a Set Coded Font Local text control and two blanks. A shift-out control is to be converted to a Set Coded Font Local text control.
- Each SO (X'0E') is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the font mapped to local ID X'02'.
- Each SI (X'0F') is replaced with a PTOCA structure that contains a Set Coded Font Local text control for the font mapped to local ID X'01', followed by two blanks (X'4040').

SOSI4   Specifies that each shift-out, shift-in control is to be skipped and not counted when calculating offsets for the print data. The conversion of the shift-out and shift-in controls for SOSI4 is the same as for SOSI2.

> **Note:** 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 computing offsets to various fields within the data.

For QR Code bar code output, the data is converted to Shift/JIS ASCII data. The SO and SI control characters are removed and are *not* converted to blanks and Set Coded Font Local text controls as they are for text output. The converted data is then used as the QR Code bar code data. This processing is the same for all SOSI processing modes. For SOSI4, each shift-out and shift-in control is not counted when computing offsets to various fields within the data.

When processing data with SOSI controls, the processor assumes that each line or record starts with single-byte code points. This means that the data is scanned for SOSI controls one byte at a time. After processing a shift-out control, the data is scanned two bytes at a time. The first byte of each pair is checked to see if it is a shift-in control. If a line is to start with double-byte data, the first byte in the line must be a shift-out control. This is due to the fact that single-byte code points are assumed at the start of each line. The processor also assumes that each field (including fixed text fields) starts with a single-byte code point when processing in SOSI1, SOSI2, and SOSI3 modes. The processing of fields in SOSI4 mode is different than the other SOSI modes. SOSI4 processing requires that SO and SI

controls not be counted as part of the field positioning. Therefore, the record is scanned to keep track of the last SO or SI prior to the field start position. The SO or SI control found prior to the field is used to determine if the field starts with a single-byte or double-byte code point.

Notes:

1. Since table reference characters (TRCs) also may use the fonts mapped to local IDs X'01' and X'02' in the AEG of the Data Map, it is recommended that the mixing of SOSI controls and TRCs be avoided when using page-based font selection.

2. Shift-out/Shift-in controls are not used in Unicode data to signify a shift into and out of DBCS processing. Therefore, it is not possible to switch processing between Unicode encoding and single-byte (SBCS) encoding within a line data field or record. That is, when a line data field is processed with a Page Definition, either the whole field is treated as Unicode-encoded, or none of it is treated as Unicode-encoded.

3. When building bar codes from line data, SOSI input data is not appropriate for bar code symbologies other than QR Code. Refer to the *Bar Code Object Content Architecture Reference*, S544-3766 for information about the valid encoding for each bar code.

4. When using Shift-out/Shift-in controls with Record Format data using delimited fields, if the field is to print using double-byte code points, the SO control must follow the delimiter for the field.

5. Shift-out/Shift-in controls are not supported when processing XML data.

Printing Bar Codes with a Page Definition

A Page Definition can be used to print a bar code symbol using data from one of the following places:
- Line data record
- XML element
- Field in the line data record
- Field in the XML element

This is done by specifying a Bar Code Symbol Descriptor (X'69') triplet on the LND, RCD, or XMD. The presence of this triplet indicates to the presentation services program that the selected field is to be presented as a bar code symbol. The position specified by the LND, RCD, or XMD indicates the position of the symbol origin, and the text orientation specified by the LND, RCD, or XMD indicates the rotation of the symbol with respect to the page X_P-axis.

Note that the text suppression function is not supported when the field is presented as a bar code. This function is supported only for text and is ignored for other data. Note also that the bar code function is not supported on LNDs, RCDs, or XMDs that specify conditional processing (flag bit 11 = B'1'); if the Bar Code Symbol Descriptor triplet is specified it is ignored.

For improved printer throughput, all bar code symbols on a page that use the same descriptor and that specify the same rotation are grouped into a single bar code object by the presentation services program before the page is presented. To align the object presentation space X_{bc}-axis with the X-axis of the bar code symbol, the origin of the object presentation space is selected as one of the four corners of the page based on the LND, RCD, or XMD text orientation. The bar code presentation space origin is therefore made coincident with the current text coordinate system (I,B) origin. For example, if an LND specifies a (90°,180°) text orientation, the symbol rotation is 90° and the origin of the bar code object

presentation space is the top-right corner of the page. The extents of the bar code object presentation space are determined by the extents of the page presentation space. For example, if the origin of the object presentation space is the top-right corner of the page, the X-extent of the object presentation space is the Y_p-extent of the page, and the Y-extent of the object presentation space is the X_p-extent of the page. The symbol origin offset from the object presentation space origin and from the current text (I,B) coordinate system origin is specified by the *IPos* and *BPos* parameters of the LND, RCD or XMD. The units of measure for the bar code object presentation space, used for determining symbol origin offsets, are the same as those defined on the page $(X_p., Y_p)$ presentation space in the PGD structured field of the Active Environment Group (AEG) for the Data Map.

The presentation services program also defines an object area presentation space for the object that is identical in size, position, and units of measure to the bar code presentation space. The rotation of the object area presentation space about the page X_p-axis is the same as the rotation of the bar code symbol about this axis, which is the same as the text orientation specified in the LND, RCD, or XMD.

Printing Graphics with a Page Definition

A Page Definition can be used to generate simple graphics primitives such as lines, boxes, circles, and ellipses when record-format line data is processed with RCDs or XML data is processed with XMDs. This is done by specifying a Graphics Descriptor (X'7E') triplet on the RCD or XMD. This triplet may specify a complete graphics primitive, as is always the case with a full arc, or it may specify the beginning or end of the primitive.

For improved printer throughput, all graphics primitives on a page with the same descriptor and the same orientation are grouped into a single graphics object by the presentation services program before the page is presented. The origin for the graphics object area is one of the four corners of the page as determined by the text orientation specified in the *TxtOrent* parameter of the RCD or XMD and is therefore coincident with the current (I,B) origin. The rotation of the graphics object area about the page X_p-axis matches the rotation of the current text (I,B) coordinate system. For example, with a (90°,180°) text orientation, the object area rotation is 90°. The extents of the object area match the extents specified in the Margin Definition (X'7F') triplet on the BDM. The position of the graphics primitive in the (I,B) coordinate system therefore maps to the same position in the object area (X_{oa}, Y_{oa}) coordinate system. This in turn is mapped to a graphics window whose upper left corner is at the graphics presentation space (GPS) origin, and whose extents match those of the object area. The upper left corner of the graphics presentation space window is therefore also coincident with the current (I,B) origin. The mapping between graphics window and object area is *position and trim*.

For example, if the RCD specifies a (90°,180°) text orientation, the upper left corner of the graphics window is at the top-right corner of the page, and graphics primitives in this object are rotated 90°with respect to the page X_p axis. The X-extent of the graphics window is the Y-extent of the graphics object area, and the Y-extent of the graphics window is the X-extent of the graphics object area. The units of measure for the graphics presentation space and for the graphics object area are the same as those defined on the page (X_p, Y_p) presentation space in the PGD structured field of the Active Environment Group (AEG) for the Data Map.

Relative Baseline Positioning—LND Processing

Records, fields, and objects are positioned using the print position specified in bytes 2–5 of the LND structured field. These bytes normally specify an absolute offset from the origin of the current text (I,B) coordinate system. With relative baseline positioning, LND bytes 4–5 may be used to specify a baseline position *relative* to an established baseline position. This allows records, fields, and objects to be positioned (in the baseline direction) relative to a previous record, field or object.

Relative baseline positioning is used when LND flag byte bit 13 is set to B'1'. The relative offset may be positive or negative and is measured using the current I,B coordinate system. Note that the origin of the current I,B coordinate system depends on the current text orientation. The baseline position used as a reference for the relative offset depends on whether the LND that specifies relative positioning is a base LND, and on whether a page or subpage boundary was crossed since the last LND was used to print. The baseline position used as a reference for the relative offset is determined as follows:

- For base LNDs, offsets are defined relative to the last base LND processed, either by printing or by spacing. However, if a page or subpage boundary was crossed after the last base LND was processed, offsets are defined relative to the first LND for the page or subpage.

- For reuse LNDs other than base LNDs, the offset is defined relative to the last LND used to print.

- If the first LND of a Data Map specifies relative positioning, the offset is defined relative to the current text coordinate system origin (I=0,B=0), using the current text (I,B) coordinate system.

- If the first LND of a subpage specifies relative positioning, the offset is defined relative to the last print position, using the current text (I,B) coordinate system. Note that when skipping into a subpage, if the skipped-to LND specifies relative positioning, the relative offset is measured with respect to the first LND of the subpage, which may specify a relative position as well. This function allows a subpage to "float" relative to the last print position.

The following restriction applies to relative baseline positioning:

- The text orientation of an LND that specifies relative baseline positioning must be the same as the text orientation of the LND that defines the baseline position from which the relative offset is measured.

Note that if the line data processed with relative baseline positioning LNDs contains carriage controls that specify double or triple spacing, the presentation system must accumulate the relative offsets of the skipped LNDs in order to achieve the proper line spacing. If an LND that specifies absolute positioning is skipped, the position is reset to the absolute position and the relative offsets of any additional skipped LNDs are accumulated with respect to the absolute position. When a page boundary is crossed, printing resumes at the first LND.

Application Note: When relative baseline positioning is used, the PageDef generator cannot check for off-page errors, since the data normally determines, with skip-to-channel carriage controls, when the relative baseline LNDs are invoked. AFP print servers will generate a page break if the active Data Map is about to position data past the page's y-extent. This will not cause the generation of an error message. Note that the page's y-extent is specified in the PGD of the Data Map.

Skip-to-Channel Processing for Relative Baseline Positioning

When a skip-to-channel carriage control is received, the remainder of the LNDs are searched sequentially for a matching channel code until the end of the subpage is reached. If no matching channel code is found, and if the skip is not to channel 1, a search is made for a *relative*. LND with matching channel code starting at the top of that subpage; if found, processing continues with this LND. If no relative LND with matching channel code is found in the subpage, or if the skip is to channel 1, the search for *any*. LND with matching channel code continues in the next sequential subpage. If the end of the Data Map is reached, a new page is started, and the Data Map is searched, starting at the beginning, for *any*. LND with matching channel code.

Relative Baseline Positioning—RCD Processing

Relative baseline positioning can also be used when record-format line data is processed with RCDs.

Relative baseline positioning is used when flag byte bit 13 is set to B'1'. The baseline position used as a reference for the relative offset depends on whether the RCD that specifies relative positioning is a record RCD and is determined as follows:

- For record RCDs, offsets are defined relative to the last record RCD processed. However, if a page boundary was crossed after the last record RCD was processed, offsets are defined relative to the top margin.
- For field RCDs, the offset is defined relative to the last RCD used to print.
- If the first RCD of a Data Map specifies relative positioning, the offset is defined relative to the top margin.

The following restriction applies to relative baseline positioning:

- The text orientation of an RCD that specifies relative baseline positioning must be the same as the text orientation of the RCD that defines the baseline position from which the relative offset is measured.

Relative Baseline Positioning—XMD Processing

Relative baseline positioning can also be used when XML data is processed with XMDs.

Relative baseline positioning is used when flag byte bit 13 is set to B'1'. The baseline position used as a reference for the relative offset depends on whether the XMD that specifies relative positioning is an element XMD and is determined as follows:

- For element XMDs, offsets are defined relative to the last element XMD processed. However, if a page boundary was crossed after the last element XMD was processed, offsets are defined relative to the top margin.
- For field XMDs, the offset is defined relative to the last XMD used to print.
- If the first XMD of a Data Map specifies relative positioning, the offset is defined relative to the top margin.

The following restriction applies to relative baseline positioning:

- The text orientation of an XMD that specifies relative baseline positioning must be the same as the text orientation of the XMD that defines the baseline position from which the relative offset is measured.

Relative Inline Positioning—XMD Processing

Data and objects are positioned using the print position specified in *IPos* and *BPos* parameters of the XMD structured field. The IPos normally specifies an absolute offset from the origin of the current text (I,B) coordinate system. With relative inline positioning, the IPos parameter may be used to specify an inline position *relative* to an established inline position. This allows data and objects to be positioned (in the inline direction) relative to data placed previously on the page. If no data were placed on the page prior to the current data, the relative inline position is relative to the left margin. Note that the actual location of the left margin on a page is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76.

Relative inline positioning is used when XMD flag byte bit 12 is set to B'1'. The relative offset may be positive or negative and is measured using the current I,B coordinate system. Note that the origin of the current I,B coordinate system depends on the current text orientation.

The following restriction applies to relative inline positioning:

- The text orientation of an XMD that specifies relative inline positioning must be the same as the text orientation of the XMD that defines the inline position from which the relative offset is measured.

Note: Data must not exceed the boundaries of the page, which are defined in the Page Descriptor (PGD) structured field. If the new print position is outside these boundaries, printing of the page stops.

The Function of the Form Definition

A Form Definition is a MO:DCA print control object that is used to place pages on sheets and is always required for printing with PSF. Form Definitions contain information about the physical environment in which the output is to be printed, such as the paper drawer to be used and whether printing is to be done in simplex or duplex mode. The Form Definition may also specify overlays to be used with the data. Two types of overlays may be specified in a Form Definition: *medium* overlays and *PMC* overlays. Medium overlays are positioned at the medium origin, while PMC overlays are positioned with respect to the page origin. Contrast these with overlays that are mapped in a Page Definition, which are invoked for a page using an Include Page Overlay (IPO) structured field and are called *page* overlays. The overlays themselves must be generated with a separate program designed to build overlay objects. The format for medium overlays and for PMC overlays (invoked in Form Definitions) is the same as the format for page overlays (invoked with an IPO structured field and mapped in Page Definitions).

Form Definitions are like Page Definitions in that only one Form Definition can be associated with a given print file, and also in that each Form Definition includes one or more components. While the components of a Page Definition are called Data Maps or Page Formats, the components of a Form Definition are called Medium Maps or Copy Groups. An application program can switch between Medium Maps by using conditional processing, as in the example in Figure 12 on page 32. Control for presentation starts with the first Medium Map in the Form Definition. Control for presentation may be changed to a different Medium Map by using an Invoke Medium Map (IMM) structured field. If the Form Definition is used to present multiple documents in a print file, control for presentation is returned to the first Medium Map whenever a new document is encountered.

A file can be printed multiple times, each with a different Form Definition. The example in Figure 5 on page 16 can be modified to add Form Definition names in addition to Page Definition names.

Details on Form Definitions and overlay objects can be found in the *Mixed Object Document Content Architecture Reference*, SC31-6802. A set of Form Definitions which address standard requirements is provided with PSF software, but users can create customized Form Definitions by using any of the software products listed on page 14.

Chapter 4. Mixed Documents: Adding MO:DCA Structured Fields to Line Data

Chapter 3, "Using a Page Definition to Print Data" describes how Page Definitions can be used to format traditional application line data without the need to make any application programming changes. Under certain circumstances, however, functions are needed that can only be accomplished by changing the application. These functions can be invoked by using one of a small set of MO:DCA structured fields, any of which can be intermixed with line data to obtain specific results. A document of this type, in which structured fields are intermixed with line data, is called a *mixed document*.

Note: MO:DCA structured fields cannot be combined with XML data.

MO:DCA structured fields cannot be interspersed with line data records indiscriminately. The data object structured fields described in Table 15 on page 188 and the resource structured fields described in Table 16 on page 193 can appear only within their respective objects and resources, and only in the sequence shown in the tables. For example, the Map Coded Font (MCF) structured field is part of the Active Environment Group which in turn can appear in a presentation page, an overlay, or a Page Definition. However, it is not permitted to include an MCF between line-mode data records in an output file or to bracket line-mode records with Begin Page and End Page structured fields.

This chapter discusses how data and resource objects can be intermixed with line data, and provides examples of structured fields that can be included individually with line-data records. These structured fields are:
- Invoke Data Map
- Invoke Medium Map
- Include Page Segment
- Include Page Overlay
- Include Object
- Presentation Text

Note: The No Operation (NOP) structured field may appear anywhere in a mixed document and thus is not listed in the structured field groupings.

This chapter contains coding examples for some of these structured fields. Chapter 5, "Structured Fields in a Page Definition and in Line Data" contains additional information on the format of these structured fields. See the *Mixed Object Document Content Architecture Reference*, SC31-6802, for the formal definition of all MO:DCA structured fields.

The presence of structured fields in line data does not change the fact that the Page Definition is the controlling resource which determines how the data appears on the page. Structured fields other than those that change Data Maps or Medium Maps do not affect the placement of line-data records, nor do they affect the text orientation or font selection used to print line-data records. These characteristics of line-data records are defined in the Page Definition. Only when the application generates fully composed documents is a Page Definition not used.

X'5A' Carriage Control Character

When printing in a System/390 environment, if MO:DCA structured fields are used either in a fully composed document or intermixed with line data, each MO:DCA structured field must be prefixed with a X'5A' character. The X'5A' appears in the first byte position and provides a signal to PSF that the record is a structured field, not a data record.

The X'5A' character precedes the MO:DCA structured field and is not formally part of the structured field, so it is *not* counted in the structured field length value which immediately follows it. The examples in this chapter all contain a X'5A' character in the first byte position.

In a System/390 environment, each MO:DCA structured field must occupy one record. The requirement to prefix MO:DCA structured fields with X'5A' means that all other records in the data set must begin with a carriage control character, even if it is only a "print-and-space-one-line" carriage control. Either ANSI or machine code carriage control can be used for these records.

In an AIX environment, the carriage control character is optional. The New Line control, also called Linefeed, (X'25' in EBCDIC, X'0A' in ASCII) is used to determine end-of-record in AIX. The use of the Linefeed carriage control to determine end-of-record allows variable-length records to be easily created in AIX environments.

Print File Structure

An AFP print file consists of an optional inline resource group followed by one or more documents. Each document may, in turn, be preceded by an optional document index. All resources in an inline resource group must precede all other data in the print file. The group of resources is delimited by the Begin Resource Group (BRG) and End Resource Group (ERG) structured fields. Each resource object in the group is delimited by the Begin Resource (BR) and End Resource (ER) structured fields. If multiple fully composed documents are present in the print file, they must be delimited by Begin Document (BDT) and End Document (EDT) structured fields. Note that mixed line-page documents and composed documents can occur in any order following the inline resource group. Figure 15 on page 45 shows the structure of an AFP print file.
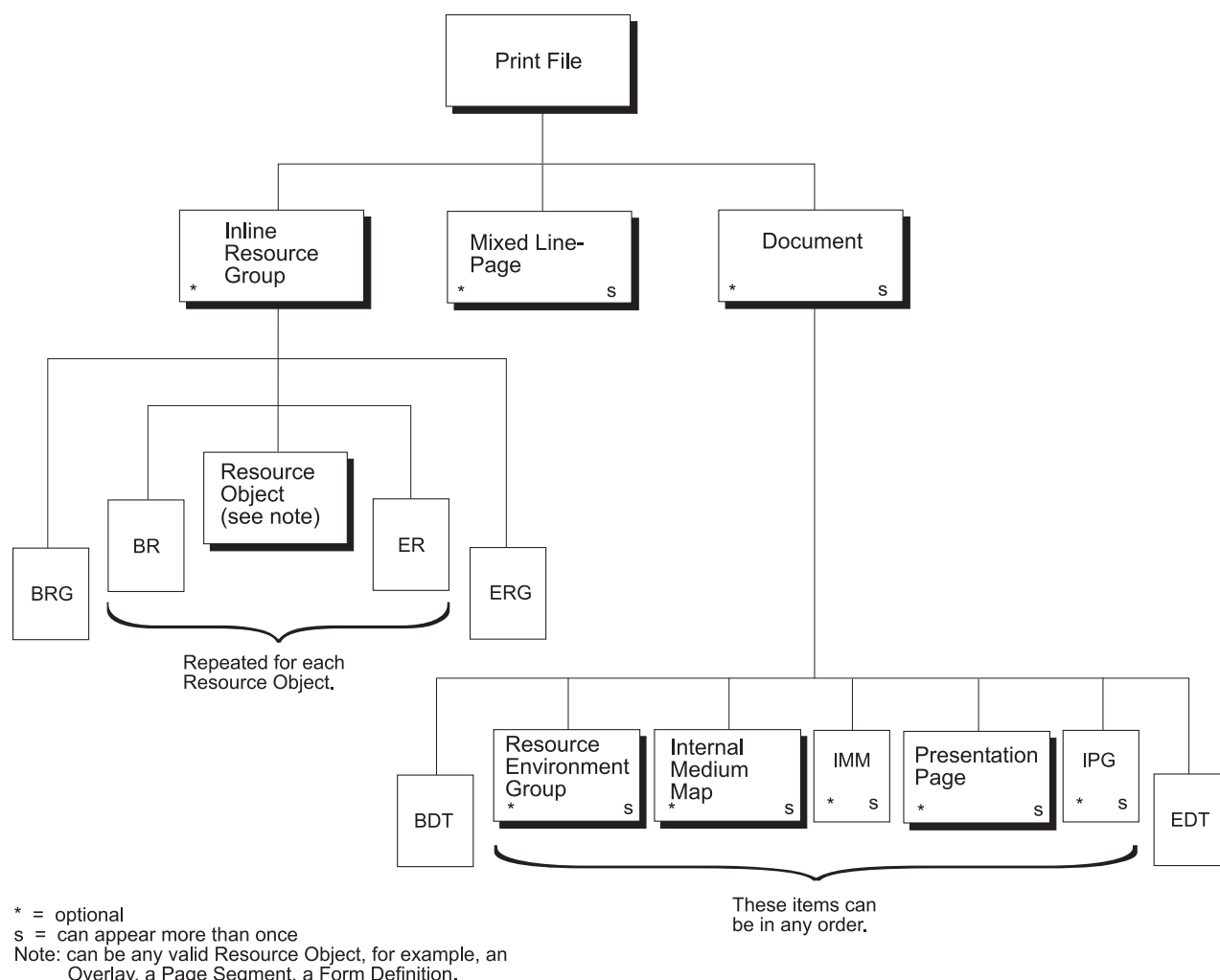
Figure 15. Structure of a Print File

Notes:

1. The mixed line-page documents and composed documents can occur in any order following the inline resource group.

2. Each AFP (MO:DCA-P) document may optionally be preceded by a single document index that is implicitly tied to the document and that indexes the document. For the formal definition of the MO:DCA-P document index, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

3. An AFP (MO:DCA-P) document may contain Link Logical Element (LLE) structured fields following the BDT. It may also group presentation pages into named page groups. MO:DCA-P page groups may in turn contain Tag Logical Element (TLE) structured fields following BNG. These structures do not affect the presentation of the document. To see which AFP print servers support these structures, see Appendix D, "System Support Information," on page 183. For the formal definition of these structures, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

4. If a Medium Map is included internal (inline) to the document, it is activated by immediately following it with an IMM that explicitly invokes it, otherwise the internal Medium Map is ignored. An IMM that does not follow an internal

Medium Map may not invoke an internal Medium Map elsewhere in the document and is assumed to reference a Medium Map in the current Form Definition.

The objects that comprise an AFP print file are as follows:

Inline Resource Group
Contains one or more resource objects to be associated with printing this file. See "Inline Resource Group Structure" on page 47 for a detailed description of the structure of the resource group and the objects it can contain.

Note: In the MO:DCA architecture, these resource groups are called *external* resource groups because they occur outside a document.

The Inline Resource Group is an optional component of the Print File. If no Inline Resource Group is defined, the resources stored in the AFP resource library of the system are used. (In MVS/ESA™ with USERLIB support, resources may be stored in private libraries which are used at print time for individual data sets. Up to eight private libraries may be used with a single data set. The libraries are named in the USERLIB parameter of the OUTPUT JCL statement.)

The scope of an inline resource group is the print file. Once the last document in the print file has been processed, the resources in the resource group are no longer available to the presentation system for use with another print file.

Documents
The print file may contain one or more documents to be printed. These may be fully composed-page documents, line-mode documents, or mixed-mode documents, in any order. If multiple composed-page documents appear, each one must be delimited by a BDT and an EDT structured field. For the complete definitions of document structure, see Appendix B, "Document and Resource Object Diagrams," on page 163.

Finishing Operations for a Print File

A Form Definition may be used to specify finishing operations to be applied to the documents in a print file. The scope of the finishing operations as well as the type of operation is specified with a Medium Finishing Control (MFC) structured field in the Document Environment Group (DEG) of the Form Definition. For a definition of the finishing operations and parameters that may be specified, see the *Mixed Object Document Content Architecture Reference*, SC31-6802. The following rules specify how the scope of the finishing operations applies to a print file when the file contains line-data and mixed-data documents, with or without BDT/EDT, as well as composed documents.

- If the MFC specifies print-file level finishing, all media in the print file is collected for finishing in a print-file level media collection, and the finishing operations are applied to the complete collection, that is, the complete print file.
- If the MFC specifies document-level finishing and selects all documents, the print file is processed as a set of documents as follows:
  - Any document bounded by BDT/EDT is processed as a single document regardless of whether the data between BDT/EDT is line data, mixed data, or composed data.
  - Line data and mixed data that is not bounded explicitly by BDT/EDT is processed as an implied document with implied BDT/EDT. When such data

follows the resource group or an EDT, a BDT is implied, and the implied document lasts until a BDT is encountered or until the end of the print file is reached. In either case, the implied document is terminated with an implied EDT.

The media in each document, whether implied or explicit, is collected for finishing in a document-level media collection, and the finishing operations are applied to each collection, that is each document, individually. Note that, in this case, the same finishing operations are applied to each document.

- If the MFC specifies document-level finishing and selects a single document, the print file is processed as a set of documents in the same manner as when all documents are selected. The offset of the selected document is calculated by counting all documents, whether implied or explicit, and the selected document may itself be an implied document. The media in the selected document are collected for finishing, and the finishing operations are applied to the single collection, that is the single document. If the same document is selected multiple times, finishing operations are applied in the order specified. Note that, using this type of MFC, unique finishing operations may be specified for each document in the print file.

Inline Resource Group Structure

A resource group begins with the Begin Resource Group (BRG) structured field and ends with the End Resource Group (ERG) structured field. *Inline resources* are included in the inline resource group and can be referred to by name within the print file. They override objects of the same name stored in resource libraries accessed by the print server.Each individual resource begins with the Begin Resource (BR) structured field and ends with the End Resource (ER) structured field. When a resource object is stored in a library, the BRG, BR, ERG, and ER structured fields are not present. When using AFP with a System/390, all structured fields of resource objects included in inline resource groups must be preceded by the X'5A' character. Refer to Appendix D, "System Support Information," on page 183 for information on which resources can be included inline in the system environments supported by AFP. Figure 16 shows the structure of an inline resource group.
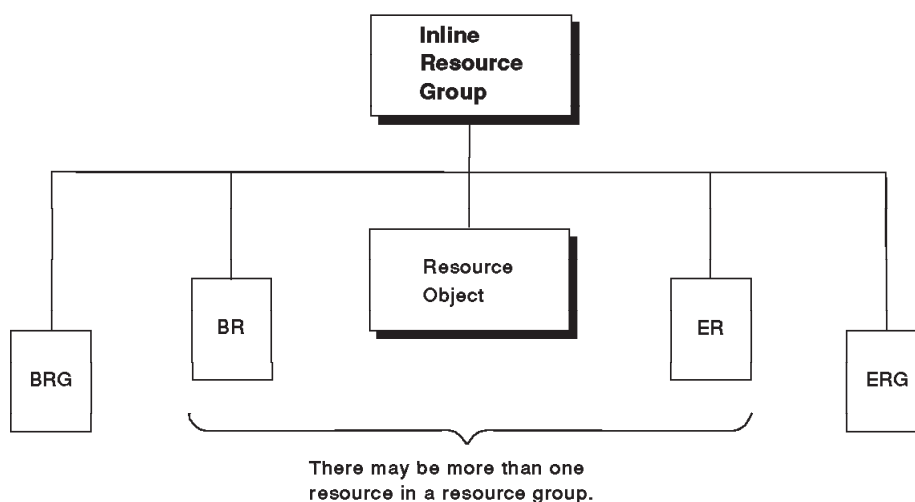
Figure 16. Structure of an Inline Resource Group

The structured fields and objects in an inline resource group are as follows. (Chapter 5, "Structured Fields in a Page Definition and in Line Data" describes the structured fields.)

BRG (Begin Resource Group)
> Begins an inline resource group in the Print File.

BR (Begin Resource)
> Begins a resource object, specifies the resource type, and specifies the name used to select the object for printing.

Resource Object
> A resource object can be one of the following:
> * A page segment
> * An overlay
> * A data object
> * An object container
> * A document
> * A Form Definition
> * A Page Definition
> * A font object (a code page, a font character set, or a coded font)
>
> See the description of the BR structured field in the *Mixed Object Document Content Architecture Reference* for the hexadecimal codes used to identify each type of resource object.

ER (End Resource)
> Ends the resource object. Any name specified in the ER must match the name specified in the BR.

ERG (End Resource Group)
> Ends the inline resource group. Any name specified in the ERG must match the name specified in the BRG.

Note: Not all PSFs support all resource objects in a Resource Group. For a definition of the supported objects, see Appendix D, "System Support Information," on page 183.

Programming Considerations for Inline Resources

Because most resource objects consist of variable-length records, any print file that includes these resources inline must be in variable-length-record format, and must use data records beginning with a carriage control byte.

Invoke Data Map

The IDM selects a new Data Map for printing line data and ends the current line-format page.

Note: When using machine carriage control characters, care must be taken to prevent a blank page from being printed at the start of a document. If the application inserts IDM structured fields following records that have a "skip to channel *nn* immediate" carriage control (X'8B') without making an exception for the start of the document, a blank page will be generated. When the first line data record contains a skip immediate carriage control, a line-format page is started even though there is no data to be printed. When the IDM follows the initial skip immediate carriage control at the start of the document to be printed, the IDM ends the current page, causing a blank page to be printed. When the skip immediate carriage control is used later

| in the document to end the page and it is followed by the IDM structured
| field, a blank page does not occur since the skip immediate carriage control
| has already ended the current line-format page.

- For traditional line data, processing begins with the first Line Descriptor (LND) structured field of the invoked Data Map for the next line-format page.

- For record-format line data, processing begins with the first Record Descriptor
| (RCD) structured field that matches the record ID of the first record processed
| following the IDM.

The IDM structured field can be used to change formatting based on some change in the application data, such as the start of output for a different department or branch office.

The IDM structured field always contains sixteen bytes of information. The Data Map name in the data portion of this structured field must be eight bytes long. If the name of the actual Data Map to be invoked is shorter than eight bytes, trailing blanks must be added.

Sample IDM Structured Field

The Invoke Data Map structured field shown in Figure 17 causes PSF to select Data Map SUMMARY.

The IDM is 16 (X'10') bytes long and has the structured field identifier X'D3ABCA'. In the example, the flag byte is set to X'00' and bytes 6 and 7 contain a sequence number of X'0000'. It is not necessary to number MO:DCA structured fields sequentially, or even to place a meaningful value in the sequence number field. However, for some errors detected by PSF, the sequence number of the structured field in error is printed as part of the error information in the PIMSG data set. This field is reserved in MO:DCA data streams and should be set to zero.

When PSF processes the IDM structured field, the current page is ended. The next record read by PSF begins on a new page, and the information contained in Data Map SUMMARY is used to format subsequent data lines. Use of this structured field assumes the currently active Page Definition contains a Data Map with the name SUMMARY in its Begin Data Map structured field. If no such Data Map exists, an error is generated.

X'5A'	X'0010'	X'D3ABCA'	X'00'	X'0000'	X'E2E4D4D4C1D9E840'

Figure 17. Sample Invoke Data Map Structured Field

Invoke Medium Map

The Invoke Medium Map (IMM) structured field is similar to the IDM structured field except that it causes PSF to select a new Medium Map, or Copy Group, in the current Form Definition at the point where the IMM structured field appears in the print file. PSF ends printing on the current sheet when an IMM is encountered. Note that if the Medium Map specifies the N-up function, the IMM may cause PSF to end printing on the current N-up partition instead of on the current sheet.

The IMM structured field can appear in line-mode, mixed-mode, or fully composed documents. For line-mode or mixed-mode data, processing resumes with the first Line Descriptor (LND) structured field in the Data Map that is active for the next line-format page. When the Data Map contains RCDs, processing resumes with the

first RCD whose Record ID matches the current data record. The IMM structured field is sixteen bytes long and must be coded as shown below.

The IMM structured field can be written by the application when some physical control of the output is required. By using the IMM, the application can offset pages in the data from the medium origin, select paper from the primary or alternate bin, or change between simplex and duplex printing, simply by selecting a Medium Map that contains the desired function.

The functions provided by the IDM and IMM structured fields are the same as those provided by changing Data Maps and Medium Maps with conditional processing in a Page Definition. It is possible to use conditional processing to make the Data Map and Medium Map change without modifying the application to add the IDM and IMM structured fields.

Note that at the beginning of a new composed document and at the beginning of a new set of line-data records, control for presentation is returned to the first Medium Map in the Form Definition. This is shown in Figure 18.

```
Form Definition
   Medium Map M1
   Medium Map M2

Line data records      <presentation controlled by M1>

.
.
.
IMM, Medium Map M2
Line data records      <presentation controlled by M2>

.
.
.
BDT                    <presentation control reverts to M1>
   Composed Pages

.
.
.
   IMM, Medium Map M2  <presentation controlled by M2>

.
.
.
   Composed Pages

.
.
.
EDT
Line data records      <presentation control reverts to M1>

.
.
.
```

Figure 18. Returning Control to First Medium Map in Form Definition

Sample IMM Structured Field

The Invoke Medium Map structured field shown in Figure 19 on page 51 causes PSF to select Medium Map BIN2. (Note that BIN2 contains four trailing blanks to fill out the eight-byte data field.) When PSF processes this structured field, the current page is ended. The next record read by PSF is placed on a new sheet, and the information contained in medium map BIN2 is used. Note that if the Medium Map specifies the N-up function, the next record may be placed on a new partition of the same sheet. If the currently active Form Definition does not contain a

Medium Map with that name in the Begin Medium Map structured field, an error is generated.

X'5A'	X'0010'	X'D3ABCC'	X'00'	X'0000'	X'C2C9D5F240404040'

Figure 19. Sample Invoke Medium Map Structured Field

Using Structured Fields to Skip to a New Page or Sheet

Chapter 3, "Using a Page Definition to Print Data" described the use of conditional processing in a Page Definition to perform a skip-to-new-page or skip-to-new-sheet operation based on a change in the value of a control field in an application data stream. The conditional processing function was added to the Page Definition to provide another way of producing the same output as by imbedding IDM or IMM structured fields in a line-data file to force a new page or sheet. When an IDM or IMM structured field appears in an application data stream, PSF ends the current page and resumes printing at the start of a new page, using the first Line Descriptor in the current Data Map. When the Data Map contains RCDs, printing resumes at the start of a new page using the first RCD whose Record ID matches the current data record.

The data stream shown in Figure 20 provides the same result as the Page Definition shown in Figure 13 on page 33, and the data stream shown in Figure 21 provides the same result as the Page Definition shown in Figure 14 on page 34.

```
Line data records (with carriage control)

X'5A0010D3ABCA000001D5C5E6D7C7404040'
More line data records

⋮
```

Figure 20. Using an IDM Structured Field to Skip to a New Page

---

```
Line data records (with carriage control)

⋮
X'5A0010D3ABCC000001D5C5E6C6D4404040'
More line data records

⋮
```

Figure 21. Using an IMM Structured Field to Skip to a New Sheet

The name of the Data Map invoked in Figure 20 is NEWPG. This is the name on the PAGEFORMAT statement in the PPFA example in Figure 13 on page 33. Re-invoking the same Data Map causes a skip to a new page. It is not necessary to have multiple Data Maps in the Page Definition to achieve this result. Consequently, standard Page Definitions supplied with the print services software can be used with this method.

The same is true of skipping to a new physical sheet. Figure 21 invokes a Medium Map named NEWFM. Even if NEWFM is the current and only Medium Map in the Form Definition, the presence of this structured field causes a skip to a new sheet of paper, or, in the case of N-up presentation, possibly a skip to a new N-up partition.

IMM Structured Fields to Insert a Blank Sheet

Occasionally an application requires that a blank sheet appear between groups of output within a single data set. This blank sheet may be selected from different-color paper loaded in the alternate bin, or it may just be another sheet from the primary bin. The blank sheet is generated by using a Form Definition which specifies the *constant data* function, which allows a sheet to be produced without any variable data on it. To generate the blank sheet, code two consecutive IMM structured fields, as shown in Figure 22.

```
Line data records (with carriage control)

:
:
X'5A0010D3ABCC000000C1D3E3C2C9D54040'
X'5A0010D3ABCC000000D7D9C9C2C9D54040'
More line data records (with carriage control)

:
:
```

Figure 22. Using Two IMM Structured Fields to Force a Blank Sheet

This example assumes a Form Definition with two Medium Maps, as could be built using the PPFA code shown in Figure 23. The first Medium Map coded in the example will be used for the initial pages. They will contain user data (the CONSTANT parameter does not appear in this Medium Map), and are printed on paper selected from the primary bin. When the point is reached where a blank sheet is to be inserted, the application writes out an Invoke Medium Map which selects the second Medium Map. This Medium Map selects a sheet of paper from the alternate bin. No user data is placed on the pages coming from the alternate bin, because CONSTANT FRONT and DUPLEX NO are coded. If the output were to be printed in duplex, CONSTANT BOTH and DUPLEX YES can be coded instead.

Immediately following the IMM structured field to select the second Medium Map (ALTBIN) is a second IMM to return to the original Medium Map (PRIBIN) for the next portion of the data. This set of two consecutive IMM structured fields can be included in the output data stream as often as necessary.

```
FORMDEF BLANKT
 OFFSET 0 0
 REPLACE YES;
COPYGROUP PRIBIN
 DUPLEX NO BIN 1;
COPYGROUP ALTBIN
 CONSTANT FRONT
 DUPLEX NO BIN 2;
```

Figure 23. Form Definition With Two IMMs to Force a Blank Sheet

Variable-Length and Fixed-Length Records

MO:DCA structured fields are variable in length so their lengths can differ. Line data records intermixed with MO:DCA structured fields may also have different lengths. Fully composed MO:DCA documents may consist of records up to 32K bytes long. However variable-length data is not always desirable. Programming requirements may make it preferable to use fixed-length records in some

circumstances. PSF can process a mixed document of fixed-length records even though some of the records contain structured fields with significant information that is much shorter than the data records to be printed. So long as the information in the length portion of the structured field is correct, and the structured field is padded with blanks to the length of the other records in the data set, no errors are generated. The structured fields shown in Figure 24 are all considered valid by PSF in a System/390 environment. The third form, however, may not be supported in a multi-system environment.

X'0010'	X'D3ABCA'	X'00'	X'0000'	PFORMAT1 (Data Map [Page Format] name—8 bytes EBCDIC)	

X'0050'	X'D3ABCA' (Identifier)	X'08' (Flag byte)	X'0000'	PFORMAT1 (Data Map [Page Format] name—8 bytes EBCDIC)	63 bytes of X'00' followed by one byte of X'40'

X'0010'	X'D3ABCA' (Identifier)	X'00' (Flag byte)	X'0000'	PFORMAT1 (Data Map [Page Format] name—8 bytes EBCDIC)	64 bytes of any information to fill the record out to 80 bytes

Figure 24. Three Versions of the Invoke Data Map Structured Field

The first structured field, at the top of Figure 24, is the most common form of Invoke Data Map. The IDM structured field is 16 bytes (X'10') long, so the value X'10' appears in the length field of the introducer. Next is the X'D3ABCA' identifier for Invoke Data Map. The flag byte is zero. The syntax rules for Invoke Data Map indicate that the eight-byte name of the requested Data Map be coded as the data portion of the structured field. This is the rightmost information in the figure.

The second structured field in Figure 24 is 80 bytes long, but here the formal MO:DCA conventions for using padding bytes have been followed. In this example, the flag byte is coded as X'08', which signals that padding bytes appear in the structured field. The padding bytes follow the variable data for the IDM structured field, and the final padding byte is coded as X'40' to signal that 64 padding bytes are present. The length field has been changed from 16 (X'10') to 80 (X'50') to reflect the increased length of the structured field.

The third structured field in Figure 24 is identical to the first, except that the actual MO:DCA data appears as the first 16 bytes in an 80-byte record. This format allows the IDM structured field to be included in a data set of fixed-length 80-byte records, and no errors would result in a System/390 environment.

Of course, fixed-length records that are longer than the number of bytes actually used to contain the MO:DCA structured field information will result in a data set that is larger than one containing variable-length records, each one no longer than necessary. This may be a consideration if the resulting data set is to be sent across a network.

Position and Orientation of Objects

Two coordinate systems are used to position and rotate objects in line data: the page (X_p, Y_p) coordinate system and the text (I, B) coordinate system. The page coordinate system is based on the fourth quadrant of a standard Cartesian coordinate system with the origin in the top-left corner, the X axis increasing from

left to right, and the Y axis increasing from top to bottom. The text (I,B) coordinate system is defined, relative to the page coordinate system, by the text orientation as follows:

Text Orientation	(I,B) Coordinate System
0°,90°	Origin at top-left corner, I increases left to right, B increases top to bottom.
90°,180°	Origin at top-right corner, I increases top to bottom, B increases right to left.
180°,270°	Origin at bottom-right corner, I increases right to left, B increases bottom to top.
270°,0°	Origin at bottom-left corner, I increases bottom to top, B increases left to right.

The coordinate system used depends on the object and how it is included in line data. Table 6 summarizes how objects are positioned and rotated in line data. The table also summarizes how objects are positioned and rotated in MO:DCA data that has been transformed from line data, using the Line Data Object Position Migration (X'27') triplet to capture the text orientation that was active when the line data was presented with a Page Definition. More details on how objects are positioned and rotated is given in the sections that follow the table.

Table 6. Position and Rotation of Objects in Line Data and MO:DCA Data

OBJECTS IN LINE DATA	OBJECTS WITH X'27' TRIPLET IN MO:DCA DATA TRANSFORMED FROM LINE DATA
Page Segment Object	
Page Segment Origin	
(XpsOset,YpsOset) in IPS specify an offset from the current text coordinate system origin (I=0,B=0). The offset is measured using the current text (I,B) coordinate system.	(XpsOset,YpsOset) in IPS specify an offset from the page origin (Xp=0,Yp=0). The offset is measured using the page (X_p,Y_p) coordinate system. The offset was adjusted to include the LND or RCD position.
IM—Image Object in Page Segment	
IM—Image Object Origin	
(XoaOset,YoaOset) in IOC specify an offset from the page segment origin. The offset is measured using the current text (I,B) coordinate system.	(XoaOset,YoaOset) in IOC specify an offset from the page segment origin. The offset is measured using the temporary (X,Y) coordinate system.
IM—Image Object Rotation	
(XoaOrent,YoaOrent) in IOC specify a rotation that is measured with respect to the page (X_p,Y_p) coordinate system X_p-axis.	(XoaOrent,YoaOrent) in IOC specify a rotation that is measured with respect to the page (X_p,Y_p) coordinate system X_p-axis.
IM—Image Cell Origin	
(XCOset,YCOset) in ICP specify an offset from the image object origin. The offset is measured using the current text (I,B) coordinate system.	(XCOset,YCOset) in ICP specify an offset from the image object origin. The offset is measured using the temporary (X,Y) coordinate system.
OCA Object in Page Segment	
OCA Object Origin—OBP Byte 23=X'00'	
(XoaOset,YoaOset) in OBP specify an offset from the page segment origin. The offset is measured using the current text (I,B) coordinate system.	(XoaOset,YoaOset) in OBP specify an offset from the page segment origin. The offset is measured using the temporary (X,Y) coordinate system.
OCA Object Origin—OBP Byte 23=X'01'	

Table 6. Position and Rotation of Objects in Line Data and MO:DCA Data (continued)

OBJECTS IN LINE DATA	OBJECTS WITH X'27' TRIPLET IN MO:DCA DATA TRANSFORMED FROM LINE DATA
(XoaOset,YoaOset) in OBP specify an offset from the page origin (Xp=0,Yp=0). The offset is measured using the page (X_P,Y_P) coordinate system.	(XoaOset,YoaOset) in OBP specify an offset from the page origin (Xp=0,Yp=0). The offset is measured using the page (X_P,Y_P) coordinate system.
OCA Object Rotation—OBP Byte 23=X'00'	
(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the current text (I,B) coordinate system I-axis.	(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the temporary (X,Y) coordinate system X-axis.
OCA Object Rotation—OBP Byte 23=X'01'	
(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the page (X_P,Y_P) coordinate system X_P-axis.	(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the page (X_P,Y_P) coordinate system X_P-axis.
Stand-alone IM—Image Object	
IM—Image Object Origin	
(XoaOset,YoaOset) in IOC specify an offset from the current LND or RCD position. The offset is measured using the current text (I,B) coordinate system.	(XoaOset,YoaOset) in IOC specify an offset from the temporary coordinate system (X=0,Y=0) origin. The offset is measured using the temporary (X,Y) coordinate system. The offset was adjusted to include the LND or RCD position.
IM—Image Object Rotation	
(XoaOrent,YoaOrent) in IOC specify a rotation that is measured with respect to the page (X_P,Y_P) coordinate system X_P-axis.	(XoaOrent,YoaOrent) in IOC specify a rotation that is measured with respect to the page (X_P,Y_P) coordinate system X_P-axis.
IM—Image Cell Origin	
(XCOset,YCOset) in ICP specify an offset from the image object origin. The offset is measured using the current text (I,B) coordinate system.	(XCOset,YCOset) in ICP specify an offset from the image object origin. The offset is measured using the temporary (X,Y) coordinate system.
Stand-alone OCA Object	
OCA Object Origin—OBP Byte 23=X'00'	
(XoaOset,YoaOset) in OBP specify an offset from current LND or RCD position. The offset is measured using the current text (I,B) coordinate system.	(XoaOset,YoaOset) in OBP specify an offset from the temporary coordinate system (X=0,Y=0) origin. The offset is measured using the temporary (X,Y) coordinate system. The offset was adjusted to include the LND or RCD position.
OCA Object Origin—OBP Byte 23=X'01'	
(XoaOset,YoaOset) in OBP specify an offset from the page origin (Xp=0,Yp=0). The offset is measured using the page (X_P,Y_P) coordinate system.	(XoaOset,YoaOset) in OBP specify an offset from the page origin (Xp=0,Yp=0). The offset is measured using the page (X_P,Y_P) coordinate system.
OCA Object Rotation—OBP Byte 23=X'00'	
(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the current text (I,B) coordinate system I-axis.	(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the temporary (X,Y) coordinate system X-axis.
OCA Object Rotation—OBP Byte 23=X'01'	
(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the page (X_P,Y_P) coordinate system X_P-axis.	(XoaOrent,YoaOrent) in OBP specify a rotation that is measured with respect to the page (X_P,Y_P) coordinate system X_P-axis.

Positioning With Respect to Current Descriptor

When objects are included in line data, they occur between line-data records and may be positioned with respect to the inline/baseline position specified by the LNDs or RCDs used to process the records. More precisely, an included object may be positioned with respect to the *current LND*, or *current RCD*. This is also sometimes referred to as the *current line position*, which is defined as follows:

Current LND Position

If the line-data records use ANSI carriage controls, spacing or skipping is performed first and printing of the record is performed last, therefore the current LND is the LND used to process the last record. If the line-data records use machine carriage controls, printing of the record is performed first, and spacing or skipping is performed last. In this case, the current LND is the LND that is spaced to or skipped to, that is, it is the LND that will be used to process the next record. Additionally, if the record is processed as a set of fields using a reuse chain, the current LND is the base LND, that is, the LND that is at the head of the reuse chain. If the current LND does not generate a position, the LND used is the last LND that did generate a position.

Current RCD Position

Because carriage controls are ignored in record-format line data, the current RCD is always the last record RCD that was used to process a data record.

Include Page Segment

The Include Page Segment (IPS) structured field is used to place a page segment resource anywhere on the page. It contains the full eight-character name of the page segment (with trailing blanks if necessary) and the position of the page segment, often referred to as the *page segment origin*. The page segment may be mapped in a Map Page Segment (MPS) structured field in the Active Environment Group (AEG) of the current Data Map, in which case the page segment is downloaded to the printer and may be used multiple times. If it is not mapped, the page segment data is loaded as part of the page.

Objects within the page segment may be positioned with respect to the page segment origin. The page segment inherits the Active Environment Group definition of the including page.

AFP print servers initialize the following PTOCA control sequences as shown prior to processing a text object in an AFP page segment:

Control Sequence	Value
Set Baseline Increment	6 lines per inch
Set Inline Margin	0
Set Intercharacter Adjustment	
	0
Set Text Color	X'FFFF' (printer default color)
Set Text Orientation	0°,90°

The initial print position for text in the page segment is the reference point defined on the including page or overlay coordinate system by the IPS, that is, the page segment origin.

Positioning of Page Segments

Special care must be taken when including page segments in line data to ensure that the objects in the page segment are positioned and oriented properly.

Location of Page Segment Origin

The page segment origin is located on the page as follows:

- If one of the IPS offsets is specified as X'FFFFFF', the page segment origin along that axis is located at the position specified in the current LND or RCD.
- If the IPS offset is not X'FFFFF', the page segment origin is located at the IPS offset measured with respect to the current text coordinate system origin (I=0,B=0), using the current text (I,B) coordinate system. For example, if the text orientation is (90°,180°), the page segment offsets are measured from the top right corner of the page, with the I-axis running from top to bottom and the B-axis running from right to left.
- If the page segment is included with a Resource Object Include (X'6C') triplet on the LND or RCD, the page segment origin is located at the specified offset measured with respect to the position specified in the current LND or RCD, using the current text (I,B) coordinate system.

In summary, the origin of a page segment in line data is always positioned using the text (I,B) coordinate system specified in the current LND or RCD.

Position and Orientation of IM Image Objects in a Page Segment

The image object area offset, as specified in the IOC structured field is measured with respect to the page segment origin, using the text (I,B) coordinate system specified in the current LND or RCD. If the image is celled, the Image Cell Position (ICP) structured field specifies an offset from the image object origin that is measured using the current text (I,B) coordinate system.

The rotation of the IM image is specified in the IOC and is measured with respect to the page coordinate system X_p:-axis (origin is top left corner of page).

Note: For page segments in MO:DCA data, if the IM image is complex (celled), it is recommended that the rotation be set to (0°,90°). For page segments in mixed data, the rotation should be set to match the current text orientation.

Position and Orientation of Image, Graphics, and Bar Code Objects in a Page Segment

If the Object Area Position (OBP) structured field specifies byte 23 (RefCSys) = X'00' (current), the object area offset is measured with respect to the page segment origin, using the text (I,B) coordinate system specified in the current LND or RCD. The object area rotation is measured with respect to the I-axis of the current text (I,B) coordinate system.

If OBP byte 23 = X'01' (page or overlay), the object area offset is measured with respect to the page origin (top left corner of page) using the page coordinate system. The object area rotation is measured with respect to the page coordinate system X:sub.p:esub.-axis (origin is top-left corner of page).

Note: When line data that includes an IPS structured field is transformed into a MO:DCA document by a program such as the IBM AFP Conversion and Indexing Facility (ACIF), the text orientation that was set when the page segment and its objects were positioned must be captured and retained in order to properly position the page segment on the MO:DCA page. This can be done using a Page Segment Positioning Migration (X'27') triplet on the

IPS structured field in the MO:DCA document. For a description of this triplet, see the description of the IPS structured field on page 101.

Sample IPS Structured Field

Figure 25 contains a sample IPS structured field. This example places the segment SIGNAT at the current print position. If the name of the segment were S1SIGNAT, then all eight characters would have to be coded in the IPS structured field.

See the programming tip on page 58 for information on how the current print position is affected by the IPS.

X'5A'	X'0016'	X'D3AF5F'	X'00'	X'0000'	X'E2C9C7D5C1E34040'	X'FFFFFF'	X'FFFFFF'

Figure 25. Include Page Segment Structured Field

> **Programming Tip**
> The current line position is unchanged after the page segment is printed. Additional logic may be needed in the application to place subsequent print lines so that they do not overprint the page segment.

Include Page Overlay

The Include Page Overlay (IPO) structured field functions in a manner similar to Include Page Segment. The IPO structured field specifies the full name of the overlay (any O1 prefix in the overlay name must be included) and the position of the overlay origin. The IPO references an overlay resource that is to be positioned on the page.

The overlay name must appear in the Map Page Overlay structured field of the Active Environment Group of the Data Map currently in effect. The overlay contains its own Active Environment Group definition which specifies the coordinate system for positioning and rotating objects, the size of the overlay, and the names of any fonts used in it. Considerations for the current line position are the same as those discussed in the box on page 58. The current line position is unchanged after the overlay has been placed.

Note: The 3800 printer does not support the IPO function.

Positioning Overlays

Because overlays define their own coordinate system and environment, the rules for positioning an overlay and its objects are somewhat different from those for positioning a page segment and its objects.

Location of Overlay Origin

The overlay origin is located as follows:

- If the IPO offset along either the page X_p-axis or the page Y_p-axis is specified as X'FFFFFF', the overlay origin along that same axis is located by translating the current LND or RCD (I,B) position to an offset along that X_p or Y_p axis.

- If the IPO offset is not X'FFFFFF', the overlay origin is positioned at the specified (X_p, Y_p) offset measured with respect to the page origin (top-left corner of page), using the page coordinate system.
- If the overlay is included with a Resource Object Include (X'6C') triplet on the LND or RCD, the overlay origin is located at the specified offset measured with respect to the position specified in the current LND or RCD, using the current text (I,B) coordinate system.

Orientation of Overlay

If the overlay is included either with an IPO, or with a Resource Object Include (X'6C') triplet on the LND or RCD, the overlay rotation may be specified as 0°, 90°, 180°, or 270°, and is measured with respect to the page coordinate system X_p axis (origin is top-left corner of page). However, the 90°, 180°, 270° rotations of a page overlay are not supported in all AFP environments. Consult the product documentation to see which rotations are supported. Note that the MO:DCA-P IS/1 and IS/2 interchange sets only support 0° rotation of a page overlay.

Position and Orientation of IM Image Object in an Overlay

The image object area offset, as specified in the IOC structured field, is measured using the overlay coordinate system (origin is top-left corner of overlay).

The rotation of the IM image is specified in the IOC and is measured with respect to the overlay coordinate system X-axis (origin is top-left corner of overlay).

Note: If the IM image is complex (celled), AFP print servers require the rotation set to 0°,90°.

Position and Orientation of IO Image, Graphics, and Bar Code Objects in an Overlay

If the Object Area Position (OBP) structured field specifies byte 23 (RefCSys) = X'00' (current) or X'01' (page or overlay), the object area offset is measured with respect to the overlay origin (top-left corner of overlay) using the overlay coordinate system.

The rotation of the OCA object is specified and measured using the overlay coordinate system X-axis (origin is top-left corner of overlay).

Sample IPO Structured Field

A sample IPO structured field appears in Figure 26. It places overlay 01SIGNAT at the current print position on the page.

X'5A'	X'0016'	X'D3AFD8'	X'00'	X'0000'	X'D6F1E2C9C7D5C1E3'	X'FFFFFF'	X'FFFFFF'

Figure 26. Include Page Overlay Structured Field

Include Object

The Include Object (IOB) structured field references an object that is to be positioned on the page. In general, the IOB may be used to include two classes of objects:

- OCA objects (IOCA, GOCA, BCOCA) that specify an Object Environment Group (OEG), or page segments that contains such objects
- Non-OCA paginated presentation objects, such as TIFF images, that are supported by the presentation system

The current AFP support for the IOB in line data is limited to the first class, OCA objects. When referencing an OCA object, the IOB may be used to *override* position, size, orientation, mapping, and default color parameters that are specified in the OEG. When referencing a non-OCA object, the IOB is used to *specify* the position, size, orientation, and mapping parameters for the object.

The RefCSys parameter in the IOB is used to select the coordinate system for positioning and rotating the object area into which the object is mapped:

Value **Description**

X'00' The object area offset in the IOB is measured with respect to the current LND or RCD position, using the current text (I,B) coordinate system. The object area rotation in the IOB is measured with respect to the current text (I,B) coordinate system I-axis.

X'01' The object area offset in the IOB is measured with respect to the page origin ($X_p=0,Y_p=0$), using the page (X_p,Y_p) coordinate system. The object area rotation in the IOB is measured with respect to page (X_p,Y_p) coordinate system X_p-axis.

Including Data Objects Directly in Line Data

Previously it was described how complete AFP resources can be included in the resource group of a print file rather than having to be stored in an external resource library. This is one approach that can be used with applications where many different resources must be included in the print stream, and where it may not be feasible to store these resources externally to the application. However another approach is possible for applications that require large numbers of graphics, images, or bar codes.

One example of such an application is label printing, where many different labels are printed in multiple-up format, each one requiring a unique bar code. Another example is a financial statement application which includes a chart of specific investment performance for each customer. The programming logic required for applications such as these is simpler if each bar code or page segment can be included in the output at the same point as the other data for a given label or statement. Grouping all resources (which can number in the thousands) in an external library or in a resource group at the beginning of the print file may not be practical. In addition, it may be preferable to keep the bar codes, images, or graphics as part of the actual line data for archive purposes. Finally, including them directly in the line data can eliminate the problem of devising unique names for thousands of objects which change each time the program is run.

Graphics, images, and bar codes included with other print data in this manner are not true inline resources, because they do not follow the rules for inline resources described previously. When structured fields that make up graphics, images, or bar codes are included directly in the line data, they provide yet another example of an AFP mixed-mode document.

Including IO Image, Graphics, and Bar Code Objects

Objects that include an Object Environment Group (GOCA, IOCA, and BCOCA objects) can be included directly in a mixed-mode document intermixed with line data so long as the following rules are observed:

- The reference coordinate system (byte 23 of the data field of the Object Area Position [OBP] structured field) must be coded to provide the desired position and rotation of the object on the page:

– If OBP byte 23 (RefCSys) = X'00' (current), the object area offset is measured with respect to the position specified in the current LND or RCD, using the current text (I,B) coordinate system. The object area rotation is measured with respect to the I-axis of the current text (I,B) coordinate system.

– If OBP byte 23 (RefCSys) = X'01' (page or overlay), the object area offset is measured with respect to the page origin, using the page coordinate system (origin is top-left corner of page). The object area rotation is specified in the OBP and is measured with respect to the page coordinate system X_P-axis (origin is top left corner of page).

- If the image or graphic has been built as a page segment, delete the Begin Page Segment and End Page Segment structured fields from the object. The remaining structured fields can be placed in the print stream at the point where the image or graphic should appear.

Including IM Image Objects

Page segments containing IM image data do not have an Object Environment Group, so somewhat different considerations apply to them. Between the BPS and EPS structured fields are the records that provide positioning information for the bits that define the image, and the actual bits themselves in uncompressed form.

Just as for GOCA, IOCA, and BCOCA objects, the positioning information contained in the IOC structured field should be coded to provide the desired placement of the image. Bytes 0 through 5 in the IOC specify the image object area origin for IM images. The offset is measured with respect to the I,B position specified in the current LND or RCD, using the current text (I,B) coordinate system. The image object area offset should be coded as X'000000000000' to position the image at the current LND or RCD. If the image is celled, the Image Cell Position (ICP) structured field specifies an offset from the image object origin that is measured using the current text (I,B) coordinate system.

The rotation of the IM image is specified in the IOC and is measured with respect to the page coordinate system X_P-axis (origin is top-left corner of page).

Note: For page segments in MO:DCA data, if the IM image is complex (celled), it is recommended that the rotation be set to (0°,90°). For page segments in mixed data, the rotation should be set to match the current text orientation.

The Begin Page Segment (X'D3A85F') and End Page Segment (X'D3A95F') structured fields should be deleted. The remaining structured fields can then be placed in the print stream at the point where the image is to appear.

Including Presentation Text Objects

The Presentation Text (PTX) structured field is used to specify text data and the position, rotation, and fonts to be used when presenting text data. The PTX structured field was previously known as Composed Text (CTX), but its identifier of X'D3EE9B' and all its components remain the same. PTX structured fields are made up of control sequences and data. The PTX structured field is described in *Presentation Text Object Content Architecture Reference*, SC31-6803, and provides different functions in the form of control sequences. PTX is probably the most frequently used structured field in fully composed MO:DCA documents. PTX structured fields can be intermixed with line data records so long as a few rules are followed:

- Each PTX structured field should be coded as a self-contained environment. While PTX control sequences can be used to set the line spacing, page margin,

data position, font, etc., these settings remain in effect only for the current PTX structured field. Processing of follow-on line data records or structured fields might change the settings. If a line data record follows a PTX, settings such as its placement and font is determined by the information in the current LND or RCD of the active Page Definition. A PTX can affect the printing of line-data records if it contains text control sequences that change inter-character and inter-word spacing, because these characteristics are not controlled by a Page Definition. If another PTX structured field follows the PTX, the text environment established by the last-used LND or RCD is re-issued before the new PTX is processed. Some presentation systems that convert the mixed-mode data to MO:DCA may also place Begin Presentation Text (BPT) and End Presentation Text (EPT) structured fields around each imbedded PTX. Subsequent processing of the BPT will cause initial text conditions to be set prior to the processing of the PTX. See *Mixed Object Document Content Architecture Reference*, SC31-6802 for more information on initial text conditions set when the BPT is processed.

- Because the print services software considers line-data files to be mapped totally with a Page Definition, PSF generates IPDS commands containing positioning and font information for every record in the file. If a record turns out to be a PTX structured field, the information in the PTX is used to create a subsequent IPDS Write Text command. If a large number of PTX structured fields are included in a line-mode data set, the additional IPDS commands generated by the print services software could add an unacceptable amount of processing overhead when the data set is printed.

- Page Definition information, PTX information, and any additional information contained in objects such as bar code and image placed on the page interact, so the programmer must keep careful track of the page position and fonts in effect as records are written. For example, if the text position, text orientation, or font is not defined in a structured field or object, the values specified in the Page Definition for the current line-data record will be used. Depending on the complexity of the application, it may be easier to write fully composed output rather than using a Page Definition to set up the environment.

Length	X'D3EE9B'	Flag byte	Sequence number	Data

Figure 27. Presentation Text Structured Field

Record Format When Using PTX Structured Fields

When creating a mixed-mode data set which includes PTX structured fields, it is generally easier to use variable-length records. The PTX structured field length ranges up to 32,759 bytes. Much spool space is wasted if every record is padded out to this length, regardless of whether or not the entire 32K bytes contains valid data.

Using the PTX Structured Field

The PTX structured field contains PTOCA data, as defined in the *Presentation Text Object Content Architecture Reference*, SC31-6803. The general format of the PTX structured field is shown in Figure 27. Either of two types of data can follow the PTX structured field introducer:

- The X'2BD3' escape sequence, followed by one or more text control sequences
- "Free-standing text", which is a series of code points representing data to be printed

The first alternative is by far the most common use of PTX. A table of the control sequences that can be used with the PTX structured field appears in Table 7 on page 64.

The PTOCA Architecture groups control sequences into *function sets*, or *subsets*. PT1 is the base subset that is supported by all AFP page printers. PT2 is a superset of PT1 that contains three additional control sequences: Underscore (USC), Overstrike (OVS), and Temporary Baseline Move (TBM). PT3 is a superset of PT2 that contains the Set Extended Text Color (SEC) control sequence for supporting spot colors and process colors in text. See *Advanced Function Presentation Printer Information*, G544-3290, for information on which PTOCA subsets are supported by your printer.

In a PTX structured field, a control sequence immediately follows each X'2BD3' escape sequence. Each control sequence can be coded as *unchained* (even-numbered functions) or *chained* (odd-numbered functions). If unchained controls are used, each one must be preceded by the X'2BD3' escape sequence. In the chained format, each control sequence immediately follows the previous one with no intervening X'2BD3' escape sequence. The last control sequence in a chain must have the even-numbered (unchained) format to signal the end of the chain.

Each text control sequence is a minimum of two bytes long, where the X'2BD3' escape sequence, if present, is not counted as part of the length. The first byte indicates the length of the entire control sequence, including the length byte itself, the function byte, and any parameter bytes. The second byte contains the odd or even function code for the control sequence. A data field ranging from zero to 253 bytes follows.

One reason why free-standing text is seldom used is that one of the PTX control sequences available is *Transparent Data* (TRN), which has a string of code points as its data field, and thereby provides the actual text to be printed. Use of the TRN control sequence allows data whose encoding scheme uses the code points X'2B' or X'D3' to be included in a PTX without having these code points interpreted as an escape sequence.

The usual sequences for placing text on a page are as follows:
- Specify the beginning print position using Absolute Move Inline (AMI) and Absolute Move Baseline (AMB) control sequences
- Select the coded font to be used with the Set Coded Font Local (SCFL) control sequence
- Specify the code points of the text to be printed using a Transparent Data (TRN) control sequence.

Here is an example:
```
X'5A001BD3EE9B0000002BD304D300F004C700B403F10106DAC4C1E3C1'
```

This example begins with a X'5A' carriage control character, as would be required in the System/390 environment. Following this byte in the example is a two-byte length field, which provides the length of the entire record (27, or X'001B'). TheX'5A' character is not included in this count. The next three bytes are the Presentation Text identifier (X'D3EE9B'). Following that is the X'00' flag byte and the two-byte sequence number (X'0000'). The first two bytes of the data are the escape sequence (X'2BD3'), followed by a text control sequence that indicates chaining.

The first control sequence is an Absolute Move Baseline that specifies a baseline offset of X'00F0' logical units from the page origin. For a 240 units-per-inch coordinate system, this indicates an offset of one inch down the page.

The second control sequence is an Absolute Move Inline that specifies an Inline offset of X'B4' or 180 units from the left margin.

Following this is a Set Coded Font Local that selects the coded font that maps to font local ID 1 in the MCF structured field in the Active Environment Group for the Data Map.

The last control sequence is a six-byte-long Transparent Data, which simply contains the word DATA and ends the chaining sequence because it uses the X'DA' (even) function type.

Programming Tip

When deciding how to code Presentation Text structured fields, keep in mind that it is good programming practice to build as long a PTX structured field as possible, to reduce overhead in the print server associated with reading and processing many short records written by the application. Text control sequences should be chained wherever possible. While a string of unchained control sequence pairs will work also, the presence of the X'2BD3' escape sequences can use up many of the 32,759 bytes of the PTX structured field unnecessarily.

Within a fully composed document, the last control sequence in any text object must always indicate end of chaining. If PTX structured fields are intermixed with line data in a mixed-mode document, the last control sequence in the PTX must also indicate end of chaining. This can be accomplished either by specifying an even function type for the last control sequence, or by ending every PTX with a No Operation control sequence with an even function type (X'02F8').

Use of Fonts

Either fixed-pitch or proportionally spaced fonts can be used to present text with the PTX structured field. Positioning of the first character in a string of data contained in the TRN control sequence can be accomplished by preceding the TRN with one of the absolute or relative move text controls, as shown on page 63. If no *move* control sequences follow in the same PTX, data contained in any subsequent TRN controls will be placed immediately following the text in the preceding TRN. Font information stored in the printer is used to ensure that data does not overlap. As a result, it is possible to highlight one word in a string simply by using a Set Coded Font text control. If the PTX record shown on page 63 is extended to print the word DATA a second time in a different font, as in this example:

```
X'...2BD304D3010004C700B403F10106DBC4C1E3C103F10206DAC4C1E3C1'
```

then the resulting output will look like this:

```
DATADATA
```

Table 7. Control Sequences Used in PTX Structured Field

PTOCA Control Sequence Function	Unchained (Even Function)	Chained (Odd Function)
PT1 Control Sequences		

Table 7. Control Sequences Used in PTX Structured Field (continued)

PTOCA Control Sequence Function	Unchained (Even Function)	Chained (Odd Function)
Absolute Move Baseline	04D2	04D3
Absolute Move Inline	04C6	04C7
Begin Line	02D8	02D9
Begin Suppression	03F2	03F3
Draw Baseline Rule	07E6	07E7
Draw Inline Rule	07E4	07E5
End Suppression	03F4	03F5
No Operation	*xx*F8	*xx*F9
Relative Move Baseline	04D4	04D5
Relative Move Inline	04C8	04C9
Repeat String	*xx*EE	*xx*EF
Set Baseline Increment	04D0	04D1
Set Coded Font Local	03F0	03F1
Set Intercharacter Increment	04C2	04C3
Set Inline Margin	04C0	04C1
Set Text Color	0574	0575
Set Text Orientation	06F6	06F7
Set Variable Space Character Increment	04C4	04C5
Transparent Data	*xx*DA	*xx*DB
PT2 Control Sequences		
Overstrike	0572	0573
Temporary Baseline Move	*xx*78	*xx*79
Underscore	0376	0377
PT3 Control Sequences		
Set Extended Text Color	*xx*80	*xx*81

Right-justification and centering of text cannot be done simply by using PTX control sequences. Calculations must be done in the program to place each character at the correct position on the page. This can become fairly complex if proportional fonts are used.

Boxes and Rules

The Draw Baseline Rule and Draw Inline Rule control sequences may be used to draw rules and boxes on the page to highlight information or to separate one area of the output from an adjacent area. The length and thickness of the rule must be specified in the control sequence, and these values are expressed in the units of measure specified in the Presentation Text Descriptor (PTD) structured field. If the rule is to be drawn in the positive baseline or inline direction (that is, from top to bottom or from left to right), the positive number expressing the length and thickness is used. If the rules are to be drawn in the direction opposite the baseline direction ("up" relative to the data on the page) or the direction opposite the inline direction ("backward" relative to the data on the page), the line length or thickness must be coded in two's complement form.

The two's complement of a two-byte hexadecimal number is obtained by inverting each bit of the number and adding a one to the low-order bit position. For example, a one-inch rule is 240 logical units long, or X'F0' L-units, when using 240 units per inch. This value can be placed directly in a Draw Inline Rule control sequence. To obtain the value to use when drawing this rule in the opposite direction, you calculate the two's complement of X'F0' by inverting to get X'FF0F' and then adding X'0001'. The result is X'FF10'. The full, chained control sequence that draws a 3-unit thick rule one inch long in the "backward" direction is X'07E5FF10000300'.

The third and fourth data bytes of the draw rule control sequence specify the thickness of the rule. To determine whether a positive number or the two's complement number is needed, you should decide in which direction to add pels, starting from the initial print position. For inline rules, a positive thickness value adds pels from top to bottom, while a two's complement value adds pels from bottom to top. For baseline rules, pels are added to the right if the thickness value is positive, and to the left if the thickness is expressed as a two's complement (negative) number.

These details come into play when drawing boxes with mitered corners. To make the box outline complete and not have a gap between the end of a baseline rule and the start of an inline rule beneath it, you may have to change the origin point of the rule, the length of the rule, or the rule thickness from positive to negative. Gaps between inline and baseline rules become increasingly visible as the thickness of the rules increase.

Figure 28 illustrates a text-control sequence to draw a box one inch high by two inches wide. The rules that generate the box are four pels thick, so the lengths of the rules in the Draw Rule control sequences have been extended by 4 pels where necessary to make sure the corners are complete.
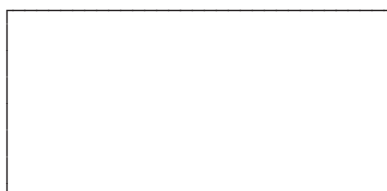
```
... 04C7000F04D300F007E501E000040007E700F0000400...
    < AMI  >< AMB  ><Inline Rule><Baseline Rule>
                    (bottom side) (left side)

... 04C901E004D500F007E5FE20FFFC0007E6FF10FFFC00
    < RMI  >< RMB  ><Inline Rule><Baseline Rule>
                    (top side)    (right side)
```

Figure 28. Text Controls to Draw a Box

Composed Documents

The discussion up to this point has described how line-mode data can be printed in any desired format by using an appropriate Page Definition. Information has also been provided on how formatting can be changed at selected points in the output by using conditional processing or imbedded IDM or IMM structured fields to select a new Data Map or Medium Map for use with subsequent line-data records. This technique of switching among maps in the Page Definition or Form Definition requires advance knowledge of all the output formats that will be used, and the appropriate Data Maps and Medium Maps must be coded. Using this technique also means that formatting can be changed only on a page basis. When a new Data Map or Medium Map is selected, processing of the current page is ended, and the next line-data record appears on the following page. It is not possible to reach an intermediate point on a page and then select a new data map for use in processing the remaining records on the page.

These and other limitations often make it impossible to use external formatting objects to produce the exact kind of output desired for a particular application. If the positioning and formatting needed for each page of your application output is not known in advance, then the application data probably does not lend itself to outboard formatting using a Page Definition. In these cases, you should consider generating fully-composed documents, rather than line-mode or mixed-mode data.

Some examples of applications that have been developed using fully-composed output are:

- Utility bills containing line-by-line details and graphical representations of energy use for each customer compared to the average for all customers
- Insurance policies with clauses, supplements, and detailed client-specific information that vary from one policy to the next
- Financial statements containing sections which describe specific investments, payments, or accounts, each of which vary considerably from one statement to the next. Boxes and shading may be used to highlight certain items of information, and the location of the boxes or shaded areas can be anywhere on the page.

In many cases the text in the output from these applications is also printed using proportional fonts, which can be more difficult to place using a Page Definition than fixed-pitch fonts. Note that in composed documents, the PTX structured fields must be bracketed by Begin Presentation Text (BPT) and End Presentation Text (EPT) structured fields. These structured fields are allowed only in fully composed documents. They cannot be used to bracket a set of PTX structured fields to be included in a line-data file.

For a definition of composed documents, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Programming Options

Software packages are available that can be used to generate fully composed MO:DCA documents. Document Composition Facility (DCF) and Graphical Data Display Manager (GDDM®) are two examples.Such packages, however, are general-purpose formatting programs, and may not be practical for use in high-volume production print applications. The best solution for print requirements of the kinds listed above may be to develop a custom application that produces only the specific output desired. The Advanced Function Presentation Toolbox can be used to develop a customized application in COBOL,

RPG, C, or C++, using calls to routines that perform the actual tasks of generating MO:DCA output. If graphics are needed along with the output, calls to GDDM can be included in a program that uses the AFP Toolbox.

The MO:DCA data stream generated by such an application will be transformed directly into IPDS by the print services program. No optimizing is performed on MO:DCA data. As a result, the application developer should be aware of throughput considerations associated with the MO:DCA structure. Such considerations are highlighted in this chapter in boxes titled "Programming Tip".

Overall Document Structure

A fully composed document will conform to the structure shown on the right side of Figure 30 on page 164. Each document is composed of one or more pages which have the format shown in Figure 32 on page 166. Each page must begin with an Active Environment Group, but the actual objects that appear on the page (text, image, graphics, or bar code) follow the AEG and can appear in any order. The application programmer works with these objects, so an understanding of their format, use, and placement on the page can be helpful when developing an AFP program.

Document Indexing

Indexing and attribute tagging structured fields may be added to documents to permit the selective retrieval of specific pages and page groups for later viewing or printing. The MO:DCA architecture defines six structured fields specifically for this purpose:
• Begin Document Index
• Index Element
• Tag Logical Element
• End Document Index
• Begin Named Page Group
• End Named Page Group

An index is bracketed by Begin Document Index and End Document Index structured fields. It may contain Index Element (IEL) structured fields used to locate objects in a document, and Tag Logical Element structured fields, used to tag pages and page groups with attribute names and their values. Pages in a document may be grouped for indexing using the Begin Named Page Group (BNG) and End Named Page Group (ENG) structured fields.

In AFP environments, the document index is located external to the document.

Document indexes are not used by PSF, but are built for products such as the Viewer function of AFP Workbench, which uses index information to navigate the document.

Programmers are free to include Begin Named Page Group (BNG), End Named Page Group (ENG), and Tag Logical Element (TLE) structured fields in the body of fully composed documents. The print server ignores these structured fields. However, the BNG, ENG, and TLE structured fields are not supported for indexing in a line-data or mixed-data environment.

Document Links

Fully composed MO:DCA documents may contain logical links between document components. An example is a hypertext link from an area on page N that contains a technical term to an area on page M that contains the term's definition. Such links are specified using Link Logical Element (LLE) structured fields. LLE structured fields are not supported in line-data or mixed-data documents. See Appendix D, "System Support Information," on page 183 to find out if the LLE structured field is supported in your environment.

Chapter 5. Structured Fields in a Page Definition and in Line Data

This chapter defines the structured fields used in a Page Definition print control object. It also describes special functions used with certain MO:DCA structured fields when they occur in line-mode or mixed-mode data. Refer to the *Mixed Object Document Content Architecture Reference*, SC31-6802, for definitions of structured fields not included in this manual. Any conflicts arising from definitions in this manual and the *Mixed Object Document Content Architecture Reference* are resolved by the *Mixed Object Document Content Architecture Reference*.

Structured Field Format

Structured fields used in Page Definitions are registered in the MO:DCA architecture and follow the MO:DCA structured field syntax rules. A summary of the syntax rules for Page Definition structured fields follows.

Structured Field Introducer				
Length (2B)	SF Identifier (3B)	Flags (1B)	Reserved X'0000'	Data

Length   A two-byte count field that specifies the length of the structured field. The length value can range from 8 to 32,767. The count includes the length field itself, the other structured field introducer parameters, and the structured field data contents, including padding bytes when present.

SF Identifier   A three-byte value that identifies the type of structured field.

Flag   A one-byte field that specifies the value of optional structured-field indicators.

Bits 0–3 and 5–7 are not used and must be set to zero.

Bit 4 of this byte is a padding indicator. If this bit is on, the structured field data includes padding bytes.

Reserved; X'0000'
The two-byte reserved field is used by some applications to specify a structured field sequence number so that the structured field can be located more easily in a data stream. These applications define the sequence-numbering conventions used. Print-services products do not validate sequence numbering; therefore, applications can use any numbering convention. This field is reserved in MO:DCA data streams and should be set to zero.

Data   Not all structured fields include a data portion. If present, this portion contains specific orders, parameters, and data appropriate for the given structured field. The length of the data field can range from 0 to 32,759.

Padding   If bit 4 in the flag byte (the padding indicator) is set to B'1', padding bytes follow the data field. If padding is indicated, the length of the padding is specified in the following manner:

- For 1 or 2 bytes of padding, the length is specified in the last padding byte.
- For 256 to 32,759 bytes of padding, the length is contained in the last three bytes. The last byte is X'00', and the two preceding bytes specify the padding length.
- For 3 to 255 bytes of padding, the length can be specified by either method.

Note: The length count of the padding data includes the length field itself.

Structured Field Descriptions

The description for each structured field contains the following information:
- Purpose
- Meaning and allowed values of variable parameters
- Contents of constant parameters

Notation Conventions

The bold-faced heading for each structured field contains the following information:
- The three-letter abbreviation
- The three-byte hexadecimal code
- The full name of the structured field

The following conventions apply in the descriptions of the structured fields:
- The byte position of each structured field parameter is given. The first byte of the data field is byte 0
- If the parameter is a variable, the name and function of the parameter is given.
- Each parameter is specified as either *optional* (O) or *mandatory* (M).
- Any valid triplets (see "Structured Field Triplets" on page 73 for a description) listed in the tables are specified as either optional or mandatory. The definition of each triplet follows the description of the structured field in which it appears.
- Any number not preceded by X (hexadecimal) or B (binary) is a decimal number.
- If the parameter is a constant or reserved parameter, only the parameter contents are given; for example, X'0960' or B'001'. A reserved parameter is one that has no meaning at present, but may in the future. Reserved bytes should be set to X'00' by data stream generators and should be ignored by data stream receivers.
- If the same parameter occurs more than once but contains different values, the values in the last parameter are used.
- The tables specify the type of parameter, where appropriate. The parameter type notations are:

  UBIN Unsigned binary integer: a positive integer where the high-order bit may be used as a data bit.

  SBIN Signed binary integer: an integer where the high-order bit is the sign (plus or minus) of the integer and cannot be used as a data bit.

  BITS Bit String: a string consisting solely of bits; binary flags where each binary bit can be assigned a specific meaning.

CHAR

Character encoding: a string of one or more code points for alphanumeric characters. For example, X'C1C2C3F1' in EBCDIC can represent ABC1.

CODE Architected value: a code assigned to a specific item.

Structured Field Triplets

Several structured fields contain self-identifying parameters called *triplets* in their data field. A triplet contains three components: the triplet length, the triplet identifier, and the triplet values. See Table 8.

Table 8. Structured Field Triplet Syntax

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Triplet Length	3–254	Specifies the length of the triplet, including this byte	M
1	CODE	Triplet Identifier		Identifies the triplet	M
2–*n*		Triplet Data		Contains the data for this triplet	M

External Resource Object Naming Conventions

MO:DCA-P objects can be named using one of the following two formats:

- Token name. This name is specified using a fixed-length 8-byte parameter on Begin, Invoke, Map, and Include structured fields

- Fully qualified name. This name can be up to 250 bytes long and is specified using the Fully Qualified Name (FQN) X'02' triplet on Begin, Map, and Include structured fields, as well as on object-processing structured fields. For names, the FQNFmt parameter on this triplet is set to X'00' to specify a character string format, and the FQNType parameter specifies how the name is used. When a fully qualified name is specified using FQNType X'01' on a Begin structured field, it overrides any token name that may have been specified on the structured field.

MO:DCA-P object names are encoded using the code page and character set specified in a Coded Graphic Character Set Global ID X'01' triplet, except in those cases where the name defines a fixed encoding. The X'01' triplet can specify the encoding in two forms; use of the Coded Character Set Identifier (CCSID) form is recommended. For a definition of the X'01' triplet see *Mixed Object Document Content Architecture Reference*.

The X'01' triplet may be specified on most MO:DCA structured fields that contain character data such as an object name. Careful specification of code page and character set is essential for interchange since the system defaults for code page and character set may vary from one system environment to another.

In AFP environments, print servers treat the object name—other than TrueType and OpenType full font name—as an external resource library member name and attempt to process a resource library member with the same name. This means that the external names are subject to the system imposed file naming rules.

To ensure portability across all AFP platforms, external object names other than TrueType and OpenType full font names must be composed according to the following conventions:

- Names consist only of the following characters: A–Z, 0–9, $, #, @. When the object name is specified using the fixed-length 8-byte token name parameter, a trailing space character (X'40' in the EBCDIC encoding) or a trailing null code point (X'00') is assumed to terminate the name.

- To ensure portability across older versions of print servers that do not support encoding definitions in the X'01' triplet, names should use only the recommended characters and be encoded in EBCDIC using code page 500 and a character set that includes the above-mentioned characters. The preferred character set is 961, which includes only those characters, however character sets such as 697, which contain additional characters, are also appropriate. With this encoding, the code points for the characters are:

  A–I (code points X'C1'–X'C9'),
  J–R (code points X'D1'–X'D9'),
  S–Z (code points X'E2'–X'E9'),
  0–9 (code points X'F1'–X'F9'),
  $, #, and @ (code points X'5B', X'7B', and X'7C', respectively)

  Note that such older print servers normally assume this EBCDIC encoding as the default encoding for the document. This EBCDIC encoding can be identified with CCSID 500, which represents the combination of code page 500 and character set 697.

TrueType and OpenType full font names specified in the MDR structured field are not restricted to these characters and may be encoded as required by the AFP-generating application. However, since these names are used to search inline font containers and Resource Access Tables (RATs) which use a fixed UTF-16BE encoding for full font names, efficiency is gained if the full font names in the MDR are also encoded in UTF-16BE. This avoids an encoding conversion. The UTF-16BE encoding can be identified with CCSID 1200. This encoding needs to be specified with a X'01' triplet on the MDR that specifies the full font name.

Begin and End Structured Fields

Begin structured fields identify the beginning of an object in a print data stream or of a data stream resource. All Begin structured fields are used in conjunction with corresponding End structured fields to delimit specific objects in a document, page, or resource.

When a Begin data object structured field (BPT, BGR, BIM, or BBC) is encountered in a page, an overlay, or a page segment, the initial data presentation conditions are set from values specified in the data descriptor structured field for the object, prior to processing the object data. If no initial conditions are specified in the object data descriptor, or if null values are specified, then the initial conditions are set to the system default. If no system default is defined, the device default is used.

Any triplet parameter encountered on any of the Begin structured fields that is not explicitly defined to be valid in the structured field definition is ignored. The document presenter assumes that these fields are informational although they may have meaning in other applications.

Begin Data Map (BDM)

The Begin Data Map structured field begins a Data Map resource object.

BDM (X'D3A8CA') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8CA'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	DMName		Name of the Data Map	M
8	CODE	DatFmt	X'00'–X'02'	Data formatting specified by this Data Map **X'00'** Data Map formats traditional line data using LNDs **X'01'** Data Map formats line data containing record format IDs using RCDs **X'02'** Data Map formats XML data containing start and end tags using XMDs	O
9–n		Triplets		See "BDM Semantics" for triplet applicability.	M

BDM Semantics

DMName The token name of the Data Map. This is a mandatory parameter because an Invoke Data Map (IDM) structured field selects a Data Map by specifying its token name.

DatFmt An optional parameter that determines how the Data Map is used to format line data. If this parameter is not specified, the architected default is X'00'.

Value	Description
X'00'	The Data Map contains LNDs and is used to format traditional line data that may contain CCs and TRCs.
X'01'	The Data Map contains RCDs and is used to format line data that contains record format IDs and may also contain CCs.
X'02'	The Data Map contains XMDs and is used to format XML data that contains start and end tags.

Notes:

1. If a triplet is included on this structured field, the optional positional parameters become mandatory.
2. If one of the Data Maps in a PageDef contains LNDs, then all of the Data Maps in a PageDef must be LND based.
3. If one of the Data Maps in a PageDef contains RCDs, then all of the Data Maps in a PageDef must be RCD based and subpages are not used (the Data Map contains only one subpage).

4. If one of the Data Maps in a PageDef contains XMDs, then all of the Data Maps in a PageDef must be XMD based and subpages are not used (the Data Map contains only one subpage).

BDM Triplets

Encoding Scheme ID (X'50') Triplet

This triplet is an optional triplet when used with DatFmt X'00' (formatting with LNDs) and X'01' (formatting with RCDs) but it is mandatory when used with DatFmt = X'02' (formatting with XMDs). It is used to specify the encoding scheme associated with the user data, the XML Name (X'8A') triplet specified on XMDs, the Record Descriptor ID field specified on RCDs, the Field Delimiter specified on RCDs and XMDs, the Fixed text specified in the Fixed Data Text (FDX) structured field, and the Comparison String field of the Conditional Processing Control (CCP) structured fields.

The values supported in the ESidUD field of the Encoding Scheme ID triplet when formatting data with a Page Definition are:
- X'6100': EBCDIC Presentation SBCS
- X'2100': PC Data SBCS (ASCII)
- X'7807': UTF-8
- X'7200': UTF-16

When this triplet is specified for LND or RCD processing, it is used to determine if searching for the Byte Order Mark (BOM) is necessary. If the triplet specifies UTF-8 or UTF-16, the first bytes (following any CC or TRC) of the first line or record of the line data are examined to see if the BOM has been placed in the data. If the BOM is in the data, it is removed so it is not considered part of the user data.

For XMD processing, this triplet is mandatory and the first bytes of the first line or record of the print file are always examined to see if the BOM is in the data.

For more information about BOM processing, see "Unicode Line Data" on page 12.

The Encoding Scheme ID Triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Notes:

1. ESidUD is required for Data Maps that are to print XML data.
2. This triplet may occur once. If this triplet is specified more than once, only the first is used.
3. Each Encoding Scheme ID triplet specified on each BDM structured field of a single Page Definition must specify the same encoding.
4. This Encoding Scheme ID triplet overrides the encoding specified in the XML data.
5. Except for certain situations in processing XML data with FOCA fonts (see "XML Data" on page 13), the font selected to print the data must match the encoding of the user data specified in this triplet.

Margin Definition (X'7F') Triplet

This is an optional triplet that is used only if DatFmt = X'01' (formatting with RCDs) or X'02' (formatting with XMDs). This triplet may occur once. If this triplet is specified more than once, only the first is used. It is used to specify the page

margins for the Data Map. These margins are used for logical page eject processing and for graphics processing. If this triplet is specified on a Data Map that contains LNDs, it is ignored.

Triplet X'7F' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	14	Length of the triplet, including Tlength	M
1	CODE	Tid	X'7F'	Identifies the Margin Descriptor triplet	M
2–5	CODE	TxtOrent	X'0000 2D00', X'2D00 5A00', X'5A00 8700', or X'8700 0000'	Text (I,B) Orientation: **X'0000 2D00'** 0,90 degrees **X'2D00 5A00'** 90,180 degrees **X'5A00 8700'** 180,270 degrees **X'8700 0000'** 270,360 degrees	M
6–7	UBIN	LeftMar	0 to page extent minus 1	Left Margin Offset from page edge	M
8–9	UBIN	TopMar	0 to page extent minus 1	Top Margin Offset from page edge	M
10–11	UBIN	RightMar	0 to page extent minus 1	Right Margin Offset from page edge	M
12–13	UBIN	BotMar	0 to page extent minus 1	Bottom Margin Offset from page edge	M

Triplet X'7F' Semantics:

TxtOrent Text Orientation. Specifies the text orientation that is used to fix the page margins for this Data Map. See Figure 29 on page 79.

The margins are fixed for the Data Map; that is, they do not change when the text orientation changes in the Data Map. For example, if this parameter specifies a (0,90) degree text orientation, the top-left diagram in Figure 29 on page 79 shows how the LeftMar, TopMar, RightMar, and BotMar parameters define the left, top, right, and bottom margins, respectively. Once specified, these margins define a bounding box for the Data Map that is indicated by the dashed lines.

Note that if the text orientation is changed within the Data Map, the same bounding box applies to the new orientation, but the name of the margins change in the new orientation. For example, if the new text orientation is (90,180) degrees, as shown in the top-right diagram of Figure 29 on page 79, the left margin in the new orientation is actually defined by the TopMar parameter, the bottom margin is defined by the LeftMar parameter, and so on. Therefore, when a possible baseline overflow is evaluated in the new orientation, the print position is actually checked against the LeftMar parameter, which is the bottom margin for that text orientation.

Similar processing occurs for graphics that are generated by an RCD or XMD. If the text orientation for the graphics is (90,180) degrees, the bottom margin is defined by the LeftMar parameter. Therefore, active lines or boxes that are ended at the bottom margin are actually ended at the margin defined by the LeftMar parameter, which is the bottom margin for that text orientation.

Begin Data Map (BDM)

LeftMar	Left Margin. Specifies the offset of the left margin along the i axis from the left edge of the page. The left edge of the page is the zero position on the i axis.
TopMar	Top Margin. Specifies the offset of the top margin along the b axis from the top edge of the page. The top edge of the page is the zero position on the b axis.
RightMar	Right Margin. Specifies the offset of the right margin along the i axis from the right edge of the page.
BotMar	Bottom Margin. Specifies the offset of the bottom margin along the b axis from the bottom edge of the page.

If this triplet is not specified, the default is a text orientation of (0,90) degrees, and all margin offsets set to X'0000'.
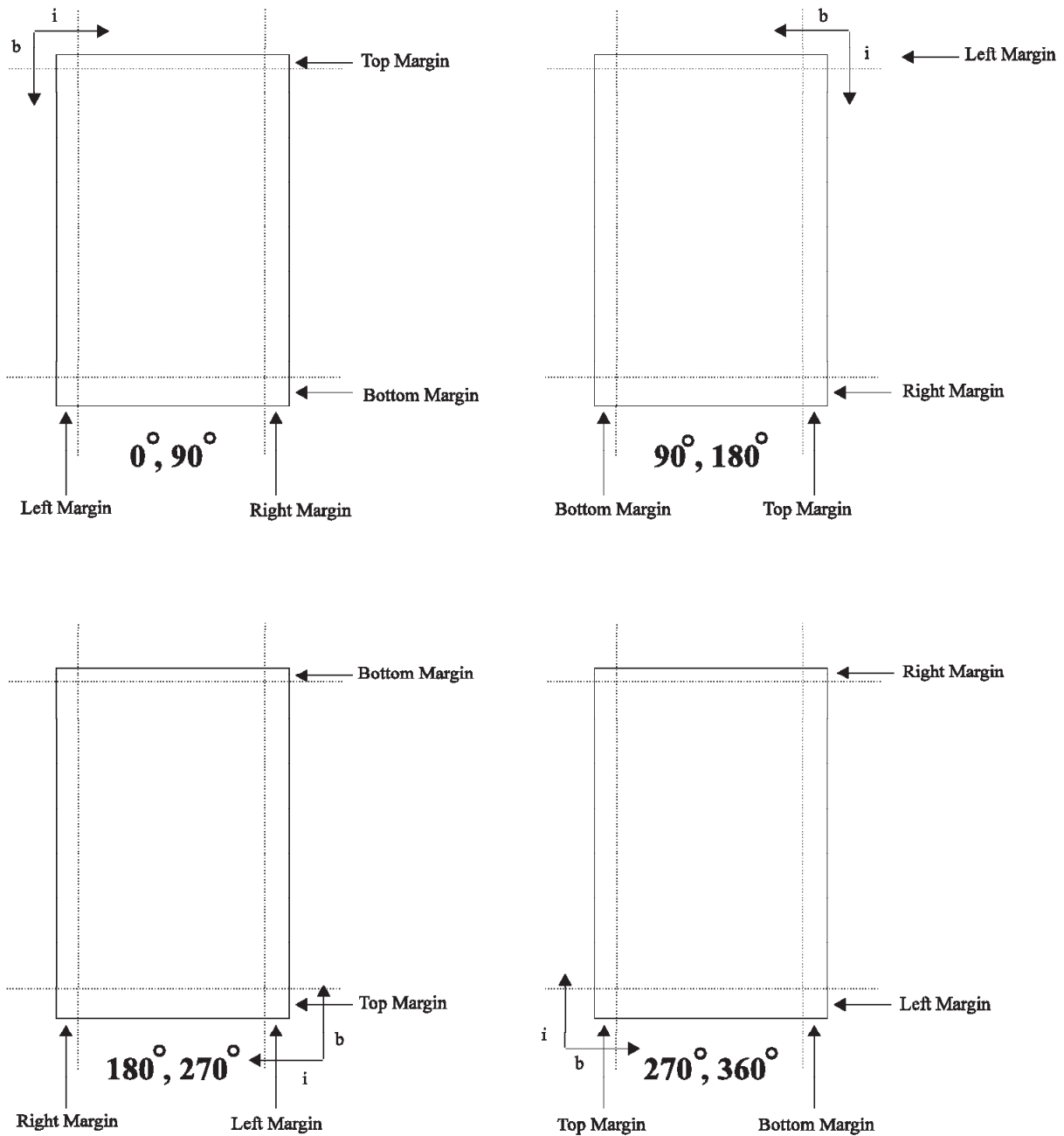
Figure 29. Relationship of Margin Definition to Text Orientation

Page Count Control (X'7C') Triplet

This is an optional triplet that may occur once. If this triplet is specified more than once, only the first is used. It is used only if DatFmt = X'01' (formatting with RCDs) or X'02' (formatting with XMDs). It is used to specify how the page count is initialized and maintained for the active Data Map. If this triplet is specified on a Data Map that contains LNDs, it is ignored.

Triplet X'7C' Syntax:

Begin Data Map (BDM)

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M
1	CODE	Tid	X'7C'	Identifies the Page Count Control triplet	M
2–3	CODE	PageNum	X'0001'–X'FFFF'	Initial page number	M
4			0	Reserved; must be zero	M
5	CODE	CountCtr	X'00'–X'03'	Page count control for Data Map: **X'00'** Stop **X'01'** Resume **X'02'** Continue **X'03'** Reset	M
6	BITS	CountFlgs		Bits that specify additional page count controls. See "Triplet X'7C' Semantics" for bit definitions.	M

Triplet X'7C' Semantics:

PageNum   Initial page number. Specifies the initial page number to be set into the page count when the page count control specifies X'01' (Resume), or X'03' (Reset).

CountCtr   Page count control. Specifies how the page count is initialized and maintained for the active Data Map.

Value   **Description**

X'00'   Stop. When this Data Map is invoked, the page count is initialized to the last page number used with the previous Data Map, which is the current page number. If there is no current page number, it is initialized to the value specified by PageNum. Once the page count is initialized, it is *not* incremented for the duration of this Data Map.

X'01'   Resume. On the *first* invocation of this Data Map, the page count is initialized to the value specified by PageNum. On each successive invocation of this Data Map, the page count is initialized to the last page number used on the previous invocation of this Data Map. Once the page count is initialized, it is incremented for every page presented with this Data Map.

X'02'   Continue. When this Data Map is invoked, the page count is initialized to the last page number used with the previous Data Map, which is the current page number. If there is no current page number, such as when this is the first Data Map invoked for the job, it is initialized to the value specified by PageNum. Once the page count is initialized, it is incremented for every page presented with this Data Map.

X'03'   Reset. When this Data Map is invoked, the page count is initialized to the value specified by PageNum. Once the page count is initialized, it is incremented for every page presented with this Data Map.

All others
    Reserved

CountFlgs Bits that specify additional page count controls.

  Bit 0 Count control for MO:DCA pages:

    Value **Description**

    B'0' Do not count MO:DCA pages that occur in mixed-mode data.

    B'1' Count MO:DCA pages that occur in mixed-mode data.

  Bit 1 Count control for constant pages, that is, pages that contain no variable data. Such pages are generated when:
    • .The Formdef specifies "constant form".
    • The Formdef specifies N-up and no variable data is allowed on

    Value **Description**

    B'0' Do not count constant pages.

    B'1' Count constant pages.

  Bits 2–7
    Reserved; all bits must be B'0'.

If this triplet is not specified, the defaults are CountCtr = X'02' (Continue), and CountFlgs = B'00' (do not count MO:DCA pages and constant pages that occur in mixed-mode data).

Begin Data Map Transmission Subcase (BDX)

The Begin Data Map Transmission Subcase structured field begins a Data Map Transmission Subcase object, which contains the structured fields used to map lines of data to the page.

BDX (X'D3A8E3') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8E3'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	DMXName		Name of the Data Map Transmission Subcase	O

BDX Semantics

DMXame   The token name of the Data Map Transmission Subcase. This is an optional parameter.

Begin Page Map (BPM)

The Begin Page Map structured field begins a Page Map resource object, also called a *Page Definition* or *PageDef*. A Page Definition is a print control resource object used to compose line data into pages for printing on page printers.

BPM (X'D3A8CB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8CB'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	PMName		Name of the Page Map	O

BPM Semantics

PMName   The token name of the Page Map. This is an optional parameter.

Conditional Processing Control (CCP)

The Conditional Processing Control structured field defines tests to be performed on selected input records in line data and specifies the actions to take based on the test results. This optional structured field is selected with LND, RCD or XMD structured fields in the Page Definition. An LND, RCD or XMD can have a unique CCP associated with it or it can reference a CCP that has already been used. In either case, the CCP is referenced with the *CCPID* field of the LND, RCD or XMD. If a CCP structured field is included in a Page Definition, it must appear before the Data Maps in the Page Definition.

CCP (X'D3A7CA') Syntax

Structured Field Introducer				Structured Field Data
SF Length (2B)	ID = **X'D3A7CA'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–1	CODE	CCPid	X'0001'–X'FFFF'	CCP Identifier	M
2–3	CODE	NxtCCPid	X'0001'–X'FFFF'	Next CCP Identifier	M
4	BITS	CCPFlgs			M
Bit 0			B'0'–B'1'	Before subpage actions	
Bit 1			B'0'–B'1'	After subpage actions	
Bit 2			B'0'–B'1'	Spacing actions	
Bits 2–7			B'0000'	Reserved	
5			X'00'	Reserved	M
6–7	UBIN	NumRGs	X'0001'–X'FFFF'	Number of repeating groups	M
8–9	UBIN	RGLgth	X'0015'–X'FFFF'	Length of each repeating group	M
10–11	UBIN	CSLgth	X'0000'–X'FFFF'	Length of comparison string	M
12–*n*		Repeating groups		One or more repeating groups	M

CCP Semantics

CCPid   CCP Identifier. CCPs can be chained to handle complex data within multiple CCP records. If this is the first or only CCP, this field matches the CCP Identifier in *CCPID* field of the LND, RCD or XMD. Subsequent CCPs in a chain have unique identifiers.

NxtCCPid   Next CCP Identifier. Contains the identifier of the next CCP to be processed. A value of zero indicates that this is the last or only CCP to process.

CCPFlgs   Conditional Processing Flags.

Bit 0   If B'1', this CCP requires action before a subpage boundary.

Bit 1   If B'1', this CCP requires action after a subpage boundary.

Bit 2   Provides spacing action at the top of a page when the following conditions are present:
- A new page is started by a true condition.

Conditional Processing Control (CCP)

- Conditional processing is active.
- ANSI control characters are used.

If B'0', space according to the ANSI carriage control value. If B'1', suppress spacing; print only.

Bits 3–7
Reserved; must be B'000000'.

Note that when CCPs are used with RCDs to process record-format line data or XMDs to process XML data, the whole page is the only sub-page, therefore timing actions that specify a subpage are processed against the complete page.

NumRGs   Number of Repeating Groups. Indicates the number of repeating groups for this CCP. The value must be greater than zero.

RGLgth   Length of each repeating group. Indicates the length of the repeating groups to follow. The length must be at least 21 bytes.

CSLgth   Length of Comparison String. Indicates the length of the text string in the Comparison String parameter within the repeating group.

Each repeating group of the CCP contains action information. See Table 9 for the definitions of the CCP repeating groups.

Table 9. CCP Repeating Group Structure

Bytes	Parameter Name	Type	Description and Values	
0	Timing of Action	UBIN	0	Take default action. (The default is for action to be taken immediately before presenting current line.)
			1	Take action immediately before presenting current line.
			2	Take action before presenting current subpage.
			129	Take action immediately after presenting current line.
			130	Take action after presenting current subpage.
1	Medium Map Action	UBIN	0	Ignore
			1	Continue using current medium map with page eject
			2	Invoke named Medium Map
			3	Invoke first Medium Map
			4	Invoke next Medium Map
2–9	Medium Map Name	CHAR	Any 8-byte value	
10	Data Map Action	UBIN	0	Ignore
			1	Continue using current Data Map with page eject
			2	Invoke named Data Map
			3	Invoke first Data Map
			4	Invoke next Data Map.
11–18	Data Map Name	CHAR	Any 8-byte value	

Table 9. CCP Repeating Group Structure (continued)

Bytes	Parameter Name	Type	Description and Values	
19	Comparison	UBIN	0	Any change
			1	Equal to
			2	Less than
			3	Equal to or less than
			4	Greater than
			5	Equal to or greater than
			6	Not equal
			7	Take the action without comparison
20–*nnn*	Comparison String	CHAR	1 to *nnn* bytes, where *nnn* plus the total length of the fixed-length fields in the CCP is less than, or equal to, 32759 bytes.	

Byte 0   Timing of Action. This parameter indicates when the action specified for the CCP is to be taken. Only values of 0, 1, 2, 129, and 130 are allowed.

Byte 1   Medium Map Action. This parameter indicates what action is to be taken for the Medium Map when the conditional processing test for a comparison field is true.

Bytes 2–9   Medium Map Name. If the Medium Map Action parameter is 2—Invoke named Medium Map, these 8 bytes contain the name of the Medium Map to be used.

Byte 10   Data Map Action. This parameter indicates the action to be taken for the Data Map when the conditional processing test for a comparison field is true.

Bytes 11–18   Data Map Name. If the Data Map Action parameter is 2—Invoke Named Data Map, these 8 bytes contain the name of the Data Map to be used.

Byte 19   Comparison. This one-byte parameter indicates the type of comparison between the input data and the comparison string (bytes 20–*nnn*).

Value 0 ("any change") specifies that the contents of the comparison field are to be compared with the contents of the comparison field in the last preceding record that was checked using the current CCP structured field. An "any change" comparison is true when the contents of the comparison field have changed from one record being checked to the next. If the field lies outside the boundary of the current input record, which may occur with variable-length records or with truncated trailing blanks, the comparison is false, and the current records are not used in future comparisons.

An "any change" comparison is always false the first time the CCP structured field is used. Whenever a new Data Map is invoked, all comparisons are reset, and comparison field values in the previous Data Map are not retained.

Byte 20–*nnn*   Comparison String. This variable-length parameter indicates the text string against which a comparison test is to be performed, if the Comparison parameter has a value between 1 and 6.

The length of the text string is determined by a value contained in bytes 10–11 of the CCP structured field.

Conditional Processing Control (CCP)

> **Note:** To be able to match this Comparison String to input data, the encoding of the text specified in this parameter must match the encoding of the input data.

Data Map Transmission Subcase Descriptor (DXD)

The Data Map Transmission Subcase Descriptor structured field is supported only for migration purposes.

DXD (X'D3A6E3') Syntax

Structured Field Introducer				Structured Field Data
SF Length (2B)	ID = **X'D3A6E3'**	Flags (1B)	Reserved X'0000'	

Offset	Type	Name	Range	Meaning	M/O
0–3		ConData		Constant data	M

DXD Semantics

ConData Constant data. Must be set to X'0001 00FF', not checked.

End Data Map (EDM)

The End Data Map structured field terminates the Data Map object initiated by a Begin Data Map structured field.

EDM (X'D3A9CA') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9CA'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	DMName		Name of the Data Map	O

EDM Semantics

DMName   The token name of the Data Map being terminated. If a name is specified, it must match the name in the most recent Begin Data Map structured field. If the first two bytes of this parameter contain the value X'FFFF', the name matches any name specified on the corresponding Begin Data Map structured field.

End Data Map Transmission Subcase (EDX)

The End Data Map Transmission Subcase structured field terminates the Data Map Transmission Subcase initiated by a Begin Data Map Transmission Subcase structured field.

EDX (X'D3A9E3') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9E3'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	DMXName		Name of the Data Map Transmission Subcase	O

EDX Semantics

DMXame   The token name of the Data Map Transmission Subcase being terminated. If a name is specified, it must match the name in the most recent Begin Data Map Transmission Subcase structured field. If the first two bytes of this parameter contain the value X'FFFF', the name matches any name specified on the corresponding Begin Data Map Transmission Subcase structured field.

End Page Map (EPM)

The End Page Map structured field terminates the Page Map object initiated by a Begin Page Map structured field.

EPM (X'D3A9CB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9CB'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	PMName		Name of the Page Map	O

EPM Semantics

PMName   The token name of the Page Map being terminated. If a name is specified, it must match the name in the most recent Begin Page Map structured field. If the first two bytes of this parameter contain the value X'FFFF', the name matches any name specified on the corresponding Begin Page Map structured field.

Fixed Data Size (FDS)

The Fixed Data Size structured field specifies the number of bytes of text found in the following Fixed Data Text (FDX) structured fields.

FDS (X'D3AAEC') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AAEC'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–1	UBIN	TxtLgth	1–65535	Number of data bytes in following FDX structured fields	M

FDS Semantics

TxtLgth   The number of bytes of text in the FDX structured fields that immediately follow. If no fixed data text exists in this Data Map Transmission Subcase, the FDS and FDX structured fields should not be specified.

Fixed Data Text (FDX)

The Fixed Data Text structured field contains text that can be selected and presented with LND, RCD or XMD structured fields in the Page Definition. This text is used when flag bit 7 of the LND, RCD or XMD is set to B'1'. Any number of FDX structured fields can appear, but the total number of data bytes must match bytes 0–1 of the Fixed Data Size (FDS) structured field. The output should fit on the page, and the fit can be affected by the size of the font used.

The *DataStrt* and *DataLgth* fields of the LND, RCD or XMD specify the fixed data offset and length.

FDX (X'D3EEEC') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3EEEC'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–*n*	CHAR	Text		Fixed text to be added	O

FDX Semantics

Text Code points of the fixed text to be added to the page. From 0 to 32,743 bytes may be specified.

Invoke Data Map (IDM)

The Invoke Data Map structured field selects a new Data Map for printing line data and ends the current line-format page. With LND Data Maps, processing begins with the first Line Descriptor (LND) structured field of the invoked Data Map for the next line-format page. With RCD Data Maps, processing begins with the first Record Descriptor (RCD) structured field that matches the Record ID of the current line-data record. With XMD Data Maps, processing begins with the first XML Descriptor (XMD) structured field that matches the current Qualified Tag.

IDM (X'D3ABCA') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABCA'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	DMName		Name of the new Data Map in the Page Definition.	M

IDM Semantics

DMName   The token name of the new Data Map in the currently active Page Definition. This name must match the name on the Begin Data Map (BDM) structured field. If the name is shorter than eight bytes, trailing blanks must be added.

Include Object (IOB)

Notes:

1. The IOB is a MO:DCA structured field. The following description documents its use in line-mode and mixed mode applications and introduces parameter values that are only valid in these environments. For the formal definition of the IOB structured field, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

2. When processing XML data, the IOB may only be used in a Page Definition resource.

An Include Object structured field references an object and optionally contains parameters that identify the object and that specify presentation parameters such as object position, size, orientation, mapping, and default color. Where the presentation parameters conflict with parameters specified in the object's environment group (OEG), the parameters in the Include Object structured field override. If the referenced object is a page segment, the IOB parameters override the corresponding environment group parameters on all data objects in the page segment.

IOB (X'D3AFC3') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AFC3'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	ObjName		Name of the object	M
8				Reserved; must be zero	M
9	CODE	ObjType	X'5F', X'92', X'BB', X'EB', or X'FB'	Object type: **X'5F'** Page Segment **X'92'** Other object data **X'BB'** Graphics (GOCA) **X'EB'** Bar Code (BCOCA) **X'FB'** Image (IOCA)	M
10–12	SBIN	XoaOset	−32768 – +32767	X-axis origin of the object area	M
			X'FFFFFF'	Use the X axis origin defined in the object (not supported if RefCSys=X'00')	
13–15	SBIN	YoaOset	−32768 – +32767	Y-axis origin of the object area	M
			X'FFFFFF'	Use the Y-axis origin defined in the object (not supported if RefCSys=X'00')	

Offset	Type	Name	Range	Meaning	M/O
16–17	CODE	XoaOrent	X'0000', X'2D00', X'5A00', or X'8700'	The object area's X-axis rotation from the X axis of the reference coordinate system: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	M
			X'FFFF'	Use the X-axis rotation defined in the object (not supported if RefCSys=X'00')	
18–19	CODE	YoaOrent	X'0000', X'2D00', X'5A00', or X'8700'	The object area's Y-axis rotation from the X axis of the reference coordinate system: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	M
			X'FFFF'	Use the Y axis rotation defined in the object (not supported if RefCSys=X'00')	
20–22	SBIN	XocaOset	−32768 − +32767	X-axis origin for object content	M
			X'FFFFFF'	Use the X-axis origin defined in the object	
23–25	SBIN	YocaOset	−32768 − +32767	Y-axis origin for object content	M
			X'FFFFFF'	Use the Y-axis origin defined in the object	
26	CODE	RefCSys	X'00'–X'01'	Reference coordinate system: **X'00'** Text (I,B) coordinate system defined by current LND, RCD or XMD **X'01'** Page or overlay coordinate system	M
27–*n*		Triplets		See the *Mixed Object Document Content Architecture Reference*, SC31-6802, for triplet applicability.	M

IOB Semantics

For a complete definition of the IOB semantics, see the *Mixed Object Document Content Architecture Reference*, SC31-6802. The following describes parameter values that are unique to the IOB when used in line-mode or mixed-mode environments.

XoaOset   If RefCSys = X'01', this parameter specifies the offset along the X axis, X_{pg} or X_{ol}, of the including page or overlay coordinate system to the origin of the X axis, X_{oa}, of the object area coordinate system. The value for this parameter is expressed in terms of the number of page or overlay coordinate system X-axis measurement units. If the referenced object specifies an Object Environment Group (OEG), this parameter overrides the corresponding parameter in the Object Area Position (OBP) structured field of the OEG. If the object is a page segment, this parameter overrides the corresponding OBP parameters in the environment groups of all objects that comprise the page segment and specifies the object area offsets from the page or overlay origin for all data objects in

Include Object (IOB)

the page segment. A value of X'FFFFFF' indicates that the X-axis offset specified in the object's OEG is to be used; therefore the offset value (−1) is excluded from the allowed range. If the object does not specify the X-axis offset in an OEG, the architected default is X'000000'.

If RefCSys = X'00', this parameter specifies the offset along the I axis to the I-position of the current LND, RCD or XMD. The value for the parameter in this case is expressed in terms of the number of I-axis measurement units.

YoaOset If RefCSys = X'01', this parameter specifies the offset along the Y axis, Y_{pg} or Y_{ol}, of the including page or overlay coordinate system to the origin of the Y axis, Y_{oa}, of the object area coordinate system. The value for this parameter is expressed in terms of the number of page or overlay coordinate system Y-axis measurement units. If the referenced object specifies an Object Environment Group (OEG), this parameter overrides the corresponding parameter in the Object Area Position (OBP) structured field of the OEG. If the object is a page segment, this parameter overrides the corresponding OBP parameters in the environment groups of all objects that comprise the page segment and specifies the object area offsets from the page or overlay origin for all data objects in the page segment. A value of X'FFFFFF' indicates that the Y-axis offset specified in the object's OEG is to be used; therefore the offset value (−1) is excluded from the allowed range. If the object does not specify the Y-axis offset in an OEG, the architected default is X'000000'.

If RefCSys = X'00', specifies the offset along the B axis to the B-position of the current LND, RCD or XMD. The value for the parameter in this case is expressed in terms of the number of B-axis measurement units.

XoaOrent If RefCSys = X'01', this parameter specifies the amount of clockwise rotation of the object area's X axis, X_{oa}, about its defined origin relative to the X axis of the page or overlay coordinate system. If the referenced object specifies an Object Environment Group (OEG), this parameter overrides the corresponding parameter in the Object Area Position (OBP) structured field of the OEG. If the object is a page segment, this parameter overrides the corresponding OBP parameters in the environment groups of all objects that comprise the page segment. A value of X'FFFF' indicates that the X-axis rotation specified in the object's OEG is to be used. If the object does not specify the X-axis rotation in an OEG, the architected default is X'0000' (0 degrees).

If RefCSys = X'00', this parameter specifies the amount of clockwise rotation of the object area's X axis, X_{oa}, about its defined origin relative to the I axis of the current LND, RCD or XMD (I,B) coordinate system.

YoaOrent If RefCSys = X'01', this parameter specifies the amount of clockwise rotation of the object area's Y axis, Y_{oa}, about its defined origin relative to the X axis of the page or overlay coordinate system. The YoaOrent value must be 90 degrees greater than the XoaOrent value or a X'01' exception condition exists. If the referenced object specifies an Object Environment Group (OEG), this parameter overrides the corresponding parameter in the Object Area Position

(OBP) structured field of the OEG. If the object is a page segment, this parameter overrides the corresponding OBP parameters in the environment groups of all objects that comprise the page segment. A value of X'FFFF' indicates that the Y-axis rotation specified in the object's OEG is to be used. If the object does not specify the Y-axis rotation in an OEG, the architected default is X'2D00' (90 degrees).

If RefCSys = X'00', specifies the amount of clockwise rotation of the object area's Y axis, Y_{oa}, about its defined origin relative to the I axis of the current LND, RCD or XMD (I,B) coordinate system.

Note: The following combinations of values are the only ones valid for the XoaOrent and YoaOrent parameters:

XoaOrent	YoaOrent	Description
X'0000'	X'2D00'	0 and 90 degrees respectively
X'2D00'	X'5A00'	90 and 180 degrees respectively
X'5A00'	X'8700'	180 and 270 degrees respectively
X'8700'	X'0000'	270 and 0 degrees respectively

RefCSys   Determines how the object is positioned and rotated on the page:

Value   **Description**

X'00'   The object area offset in the IOB is measured with respect to the current LND, RCD or XMD position using the current text (I,B) coordinate system. The object area rotation in the IOB is measured with respect to the current text (I,B) coordinate system I-axis. In this case, the object area offset and rotation parameters must be specified explicitly, that is, a value of all X'F's, which indicates that the value in the object's OEG is to be used, is not supported for {XoaOset,YoaOset} and {XoaOrent,YoaOrent} when RefCSys = X'00'.

X'01'   The object area offset in the IOB is measured with respect to the page origin (X_p=0,Y_p=0) using the page (X_p,Y_p) coordinate system. The object area rotation in the IOB is measured with respect to page (X_p,Y_p) coordinate system X_p-axis.

When line data with IOBs is transformed into MO:DCA data, the IOBs are generated in the MO:DCA data as well. If an IOB specifies RefCSys=X'00', the position and orientation must be modified for the MO:DCA IOB to specify the equivalent position and orientation based on the page (X_p,Y_p) coordinate system.

IOB Triplets

Extended Resource Local Identifier (X'22') Triplet

The Extended Resource Local Identifier (X'22') triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

This triplet is mandatory when the IOB structured field is specified in a PageDef, in which case it must occur once. If this triplet is specified more than once, only the first is used. It specifies a local identifier for the IOB which is used to reference

Include Object (IOB)

the IOB from one or more LND, RCD or XMD structured fields. The ID specified for each IOB must be unique within the PageDef.

Triplet X'22' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M
1	CODE	Tid	X'22'	Identifies the Extended Resource Local Identifier triplet	M
2	CODE	ResType	X'30'	Specifies the resource type: **X'30'** IOB Reference	M
3–6	CODE	ResLID	X'00000000'–X'FFFFFFFF'	Specifies the extended resource local ID	M

Triplet X'22' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Extended Resource Local Identifier triplet.

ResType   Specifies the resource type associated with the extended local ID.

   Value   Description

   X'30'   IOB Reference. The local identifier (LID) specified by this triplet is assigned to an IOB structured field and is used by an LND, RCD or XMD to reference the IOB that specifies this ID. This value is only used when the triplet occurs in a Page Definition in AFP line-data environments.

   All others
       Reserved

ResLID   Specifies a unique resource object Local ID. It may be in the range of X'00000000' to X'FFFFFFFF'.

Include Page Overlay (IPO)

> **Note:** The IPO is a MO:DCA structured field. The following description documents its use in line-mode and mixed mode applications and introduces parameter values that are only valid in these environments. For the formal definition of the IPO structured field, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

The Include Page Overlay structured field references an overlay resource object that is to be positioned on the page. The overlay contains its own Active Environment Group. For line-mode and mixed-mode applications only, a value of X'FFFFFF' may be used for either the X axis offset (bytes 8–10), the Y axis offset (bytes 11–13), or both.

IPO (X'D3AFD8') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AFD8'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	OvlyName		Name of the overlay resource	M
8–10	SBIN	XolOset	−32768 − +32767	X-axis origin for the page overlay	M
			X'FFFFFF'	Use X-axis position specified by current LND or RCD	
11–13	SBIN	YolOset	−32768 − +32767	Y-axis origin for the page overlay	M
			X'FFFFFF'	Use Y-axis position specified by current LND or RCD	
14–15	CODE	OvlyOrent	X'0000', X'2D00', X'5A00', or X'8700'	The overlay's X-axis rotation from the X_p axis of the page: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	O
16–*n*		Triplets		See the *Mixed Object Document Content Architecture Reference*, SC31-6802, for triplet applicability.	O

IPO Semantics

OvlyName   The token name of the overlay being referenced. If the first two characters of the overlay name are 01, then bytes 0 and 1 must contain the characters O and 1, respectively.

XolOset   The offset along the X_p axis from the page origin where the origin of the overlay is placed. The value for this offset is expressed in terms of the measurement units currently in effect for the active Data Map. A value of X'FFFFFF' indicates that the overlay is to be placed at the X_p axis point specified by the current LND or RCD; therefore the offset value (−1) is excluded from the allowed range.

Include Page Overlay (IPO)

YolOset The offset along the Y_p axis from the page origin where the origin of the overlay is placed. The value for this offset is expressed in terms of the measurement units currently in effect for the active Data Map. A value of X'FFFFFF' indicates that the overlay is to be placed at the Y_p axis point specified by the current LND or RCD; therefore the offset value (−1) is excluded from the allowed range.

OvlyOrent Overlay orientation. Specifies the amount of rotation of the page overlay's X axis about the page overlay origin relative to the X_p axis of the page. The page overlay X axis rotation is limited to 0, 90, 180, and 270 degrees. The page overlay Y axis rotation is always 90 degrees greater than the page overlay X axis rotation.

If no value is specified for this parameter, the architected default is 0 degrees. Note that the 90°, 180°, 270° rotations of a page overlay are not supported in all AFP environments. Consult the product documentation to see which rotations are supported. Also note that the MO:DCA IS/1 and IS/2 interchange sets only support 0° rotation of a page overlay.

Include Page Segment (IPS)

Note: The IPS is a MO:DCA structured field. The following description documents its use in line-mode and mixed mode applications and introduces parameter values that are only valid in these environments. For the formal definition of the IPS structured field, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

The Include Page Segment structured field references a page segment resource object that is to be positioned on the page or overlay. For line-mode or mixed-mode applications only, a value of X'FFFFFF' may be used for either the I-axis offset (bytes 8–10), the B-axis offset (bytes 11–13), or both.

IPS (X'D3AF5F') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AF5F'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–7	CHAR	PsegName		Name of the page segment resource	M
8–10	SBIN	IpsOset	−32768 − +32767	I-axis origin for the page segment	M
			X'FFFFFF'	Use I-axis position specified by current LND or RCD	
11–13	SBIN	BpsOset	−32768 − +32767	B-axis origin for the page segment	M
			X'FFFFFF'	Use B-axis position specified by current LND or RCD	
14–*n*		Triplets		See the *Mixed Object Document Content Architecture Reference*, SC31-6802, for triplet applicability.	O

IPS Semantics

PsegName The token name of the page segment being referenced. All eight bytes of the name must be specified.

IpsOset The offset along the I axis from the current text coordinate system origin (I=0,B=0) to the origin of the page segment. The value for this offset is expressed in terms of the measurement units currently in effect for the active Data Map and is measured using the current text (I,B) coordinate system. A value of X'FFFFFF' indicates that the page segment origin is to be placed at the I-axis point specified by the current LND or RCD; therefore the offset value (−1) is excluded from the allowed range.

BpsOset The offset along the B axis from the current text

coordinate system origin (I=0,B=0) to the origin of the page segment. The value for this offset is expressed in terms of the measurement units currently in effect for the active Data Map and is measured using the current text (I,B) coordinate system. A value of X'FFFFFF' indicates that the page segment origin is to be placed at the B-axis point specified by the current LND or RCD; therefore the offset value (−1) is excluded from the allowed range.

Note: The MO:DCA Page Segment Positioning Migration (X'27') triplet may be specified on the IPS in MO:DCA documents to capture the text orientation that was specified when the page segment referenced by the IPS was included in line data. The information in this triplet allows the page segment and its objects to be positioned and oriented correctly on the MO:DCA page.

Line Descriptor Count (LNC)

The Line Descriptor Count structured field specifies the number of Line Descriptor (LND), Record Descriptor (RCD) or XML Descriptor (XMD) structured fields in the Data Map Transmission Subcase.

LNC (X'D3AAE7') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AAE7'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–1	UBIN	NumDSC	1–65535	Number of LND, RCD or XMD structured fields in the Data Map Transmission Subcase	M

LNC Semantics

NumDSC   The number of LND, RCD or XMD structured fields in the Data Map Transmission Subcase.

Note: The LND, RCD or XMD in a Data Map are numbered sequentially, starting with 1. Values from 1 through the number of LND, RCD or XMD are allowed.

Line Descriptor (LND)

The Line Descriptor structured field contains information, such as line position, text orientation, font selection, field selection, and conditional processing identification, used to format line data.

Note: The LNDs in a Data Map are numbered sequentially, starting with 1. Values from 1 through the number of LNDs are allowed.

LND (X'D3A6E7') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A6E7'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–1	BITS	LNDFlgs			M
Bit 0			B'0'–B'1'	End Page if Skipping	
Bit 1			B'0'–B'1'	End Page if Spacing	
Bit 2			B'0'–B'1'	Generate Inline Position	
Bit 3			B'0'–B'1'	Generate Baseline Position	
Bit 4			B'0'–B'1'	Generate Font Change	
Bit 5			B'0'–B'1'	Generate Suppression	
Bit 6			B'0'–B'1'	Reuse Record	
Bit 7			B'0'–B'1'	Use Fixed Data	
Bit 8			B'0'	B'0'	
Bit 9			B'0'–B'1'	Use Compatibility TRC	
Bit 10			B'0'–B'1'	Set Text Color	
Bit 11			B'0'–B'1'	Conditional Processing	
Bit 12			B'0'–B'1'	Resource Object Include	
Bit 13			B'0'–B'1'	Relative Baseline Position	
Bit 14–15			B'00'	Reserved; must be zero	
2–3	UBIN	IPos	0 to page extent minus 1	Inline Position	M
4–5	UBIN	BPos	0 to page extent minus 1	Absolute baseline position	M
	SBIN		X'8000'–X'7FFF'	Relative baseline position	
6–9	CODE	TxtOrent		Text (I,B) Orientation	M
			X'0000 2D00',	0,90 degrees	
			X'2D00 5A00',	90,180 degrees	
			X'5A00 8700', or	180,270 degrees	
			X'8700 0000'	270,360 degrees	
10	CODE	FntLID	X'01'–X'7F'	Primary font local ID	M
			X'FF'	Presentation system default font	
11	CODE	ChnlCde	X'00'–X'0C'	Channel code	M
12–13	UBIN	NLNDskp	X'0001'–X'FFFF'	Next LND if skipping	M
14–15	UBIN	NLNDsp	X'0001'–X'FFFF'	Next LND if spacing	M
16–17	UBIN	NLNDreu	X'0001'–X'FFFF'	Next LND if reusing data	M
18–25	CHAR	SupName		Suppression token name. A value of all X'F's (null value) is not valid.	M

Offset	Type	Name	Range	Meaning	M/O
26	CODE	SOLid	X'01'–X'7F'	Shift-out font local ID	M
			X'00'	Not specified	
27–30	UBIN	DataStrt	X'00000000'–X'00007FFF'	Data start position	M
31–32	UBIN	DataLgth	X'0000'–X'FFFE'	Data length	M
			X'FFFF'	Place remaining bytes	
33–34	CODE	TxtColor	See "LND Semantics."	Text color value	M
35–36	UBIN	NLNDccp	X'0000'–X'FFFF'	Next LND if conditional processing	M
37	CODE	SubpgID	X'00'–X'FF'	Subpage ID	M
38–39	CODE	CCPID	X'0000'–X'FFFF'	CCP Identifier	M
40–*n*		Triplets		See "LND Semantics" for triplet applicability.	O

Architecture Note: Prior to the addition of color and conditional processing to the page definition, bytes 33 through 39 were not defined as part of the LND structured field. Older page definition generators continued to generate the shorter version of the LND for some time after conditional processing and color were added to the architecture. Some of these page definitions still exist and should be handled by page definition processors. The valid length of the LND without the color and conditional processing fields was 33 bytes. Processors of page definitions may safely assume the values for each missing field is zero. LND triplets are not allowed on the shorter version of the LND.

LND Semantics

LNDFlgs   LND flags

Bit 0   End Page if Skipping. Shows whether the page ends if the control character is set for skipping. This flag is processed if the channel code of the LND does not match the skip-to channel code.

Value	Description
B'0'	The current page does not end.
B'1'	The next data should be on a new page at the print position specified by the first LND with the indicated channel code. If the channel code is not found but the End Page if Skipping is set, the search for the channel code continues with the first LND in the Data Map.

Bit 1   End Page if Spacing. Shows whether the page ends if the control character is set for spacing.

Value	Description
B'0'	The current page does not end.
B'1'	The next data should be on a new page at the print position indicated by the first LND in the Data Map.

Line Descriptor (LND)

Bit 2 Generate Inline Position. Shows whether the data processed using this LND is placed on the page at the inline position specified in bytes 2–3. This position becomes the new print position.

Value **Description**

B'0' Use the current inline position.

B'1' Use bytes 2–3 for the new inline position.

Bit 3 Generate Baseline Position. Shows whether the data processed using this LND is placed on the page at the baseline position specified in bytes 4–5. The baseline position may be an absolute position or a relative position, as specified by bit 13. This position becomes the new print position.

Value **Description**

B'0' Use the current baseline position.

B'1' Use bytes 4–5 and bit 13 to generate either an absolute baseline position with respect to the current text (I,B) coordinate system origin or a relative baseline position with respect to a previously defined baseline position.

Bit 4 Generate Font Change. Shows whether the data processed using this LND is printed using the Font Local Identifier specified in byte 10.

Value **Description**

B'0' The following rules apply:
- If the record to be processed contains a TRC, use the font corresponding to the TRC.
- If the current Data Map maps fonts with MCF or MDR structured fields, use the first font in the font list.
- If the current Data Map does not map fonts, use the hardware default font.

B'1' Use the font specified in byte 10.

Bit 5 Generate Suppression. Shows whether the data processed using this LND is suppressible text.

Value **Description**

B'0' The data is not suppressible text.

B'1' The data is suppressible text.

If this LND is used to present the selected data as a bar code, this flag is ignored. Suppression is only supported for text.

Note: Refer to the *Presentation Text Object Content Architecture Reference*, SC31-6803, for more information on text suppression.

Bit 6 Reuse Record. Shows whether the line data processed by

this LND should be reused and processed by the LND specified in bytes 16–17 (NLNDreu).

Value   Description

B'0'   Do not reuse the record.

B'1'   Reuse the record.

Bit 7   Use Fixed Data. Shows whether to present text from the Fixed Data Text (FDX) structured fields.

Value   Description

B'0'   Do not present fixed data.

B'1'   Present fixed data.

Bytes 27–30 (DataStrt) specify the start of the text to be added, and bytes 31–32 (DataLgth) specify the number of data bytes.

Note: No data from the current record is placed within the page under the control of this LND.

Bit 9   Use Compatibility TRC. This bit indicates whether the compatibility TRC should be used.

Note: The compatibility TRC uses only the last 4 bits, but the noncompatibility TRC uses all 8 bits.

Value   Description

B'0'   Do not use the compatibility TRC.

Valid TRC values are from 0 through 126.

If a TRC is specified that is beyond the number of fonts mapped, use the first font in the MCF.

B'1'   Use the compatibility TRC.

Only the first four fonts specified are used when this bit is set. If the TRC value is greater than the number of fonts specified, use the first font.

Bit 10   Set Text Color. If this bit is B'1', it specifies that the line data processed by this LND is to be presented in the color specified by the Color Specification (X'4E') triplet, or by the value specified in bytes 33–34. The X'4E' triplet, if specified, takes precedence over the value in bytes 33–34 however the value in bytes 33–34 may be used instead of the X'4E' color if the presentation device does not support the PTOCA PT3 subset. If this bit is B'0', the line data processed by this LND is presented in the presentation process default color.

Bit 11   Conditional Processing. Indicates whether conditional processing should be performed on the current line data record.

Value   Description

Line Descriptor (LND)

B'0'	Do not perform conditional processing.
B'1'	Perform conditional processing.

If this bit is B'1', the LND is referred to as a *conditional processing LND* and is only used to specify data to be compared, not to place data on the page.

Bit 12 Resource Object Include. Indicates whether this LND specifies by name resource objects to be included on the page at a specified position.

Value	Description
B'0'	Do not include named resource objects.
B'1'	Include named resource objects identified in the Resource Object Include triplet on this LND.

Bit 13 Relative Baseline Position. Indicates whether the baseline position specified in bytes 4–5 of this LND is an absolute position or a relative position. If an absolute baseline position is specified, it is measured as a positive offset in the baseline direction from the current text (I,B) coordinate system origin. If a relative position is specified, it is measured as a positive or negative offset from a previous baseline position using the current text (I,B) coordinate system, which is defined by the text orientation specified in bytes 6–9.

Value	Description
B'0'	The baseline position specified in bytes 4–5 is an absolute position.
B'1'	The baseline position specified in bytes 4–5 is a relative position.

The following restriction applies to relative baseline positioning:

- The text orientation of an LND that specifies relative baseline positioning must be the same as the text orientation of the LND that defines the baseline position from which the relative offset is measured.

IPos Inline Position. The inline position of data, specified as an offset from the current text (I,B) coordinate system origin, which is defined by the text orientation specified in bytes 6–9 (TxtOrent). The offset is measured using the measurement units specified for the page in the PGD. If bit 2 of byte 0 is B'1', it is used. This position becomes the new print position.

Note: Data must not exceed the boundaries of the page, which are defined in the Page Descriptor (PGD) structured field. If the new print position is outside these boundaries, printing of the page stops.

BPos Baseline Position. The baseline position of data. If bit 13 specifies absolute baseline position, these bytes specify a positive offset in the baseline direction from the current text (I,B) coordinate system origin. If bit 13 specifies a relative baseline position, these bytes

specify a positive or negative offset from a previous baseline position using the current text (I,B) coordinate system, which is defined by the text orientation in bytes 6–9. The baseline position used as a reference for the relative offset depends on whether the LND that specifies relative positioning is a base LND, and on whether a page or subpage boundary was crossed since the last LND was used to print. For a definition of base LNDs see "Field Formatting—LND Processing" on page 30. The baseline position used as a reference for the relative offset is determined as follows:

- For base LNDs, offsets are defined relative to the last base LND processed, that is, the last LND used to print or the last LND processed for spacing. However, if a page or subpage boundary was crossed after the last base LND was processed, offsets are defined relative to the first LND for the page or subpage.
- For reuse LNDs other than base LNDs, the offset is defined relative to the last LND that was processed for print (whether or not data prints).
- If the first LND of a Data Map specifies relative positioning, its offset is defined relative to the current text coordinate system origin (I=0,B=0), using the current text (I,B) coordinate system.
- If the first LND of a subpage specifies relative positioning, its offset is defined relative to the last base LND used to print, using the current text (I,B) coordinate system. Note that when skipping into a subpage, if the skipped-to LND specifies relative positioning, the relative offset is measured with respect to the first LND of the subpage, which may specify a relative position as well. This function allows a subpage to "float" relative to the last print position.

The offset is measured using the measurement units specified for the page in the PGD. If bit 3 of byte 0 is B'1', this position is used and becomes the new print position.

Application Note: When relative baseline positioning is used, the PageDef generator cannot check for off-page errors, since the data normally determines, with skip-to-channel carriage controls, when the relative baseline LNDs are invoked. AFP print servers generate a page break if the active Data Map is about to position data past the page's y-extent. This does not cause the generation of an error message. Note that the page's y-extent is specified in the PGD of the Data Map.

TxtOrent Text (I,B) Orientation. The four valid text orientations are:

Orientation	Value
0°,90°	X'0000 2D00'
90°,180°	X'2D00 5A00'
180°,270°	X'5A00 8700'
270°,0°	X'8700 0000'

Note that a change in text orientation means a change in the origin of the text (I,B) coordinate system:

Orientation	Value
0°,90°	Top-left corner of page

Line Descriptor (LND)

90°,180°	Top-right corner of page
180°,270°	Bottom-right corner of page
270°,0°	Bottom-left corner of page

If relative Baseline positioning is specified in bit 13, this text orientation must be the same as the text orientation of the LND that defines the baseline position from which the relative offset is measured.

FntLID Primary Font Local Identifier. If bit 4 of byte 0 is B'1', this is the local identifier of the font for text processed by this LND. When in shift-out/shift-in (SOSI) processing mode, this is also the local identifier of the font to be used following the implicit shift-in at the start of a record, and, when using record-based SOSI font selection, it is also the font Local identifier to be used following an explicit shift-in in a record. A null value (X'FF') indicates that a presentation system default font is to be used.

> **Note:** The Map Coded Font or Map Data Resource structured field in the AEG for the Data Map must have mapped the local identifier to the name of a font.

ChnlCde Channel Code. The channel code of this LND. The value :hex.00:ehex. indicates that this LND has no channel code.

NLNDskp Next Line Descriptor if Skipping. If the record indicates skipping, this parameter points to the LND to use in scanning for the channel code indicated by the control character of the record. If in following the Next Line Descriptor if Skipping chain, the channel code is not found but the End Page if Skipping bit is set, the search for the channel code continues with the first LND in the Data Map.

The LNDs in a Data Map are numbered sequentially, starting with 1. Values from 1 through the number of LNDs are allowed.

NLNDsp Next Line Descriptor if Spacing. If the record indicates spacing, this parameter points to the LND to use next. A chain of Next Line Descriptor if Spacing values is followed until the number of entries followed equals the number of spaces desired.

The LNDs in a Data Map are numbered sequentially, starting with 1. Values from 1 through the number of LNDs are allowed.

NLNDreu Next Line Descriptor if Reusing Data. This parameter points to the LND used to continue processing the record when reusing data.

If bit 6 of byte 0 = B'1', this parameter points to the next LND in a chain to process the same data record. The end of the chain is indicated by bit 6 of byte 0 = B'0' and a value of X'0000' in this parameter. At the end of the chain, control returns to the base LND. For example, if single spacing is specified and the data does not contain a skip-to-channel carriage control, processing resumes with the LND following the base LND.

When machine carriage controls or no carriage controls are specified, the LND that started the reuse chain is used with the current data record to determine the next LND to use. When ANSI carriage controls are specified, the LND that started the reuse chain is used with the next data record to determine the next LND to use.

The LNDs in a Data Map are numbered sequentially, starting with 1. Values from 1 through the number of LNDs are allowed. The chain of LNDs that is traversed when re-using a record does not have to follow any numerical order.

SupName Suppression token name. The suppression to be used with this LND. This value must match bytes 0–7 in one of the repeating groups in the Map Suppression (MSU) structured field of the Form Definition. The value can be any 8-byte value except null.

This parameter is ignored if bit 5, byte 0 is B'0'.

Note that suppression is only supported for text. If the data selected by this LND is presented as a bar code, this parameter is ignored. Note also that if text suppression is activated, only the field or record processed by this LND is suppressed, not text data that may be included using a Resource Object Include (X'6C') triplet.

SOLid Shift-out Font Local Identifier. If this byte is non-zero, it specifies the local identifier of the font to be used when a shift-out control is encountered in the record processed by this LND. Use of this parameter signals record-based font selection for SOSI processing. If this byte is zero, the parameter is not specified.

Note: When processing line data in shift-out/shift-in (SOSI) mode, an implicit shift-in is assumed at the start of every record.

Datastrt Data Start Position. The offset of the first data byte to be processed by this LND. If bit 7 of byte 0 is B'1', the data from the Fixed Data Text (FDX) structured field is used. Otherwise, the data from the current record is used.

A value of 0 indicates that the first byte is to be used. No data is placed if this value is greater than the length of the data source.

If conditional processing is to be performed (bit 11 of byte 1 is B'1'), this parameter defines the start of the data to be compared, known as the Comparison field. The comparison is determined by the Conditional Processing Control (CCP) structured field identified in bytes 38–39 (CCPID).

Datalgth Data Length. The number of bytes of data to be processed by this LND. If bit 7 of byte 0 is B'1', the data from the Fixed Data Text (FDX) structured field is used. Otherwise, the data from the current record is used.

If this value is X'FFFF', all the remaining data bytes are processed.

If this parameter causes data to be positioned outside the boundaries of the page, which are defined in the Page Descriptor (PGD) structured field, the printing of the page stops.

If conditional processing is to be performed (bit 11 of byte 1 is B'1'), this parameter defines the length of the comparison field. The comparison is determined by the Conditional Processing Control (CCP) structured field identified in bytes 38–39 (CCPID).

TxtColor Text Color Value. The specified color is used to present text processed by this LND when LND byte 0, bit 10=B'1', and a Color Specification (X'4E') triplet is not specified for this LND. This color may also be used if the X'4E' triplet is specified but the PTOCA

Line Descriptor (LND)

PT3 subset is not supported by the presentation device. Color values are defined as shown in Table 10. They reflect the range of values defined in the Standard OCA Color Value Table. For a definition of the Standard OCA Color Value Table, see the *Mixed Object Document Content Architecture Reference*, SC31-6802. RGB values are also defined for each color, assuming that the intensity range for each component is 0–255.

Table 10. Color-Value Table

Value	Color	Red (R)	Green (G)	Blue (B)
X'0000' or X'FF00'	Device default			
X'0001' or X'FF01'	Blue	0	0	255
X'0002' or X'FF02'	Red	255	0	0
X'0003' or X'FF03'	Pink/magenta	255	0	255
X'0004' or X'FF04'	Green	0	255	0
X'0005' or X'FF05'	Turquoise/cyan	0	255	255
X'0006' or X'FF06'	Yellow	255	255	0
X'0007'	White; see note 1	255	255	255
X'0008'	Black	0	0	0
X'0009'	Dark blue	0	0	170
X'000A'	Orange	255	128	0
X'000B'	Purple	170	0	170
X'000C'	Dark green	0	146	0
X'000D'	Dark turquoise	0	146	170
X'000E'	Mustard	196	160	32
X'000F'	Gray	131	131	131
X'0010'	Brown	144	48	0
X'FF07'	Device default	—	—	—
X'FF08'	Color of medium	—	—	—
All others	Reserved	—	—	—

Notes:

1. The color rendered on presentation devices that do not support white is device-dependent. For example, some printers simulate white with the color of the medium, which results in white if a white medium is used.

2. The value X'FFFF' is supported for migration purposes only and specifies the presentation process default.

NLNDccp   Next Line Descriptor if Conditional Processing. A non-zero value in this parameter means that conditional processing is to be performed on the current line data record. This parameter points to the LND used to perform conditional processing on the current input line data record. The target LND, called a *conditional processing LND*, must specify flag bit 11=B'1' and must point to a CCP structured field that defines the conditional processing. The conditional processing LND defines the data to be compared and does not place data on the page. Note that a conditional processing LND may point to another conditional processing LND.

SubPgID   Subpage Identifier. For conditional processing, a page can be divided into subpages to specify boundaries on which conditional processing actions can be taken. All LND structured fields of one subpage should have the same value in their Subpage Identifier byte, differing from the value in the LND structured fields of neighboring subpages. See Chapter 3, "Using a Page Definition to Print Data," on page 15 for more information on conditional processing and subpages.

                     PSF detects the beginning and end of subpages by checking for a change in the value of byte 37 (SubPgID). If conditional processing is to be performed (bit 11 of byte 1 is B'1'), this parameter is used to identify subpages when the timing of the conditional action is relative to the subpage.

CCPID   CCP Identifier. If bit 11 of byte 1 is B'1' this parameter is used to locate the associated Conditional Processing Control (CCP) structured field that describes the conditional processing to be performed for this Page Definition.

Triplets   Optional Triplets. See "Extended Resource Local Identifier (X'22') Triplet" on page 114, "Color Specification (X'4E') Triplet" on page 115, "Bar Code Symbol Descriptor (X'69') Triplet" on page 116, "Additional Bar Code Parameters (X'7B') Triplet" on page 121, "Resource Object Include (X'6C') Triplet" on page 122, for detailed information about the triplets permitted on the LND structured field.

LND Triplets

Fully Qualified Name (X'02') Triplet

The Fully Qualified Name (X'02') triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

This triplet is optional and may occur one or more times when a Bar Code Symbol Descriptor (X'69') triplet is specified on the LND, RCD, or XMD. The Fully Qualified Name type that may appear is X'DE'- *Data Object External Resource Reference*. The FQN triplet specifies the external identifier of a Color Management Resource (CMR) object that is used for the Bar Code object being generated. The identifier is used by the presentation system to locate the resource object in the resource hierarchy. The identifier is a character-encoded name which must be specified using FQNFmt = X'00'. The encoding for the external identifier of the CMR must be UTF-16BE.

Triplet X'02' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	150	Length of the triplet, including Tlength	M
1	CODE	Tid	X'02'	Identifies the Fully Qualified Name triplet	M
2	CODE	FQNType	X'DE'	Data Object External Resource Reference	M
3	CODE	FQNFmt	X'00'	GID format is character string	M

Line Descriptor (LND)

Offset	Type	Name	Range	Meaning	M/O
4–149		FQName		GID of the CMR. Must be 146 bytes in length and encoded using UTF-16BE.	M

Triplet X'02' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Fully Qualified Name triplet.

FQNType   Specifies the how the fully qualified name is to be used.

Value	Description
X'DE'	This GID specifies the name of a Color Management Resource to use when generating the Bar Code object.
All others	Reserved

FQNFmt   Specifies the format of the Global Identifier.

Value	Description
X'00'	The GID is a character-encoded name which means the data type is CHAR.
All others	Reserved

FQName   Contains the Global Identifier (GID) to be used as the name of the CMR. The encoding for the external identifier of the CMR must be UTF-16BE.

Extended Resource Local Identifier (X'22') Triplet

The Extended Resource Local Identifier (X'22') triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

This triplet is optional and may occur one or more times to reference an IOB structured field in the PageDef. The reference consists of a local identifier which must match the local identifier on an IOB in the PageDef. When an IOB with matching ID is found, the IOB is processed to present the object on the page. If an IOB with matching ID is not found, an exception is generated.

Triplet X'22' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M
1	CODE	Tid	X'22'	Identifies the Extended Resource Local Identifier triplet	M
2	CODE	ResType	X'30'	Resource type: **X'30'** IOB Reference	M
3–6	CODE	ResLID	X'00000000'–X'FFFFFFFF'	Specifies the extended resource local ID	M

Triplet X'22' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Extended Resource Local Identifier triplet.

ResType   Specifies the resource type associated with the extended local ID.

        Value **Description**
        X'30'   IOB Reference. The local identifier (LID) specified by this triplet is assigned to an IOB structured field and is used by an LND, RCD or XMD to reference the IOB that specifies this ID. This value is only used when the triplet occurs in a Page Definition in AFP line-data environments.

        All others
            Reserved

ResLID   Specifies a unique resource object Local ID. It may be in the range of X'00000000' to X'FFFFFFFF'.

        Note: RefCSys parameter in IOB byte 26 determines how the object included with the IOB is positioned and rotated on the page:

            Value **Description**
            X'00'   The object area offset in the IOB is measured with respect to the current LND, RCD or XMD position using the current text (I,B) coordinate system. The object area rotation in the IOB is measured with respect to the current text (I,B) coordinate system I-axis.
            X'01'   The object area offset in the IOB is measured with respect to the page origin (X_p=0,Y_p=0) using the page (X_p,Y_p) coordinate system. The object area rotation in the IOB is measured with respect to page (X_p,Y_p) coordinate system X_p-axis.

Color Specification (X'4E') Triplet

The Color Specification triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802. Support for this triplet is tied to PTOCA PT3 support in the PSFs and in printers. See your PSF documentation and also *Advanced Function Presentation Printer Information*, G544-3290.

This is an optional triplet that specifies a color for text processed by this LND when LND byte 0 bit 10=B'1' or when the selected data is to be presented as a bar code symbol. One Color Specification triplet may be specified on an LND. If this triplet is specified more than once, only the first is used. If this triplet is specified for text when LND byte 0 bit 10=B'0', it is ignored.

With this triplet a color is specified by selecting a color space, an encoding for the components of the color value in that space, and a color value. The color spaces supported are:
- RGB
- CMYK
- Highlight
- CIELAB
- Standard OCA color space

The selected encoding defines the number of bits used to specify each component. For example, with the RGB color space, one supported encoding is 8 bits per component, which maps the component intensity range 0 to 1 to the binary values 0 to 255. The color value specifies the color. With the RGB color space and an 8 bit

per component encoding, the color value (255,255,255) specifies full intensity for each component, which defines the color white.

Bar Code Symbol Descriptor (X'69') Triplet

Architecture Note: The Bar Code Symbol Descriptor triplet is registered in the MO:DCA architecture as a private-use triplet since it is used only in the PageDef object, which is not a MO:DCA object.

This is an optional triplet and may occur once. If this triplet is specified more than once, only the first is used. When present, it specifies that the data selected by LND, RCD or XMD parameters *DataStrt* and *DataLgth* or the data selected by the RCD or XMD parameter *Fldno* is to be presented as a bar code symbol. The data can be from the current record or it can be fixed data. The origin of the bar code symbol or the character reference point (see byte 4, bit 5, for details) on the page presentation space is specified by the LND, RCD or XMD position parameters *IPos* and *BPos*, and the orientation of the bar code symbol with respect to the page presentation space X_p-axis is specified by the LND, RCD or XMD text orientation parameter *TxtOrent*. Note that the suppression function that may be specified on an LND, RCD or XMD is not supported when the record or field is presented as a bar code. Note also that when an LND that specifies bar code presentation is re-used because the carriage control in the line data record specifies suppress spacing, only the last bar code generated by this LND is presented. If this triplet is specified, LND, RCD or XMD flag bit 5—Generate Suppression, and the SupName parameter are ignored.

This triplet is not supported on conditional processing LNDs, RCDs or XMDs that is LNDs, RCDs or XMDs that specify flag bit 11=B'1'; if present, it is ignored.

This triplet carries parameters defined by the BCOCA architecture, for more information see the *Bar Code Object Content Architecture Reference*, S544-3766. Not all PSFs support this triplet. See Appendix D, "System Support Information," on page 183 to determine if this triplet is supported in your environment.

Triplet X'69' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	4 or 18	Length of the triplet, including Tlength	M
1	CODE	Tid	X'69'	Identifies the Bar Code Symbol Descriptor triplet	M
2–3	CODE	DescID	X'0001'–X'FFFE'	Identifies a bar code symbol descriptor	M

Offset	Type	Name	Range	Meaning		M/O
4	BITS	SymbFlgs				O
Bit 0		HRI	B'0'–B'1'	**B'0'**	HRI is presented	
				B'1'	HRI is not presented	
Bits 1–2		HRIPos	B'00', B'01', or B'10'	**B'00'**	Default	
				B'01'	HRI below	
				B'10'	HRI above	
Bit 3		Astrsk	B'0'–B'1'	**B'0'**	Asterisk not presented	
				B'1'	Asterisk presented	
Bit 4			B'0'	Reserved; must be zero		
Bit 5		SuppSym	B'0'–B'1'	**B'0'**	Present bar code symbol	
				B'1'	Suppress presentation of symbol	
Bit 6		TrlBlk	B'0'–B'1'	**B'0'**	Do not suppress trailing blanks in data	
				B'1'	Suppress trailing blanks in data and select type and modifier	
Bit 7			B'0'	Reserved; must be zero		
5–6				Reserved; must be zero		O
7	CODE	BCType	See BCOCA.	Bar code type		O
8	CODE	BCMod	See BCOCA.	Bar code modifier		O
9	CODE	FntLID	X'00'–X'FE'	Font local identifier		O
			X'FF'	BCOCA default font		
10–11	CODE	Color	See BCOCA.	Bar code color		O
12	UBIN	ModWdth	See BCOCA.	Module width in mils		O
13–14	UBIN	ElmtHt	See BCOCA.	Element Height in L-units		O
15	UBIN	Mult	See BCOCA.	Height multiplier		O
16–17	UBIN	WE:NE	See BCOCA.	Wide-to-narrow ratio		O

Triplet X'69' Semantics:

Tlength Contains the length of the triplet.

Tid Identifies the Bar Code Symbol Descriptor triplet.

DescID Specifies the ID of a bar code symbol descriptor. The descriptor is defined by bytes 4–17 of this triplet. If the ID matches the ID of a bar code symbol descriptor defined previously on an LND for this page, the previous descriptor is used regardless of whether bytes 4–17 are specified in the current descriptor. If the ID does not match a previously-defined ID, bytes 4–17 must be specified and define the bar code symbol descriptor that is to be identified with this ID and that is to be used to generate this symbol. The valid range for the ID is X'0001'-X'FFFE'.

For a given page, the presentation services program collects all bar code symbols that have the same bar code symbol descriptor ID and the same bar code symbol orientation and groups them into a single bar code object.

Line Descriptor (LND)

The origin for the bar code object presentation space that contains the symbols is one of the four corners of the page as determined by the text orientation specified in LND bytes 6–9 (TxtOrent). The bar code presentation space origin is therefore coincident with the current text coordinate system (I,B) origin.

For example, if the LND specifies a (90°,180°) text orientation, the origin of the bar code object presentation space is the top-right corner of the page, and bar code symbols in this object are rotated 90° with respect to the page X_p axis.

The extents of the object presentation space are determined by the extents of the page presentation space. For example, if the origin of the object presentation space is the top-right corner of the page, the X-extent of the object presentation space is the Y-extent of the page, and the Y-extent of the object presentation space is the X-extent of the page. The symbol origin offset from the object presentation space origin and from the current text (I,B) coordinate system origin is specified by LND bytes 2–5 (IPOS,BPOS).

The units of measure for the bar code object presentation space, used for determining symbol origin offsets, are the same as those defined on the page (X_p,Y_p) presentation space in the PGD structured field of the Active Environment Group (AEG) for the Data Map. The presentation services program also defines an object area presentation space for the object that is identical in size, position, and units of measure to the bar code presentation space, and that has the same rotation about the page X_p-axis as the bar code symbols in the object.

SymbFlgs   These flags specify additional controls.

   Bits 0–3   These bits have the same syntax and semantics as the corresponding bits in byte 0 of the Bar Code Symbol Data (BSA) structure defined by the BCOCA architecture.

   Bit 5   Bar code symbol suppression

   This flag specifies whether or not the bar code symbol is presented, as follows:

Value	Description
B'0'	Present the bar code symbol
B'1'	Suppress presentation of the bar code symbol. This can be used to print just the HRI. If both bit 0 and bit 5 are B'1' or the bar code does not support HRI, nothing is presented for this bar code object.

   When bit 5 = B'1', LND, RCD or XMD parameters *IPos* and *BPos* specify the character reference point for the first character of the HRI.

   Not all BCOCA receivers support suppression of the bar code symbol; receivers that do not support this optional function ignore bit 5.

   Bit 6   This flag is defined for AFP Line Data. It identifies

the desired method of handling trailing blanks in the bar code data; for some symbologies, the resulting data length is used to adjust the bar code type and modifier to match the resulting data length. The PageDef supports fixed-length fields for data that is to be bar encoded. Since some bar codes allow variable-length data, these fixed-length fields often are padded on the right with blanks; these blanks are often not intended to be included in the BCOCA object, particularly for a bar code type that does not allow blanks. This flag identifies how these trailing blanks should be handled when a BCOCA bar code object is built from the line data and PageDef information.

It is used as follows:

Value   Description

B'0'   Do not suppress trailing blanks in the bar code data.

B'1'   Suppress all trailing blanks in the bar code data and adjust the bar code type and modifier to match the resulting data length.

When the flag = B'1', the bar code data is first adjusted by suppressing trailing blanks and then the bar code type and modifier is adjusted based on the resulting length as follows:

- If the user specified an EAN bar code type (X'08', X'09', X'16', or X'17'), truncate the data and set the bar code type and modifier based on the resulting data length:

Resulting Data Length	Bar Code Type	Bar Code Modifier
2	X'16'—two-digit supplemental	X'00'
5	X'17'—five-digit supplemental	X'00'
7	X'08'—EAN-8	X'00'
12	X'09'—EAN-13	X'00'
14	X'16'—two-digit supplemental	X'01'
17	X'17'—five-digit supplemental	X'01'
Any other value	Error	

- If the user specified a UPC bar code type (X'03', X'05', X'06', or X'07'), truncate the data and set the bar code type and modifier based on the resulting data length:

Resulting Data Length	Bar Code Type	Bar Code Modifier
2	X'06'—two-digit supplemental	X'00'
5	X'07'—five-digit supplemental	X'00'

Line Descriptor (LND)

Resulting Data Length	Bar Code Type	Bar Code Modifier
10	X'05'—UPC version E	X'00'
11	X'03'—UPC version A	X'00'
12	X'06'—two-digit supplemental	X'02'
13	X'06'—two-digit supplemental	X'01'
15	X'07'—five-digit supplemental	X'02'
16	X'07'—five-digit supplemental	X'01'
Any other value	Error	

- If the user specified a POSTNET bar code type (X'18'), truncate the data and set the bar code type and modifier based on the resulting data length:

Resulting Data Length	Bar Code Type	Bar Code Modifier
5	X'18'—POSTNET	X'00'
9	X'18'—POSTNET	X'01'
11	X'18'—POSTNET	X'02' or X'04'
Any other value	X'18'—POSTNET	X'03'

Note: Since both the X'02' and X'04' modifiers have the same length, the processor of the PageDef cannot always determine which modifier is desired if the data is truncated to match. Therefore, the following rule is used to set the modifier:
- If the original modifier requested in the triplet is not X'04' and the resultant length of the field is 11 bytes, modifier X'02' is used as the modifier for the generated bar code.

- If the user specified a USPS Four-State bar code type (X'22'), truncate the data and set the bar code type and modifier based on the resulting data length:

Resulting Data Length	Bar Code Type	Bar Code Modifier
20	X'22'—USPS Four-State	X'00'
25	X'22'—USPS Four-State	X'01'
29	X'22'—USPS Four-State	X'02'
31	X'22'—USPS Four-State	X'03'
Any other value	error	

- If the user specified any other bar code type, use the user-specified bar code type and modifier.

BCtype   Indicates the type of bar code symbol to be generated. This

parameter has the same syntax and semantics as byte 12 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture.

BCMod
Gives additional processing information about the bar code symbol to be generated. This parameter has the same syntax and semantics as byte 13 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture.

FntLID
Specifies the local ID of a font used to render the HRI and to provide the code point to bar code character mappings. This parameter has the same syntax and semantics as byte 14 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture. The value X'FF' specifies the BCOCA default font Any other value needs to be mapped with a Map Coded Font (MCF) or Map Data Resource (MDR) structured field in the AEG of the Data Map.

Color
Specifies the color in which the bars of the bar code symbol and the HRI are to be presented when a Color Specification (X'4E') triplet is not specified. This color may also be used if the X'4E' triplet is specified but extended colors for BCOCA is not supported by the presentation device. This parameter has the same syntax and semantics as bytes 15–16 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture.

ModWdth
Specifies the width in mils (thousandths of an inch) of the smallest defined bar code element. This parameter has the same syntax and semantics as byte 17 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture.

ElmtHt
Specifies the height in L-units along the Y_{bc} axis of the bar code presentation space. The units of measure for the L-units are defined in the PGD structured field of the Active Environment Group of the Data Map. This parameter has the same syntax and semantics as bytes 18–19 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture.

Mult
Specifies a value that, when multiplied by the element height, yields the total bar and space height presented. This parameter has the same syntax and semantics as byte 20 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture.

WE:NE
Specifies the ratio of the wide-element dimension to the narrow-element dimension when only two different size elements exist, that is, for a two-level bar code. This parameter has the same syntax and semantics as bytes 21–22 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture.

Note: The last 14 bytes (bytes 4–17) in this triplet are optional as a group. That is, either they are all specified or none are specified. If the descriptor ID is intended to match a previously-defined descriptor ID, these bytes should not be specified. When present, byte 4 is identical in syntax and semantics to byte 0 of the Bar Code Symbol Data (BSA) structure defined by the BCOCA architecture. Bytes 5–17 are identical in syntax and semantics to bytes 10–22 of the Bar Code Symbol Descriptor (BSD) defined by the BCOCA architecture, except for the font local ID parameter, which must be set to X'FF' in the triplet to specify the device default font.

Additional Bar Code Parameters (X'7B') Triplet
This is an optional triplet that specifies additional parameters for non-linear bar code symbologies (for example, 2D bar codes). This triplet may occur one or more

times when a Bar Code Symbol Descriptor (X'69') triplet is specified. If this triplet is specified more than once, the data from each triplet is concatenated in the order it is received. If a X'69' triplet is not specified, the X'7B' triplet is ignored. If a X'7B' triplet is specified and the X'69' triplet selects a linear bar code symbol, the results are unpredictable.

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	4–254	Length of the triplet, including Tlength	M
1	CODE	Tid	X'7B'	Identifies the Additional Bar Code Parameters Triplet	M
2				Reserved; must be zero	M
3–*n*	CODE	AddParm		Additional parameters for non-linear bar code symbols.	M

Triplet X'7B' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Additional Bar Code Parameters triplet.

AddParm   Specifies additional parameters for non-linear bar code symbols. These parameters are specific to the particular symbology and may include parameters like symbol size (rows/columns) and processing mode.

Note: The data carried by a Bar Code Symbol Descriptor (X'69') triplet, with the exception of the SymbFlgs parameter, is used to build the Bar Code Data Descriptor (BDD) structured field for the resulting bar code object. The data carried by the Additional Bar Code Parameters triplet, along with the SymbFlgs parameter, the LND, RCD or XMD position, and the LND, RCD or XMD data, is used to build a Bar Code Data (BDA) structured field for the resulting bar code object. For a description of the contents of the Bar Code Data structured field, see the *Bar Code Object Content Architecture Reference*, S544-3766.

Resource Object Include (X'6C') Triplet

This is an optional triplet that identifies an overlay or page segment object to be presented on the page at a specified position. Multiple Resource Object Include triplets may be specified on the same LND.

If the triplet identifies an overlay, the overlay name must be mapped with an MPO structured field in the AEG of the Data Map. If the triplet identifies a page segment, the page segment may be mapped in the AEG with an MPS structured field. If mapped, the page segment is downloaded and may be used multiple times (this is called a *hard* page segment). If it is not mapped, the page segment data is downloaded as part of the page (this is called a *soft*) page segment.

This triplet is not supported on a conditional processing LND, that is, an LND that specifies flag bit 11=B'1'. If present, it is ignored.

Note: The Resource Object Include is a MO:DCA triplet. The following description documents its use in a Page Definition, and introduces parameter values that are only valid in this environment. For the formal definition of this triplet in the MO:DCA architecture, see the *Mixed Object Document Content Architecture Reference*, SC31-6802. Note that when used on an LND, this

triplet supports a negative range for the object origin offset. Only a positive range is supported when the triplet is used in MO:DCA data streams. Note also that when used on an LND, this triplet supports ObjType=X'5F' (page segment). This value is not supported in MO:DCA data streams. Not all PSFs support this triplet. See Appendix D, "System Support Information," on page 183 to determine if this triplet is supported in your environment.

Triplet X'6C' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	17 or 19	Length of the triplet, including Tlength	M
1	CODE	Tid	X'6C'	Identifies the Resource Object Include Triplet	M
2	CODE	ObjType	X'DF' or X'5F'	Object type: **X'DF'** Overlay object **X'5F'** Page Segment object	M
3–10	CHAR	ObjName		Name of the object	M
11–13	SBIN	IobjOset	−32768 − +32767	Relative I-axis offset	M
14–16	SBIN	BobjOset	−32768 − +32767	Relative B-axis offset	M
17–18	CODE	ObOrent	X'0000', X'2D00', X'5A00', or X'8700'	The overlay's X axis rotation relative to the X_p axis of the page: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	O

Triplet X'6C' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Resource Object Include triplet.

ObjType   Specifies the object type.

        Value   **Description**
        X'DF'   Overlay object
        X'5F'   Page segment object

ObjName   Specifies the object name.

IobjOset   Relative I-axis offset. Relative offset of the object origin along the I axis of the current text (I,B) coordinate system, measured from the current LND position using the page measurement units specified in the PGD.

BobjOset   Relative B-axis offset. Relative offset of the object origin along the B axis of the current text (I,B) coordinate system, measured from the current LND position using the page measurement units specified in the PGD.

ObOrent   Only supported for ObjType X'DF' = Overlay object. Specifies the amount of rotation, about the overlay origin, of the overlay's X_{ol} axis relative to the X_{pg} axis of the page. Valid values are the following:

        Value   **Description**

        X'0000'   0 degrees

X'2D00'	90 degrees
X'5A00'	180 degrees
X'8700'	270 degrees
All others	Reserved

The overlay Y axis rotation is always 90 degrees greater than the overlay X axis rotation.

Note: If this parameter is omitted, the architected default value for the overlay rotation is X'0000', zero degrees.

Note: When a page segment is included with this triplet, the ObOrent parameter is ignored, and the rotation of objects in the page segment is summarized in Table 6 on page 54.

Object Reference Qualifier (X'89') Triplet

The Object Reference Qualifier triplet is used to specify whether the name of an object is retrieved from the input data or retrieved using normal methods. If the name is to be retrieved from the input data, that name overrides any ObjName field and any Fully Qualified Name (type X'01') triplet that would normally be used to select an object. This triplet may occur once on an LND structured field.

When this triplet is specified, the following rules apply:
- If this triplet is specified more than once, only the first is used.
- This triplet applies to the first Resource Object Include (X'6C') triplet or Extended Resource Local Identifier (X'22') triplet that follows on the LND or RCD.
- If this triplet is not followed by either the Resource Object Include triplet or the Extended Resource Local Identifier triplet, then this triplet is ignored.
- The LND or RCD DataStrt/DataLgth fields or the RCD Fldno is used to select the name of the object. The object name retrieved from the input is not presented as data on the page.
  - If this triplet is followed by the Resource Object Include triplet, the *ObjName* parameter of the Resource Object Include triplet is ignored.
  - If this triplet is followed by the Extended Resource Local Identifier triplet, the *ObjName* parameter and the Fully Qualified Name (FQN) triplet using *FQNType* X'01' of the Include Object (IOB) structured field pointed to by the Extended Resource Local Identifier triplet are ignored.

Note: If the field of data specified by the DataStrt/Datalgth or Fldno parameters does not exist in the input record, then this Object Reference triplet and the Resource Object Include triplet or the Extended Resource Local Identifier triplet that it applies to are ignored.

This triplet is not supported on conditional processing LNDs or RCDs, that is LNDs or RCDs that specify flag bit 11=B'1' if present, it is ignored.

Triplet X'89' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	4	Length of the triplet, including Tlength	M
1	CODE	Tid	X'89'	Identifies the Object Reference Qualifier Triplet	M

Offset	Type	Name	Range	Meaning	M/O
2				Reserved; must be zero	M
3	BITS	QualFlg	See "Triplet X'89' Semantics" for bit definitions.	Object reference qualifier flags	M

Triplet X'89' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Object Reference Qualifier triplet.

QualFlg   Specifies object reference qualifier flags as follows:

Bit   **Description**

0   Object reference qualifier flags

    B'0'   Do not use selected input data to override object name.

    B'1'   Use selected data to override object name.

1–7   Reserved

Notes:

1. If this triplet is omitted, the architected default for QualFlg bit 0 is B'0'.

2. When the QualFlag bit 0 is B'1', the encoding of the object name obtained from the input data is platform dependant. It is the responsibility of the user to ensure that the encoding of the input data is correct for the platform being used.

Implementation Note: For interchange in AFP environments, the name retrieved from the input data by the LND or RCD:

- Must be no more than 8 characters long; if longer, some AFP print servers use only the first 8 characters.
- Must follow the naming conventions used in AFP environments; see **External Resource Naming Conventions** in *Mixed Object Document Content Architecture Reference* for a description.
- Must not contain platform-dependent library names or path names.
- Must be encoded using EBCDIC if the resource is mapped in the active environment group.
- Must not use double-byte encoding.
- Must not contain any shift-in or shift-out characters.

Color Management Resource Descriptor (X'91') Triplet

The Color Management Resource Descriptor (X'91') triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Line Descriptor (LND)

The Color Management Resource Descriptor triplet specifies the processing mode and scope for a Color Management Resource (CMR). This triplet is mandatory when the LND references a Color Management Resource (CMR) with the FQN type X'DE' triplet, in which case this triplet specifies the processing mode for the CMR, and must occur once for each FQN type X'DE' specified. It is ignored in all other cases. This triplet must immediately follow the FQN type X'DE' triplet that specifies the CMR name.

Triplet X'91' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	5	Length of the triplet, including Tlength	M
1	CODE	Tid	X'91'	Identifies the Color Management Descriptor triplet	M
2				Reserved; must be zero	M
3	CODE	ProcMode	X'01'-X'02'	Specifies the processing mode for the CMR: **X'01'** Process the CMR as an audit CMR **X'02'** Process the CMR as an instruction CMR	M
4	CODE	CMRScpe	X'01'	Scope of CMR is a data object in this page	M

Triplet X'91' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Color Management Descriptor triplet.

ProcMode   Specifies the processing mode for the CMR. Valid values are the following:

Value	Description
X'01'	This CMR describes processing that has been done to a document component; process the CMR as an audit CMR.
X'02'	This CMR describes processing that is to be done to a document component; process the CMR as an instruction CMR.
All others	Reserved

CMRScpe   Specifies the scope of the CMR when used inside a document. Valid values are the following:

Value	Description
X'01'	The scope of the CMR is a data object in this page.
All others	Reserved

Record Descriptor (RCD)

The Record Descriptor structured field contains information, such as record position, text orientation, font selection, field selection, and conditional processing identification, used to format line data that consists of records tagged with record identifiers.

Note: The RCDs in a Data Map are numbered sequentially, starting with 1.

RCD (X'D3A68D') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A68D'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0–9	CHAR	RecID		Record descriptor ID	M
10	CODE	RecType	X'00'–X'03'	Record Type	M
11–13	BITS	RCDFlgs			M
Bit 0			B'0'	Reserved; must be zero	
Bit 1			B'0'	Reserved; must be zero	
Bit 2			B'0'–B'1'	Generate Inline Position	
Bit 3			B'0'–B'1'	Generate Baseline Position	
Bit 4			B'0'–B'1'	Generate Font Change	
Bit 5			B'0'–B'1'	Generate Suppression	
Bit 6			B'0'–B'1'	Field RCD	
Bit 7			B'0'–B'1'	Use Fixed Data	
Bit 8			B'0'	Reserved; must be zero	
Bit 9			B'0'	Reserved; must be zero	
Bit 10			B'0'	Reserved; must be zero	
Bit 11			B'0'–B'1'	Conditional Processing RCD	
Bit 12			B'0'	Reserved; must be zero	
Bit 13			B'0'–B'1'	Relative Baseline Position	
Bit 14–15			B'00'	Reserved; must be zero	
Bit 16			B'0'–B'1'	New Page	
Bit 17			B'0'–B'1'	Print Page Number	
Bit 18			B'0'–B'1'	Reset Page Number	
Bit 19			B'0'–B'1'	Group Indicator	
Bit 20			B'0'–B'1'	Field Delimiter Size	
Bit 21			B'0'–B'1'	Use Record ID	
Bit 22–23			B'00'	Reserved; must be zero	
14				Reserved; must be zero	M
15–16	UBIN	IPos	0 to page extent minus 1	Inline Position	M
17–18	UBIN	BPos	0 to page extent minus 1	Absolute baseline position	M
	SBIN		X'8000'–X'7FFF'	Relative position	

Record Descriptor (RCD)

Offset	Type	Name	Range	Meaning	M/O
19–22	CODE	TxtOrent		Text (I,B) Orientation	M
			X'0000 2D00', X'2D00 5A00', X'5A00 8700', or X'8700 0000'	0,90 degrees 90,180 degrees 180,270 degrees 270,360 degrees	
23	CODE	FntLID	X'01'–X'FE'	Primary font local ID	M
			X'FF'	Presentation system default font	
24–25	UBIN	FLDrcd	X'0000'–X'FFFF'	Field RCD Pointer	M
26–33	CHAR	SupName		Suppression token name. A value of all X'F's (null value) is not valid.	M
34	CODE	SOLid	X'01'–X'FE'	Shift-out font local ID	M
			X'00'	Not specified	
35–38	UBIN	DataStrt	X'00000000'– X'00007FFFF'	Data start position	M
39–40	UBIN	DataLgth	X'0000'–X'FFFE'	Data length	M
			X'FFFF'	Place remaining bytes	
41–42	UBIN	CONDrcd	X'0000'–X'FFFF'	Conditional Processing RCD Pointer	M
43	CODE	SubpgID	X'00'	Subpage ID (Always X'00' for RCDs)	M
44–45	CODE	CCPID	X'0000'–X'FFFF'	CCP Identifier	M
46–47	UBIN	Pgno	X'0001'–X'FFFF'	Starting page number	M
			X'0000'	Not specified	
48–49	UBIN	ESpac	0 to page extent minus 1	End Space	M
50	CODE	Align	X'00'–X'01'	Field Alignment	M
51–52	CODE	FldDelim	X'0000'–X'FFFF'	Field Delimiter	M
53–54	UBIN	Fldno	X'0001'–X'FFFF'	Field Number	M
			X'0000'	Not specified	
55–56	UBIN	AdBIncr	X'0000'–X'FFFF'	Additional baseline increment	M
57–69				Reserved; must be zero	M
70–*n*		Triplets		See "RCD Semantics" for triplet applicability.	O

RCD Semantics

The RCD uses many parameters that are defined for the LND. The definition of such parameters is deferred to the LND. When such definitions are applied to the RCD, the term "LND" should be read as "RCD", and the byte offsets of the parameters should be adjusted to the RCD.

There are three types of RCDs:

- Record RCDs, which define the processing for an input record or define a default page header or trailer
- Field RCDs, which define the processing for an input field or specify constant text or graphics
- Conditional Processing RCDs, which specify the Conditional Processing associated with an input record

The Field RCDs and Conditional Processing RCDs associated with a Record RCD are chained to that RCD using RCD number pointers. An RCD is assumed to be a Record RCD if neither the Field RCD nor the Conditional Processing RCD bits are on in the RCDFlgs byte.

RecID Record Descriptor ID. The identifier to be used with this RCD. This field in the RCD is used only if this RCD does not contain a Fully Qualified Name (FQN) X'02' triplet type X'01'. The FQN type X'01' triplet is used to extend the identifier to a range of 1 to 250 bytes instead of 10 bytes. When an input record is processed with a Data Map containing RCDs, the first 1 to 250 data byte positions in the input record are assumed to contain a Record Descriptor ID and the Data Map's Record RCDs are searched to find a matching Record Descriptor ID. (A CC byte, if used with the input data stream, is not considered part of the input record.) When a match is found, that RCD is used to format the input record. If a matching RCD is not found, an error message is generated. Except for default Page Headers, default Page Trailers, Field RCDs and Conditional Processing RCDs, all of the RCDs in a Data Map have a unique Record Descriptor ID. The Record Descriptor ID is set to all zeros for default Page Headers (one per Data Map), default Page Trailers (one per Data Map), Field RCDs, and Conditional Processing RCDs. If this RCD contains a FQN type X'01' triplet, this 10 byte Record Descriptor ID field is set to all X'FF''s.

> **Notes:**
>
> 1. To be able to find a matching Record Descriptor ID, the encoding of the identifier specified in this parameter or in the FQN type X'01' triplet must match the encoding of the input data.
>
> 2. If the FQN type X'01' triplet is used, **all** record type RCD structured fields must use the FQN X'01' triplet.
>
> 3. If the FQN type X'01' triplet is used, the names specified for **all** the FQN triplets **must** be the same length. Blanks may be used to pad the name. The encoding used for the blanks must match the encoding of the data used for the name.

RecType Record Type. This parameter is ignored on RCDs other than Record RCDs.

> **Value** **Description**
>
> **X'00'** Body RCD. This RCD does not have any special header or trailer properties associated with it and is used to format any input records with a matching Record ID.
>
> **X'01'** Page Header RCD. This RCD is used to automatically print a header (such as the current customer name) at the beginning of each new page. The baseline position of this RCD can be anywhere on a logical page and can be specified as relative. If an input record is received with a matching Record Descriptor ID, that input record is not presented on receipt but is saved as the active page header record. If a default Record Descriptor ID is specified in a Page Header RCD, it is assumed to be a default Page Header RCD (only one default Page Header RCD can be specified in a Data Map and no input record data is

Record Descriptor (RCD)

processed with a default RCD). See the logical page eject processing section for page header and New Page processing details.

Note: Once a Page Header RCD is processed, the header record is saved for the duration of the job by the presentation services program. Whenever the same Data Map is re-invoked for that job, this saved header record is always presented with each page generated with the Data Map.

X'02' Page Trailer RCD. This RCD is used to automatically print a trailer (for example, a footnote and/or page number) at the end of each page. The baseline position of this RCD can be anywhere on a logical page and can be specified as relative. If an input record is received with a matching Record Descriptor ID, that input record is not presented on receipt but is saved as the active page trailer record. If a default Record Descriptor ID is specified in a Page Trailer RCD, it is assumed to be a default Page Trailer RCD (only one default Page Trailer RCD can be specified in a Data Map, and no input record data is processed with a default RCD). See the logical page eject processing section for page trailer and New Page processing details.

Note: Once a Page Trailer RCD is processed, the trailer record is saved for the duration of the job by the presentation services program. Whenever the same Data Map is re-invoked for that job, this saved trailer record is always presented with each page generated with the Data Map.

X'03' Group Header RCD. This RCD is used to automatically print a group header (for example, column headings for a group of data records) on a page. Note that the group header is not actually printed and causes no action until a Body RCD with Group Indicator (RCDFlgs bit 19) set to B'1' is processed for the page. The baseline position of this RCD can be specified as relative. If an input record is received with a matching Record Descriptor ID, that input record is saved as the active group header record and then presented. If that input record and/or RCD causes a page eject, that input record is used as the active group header record for the new page. See the logical page eject processing section for active group header and New Page processing details.

Note: Once a Group Header RCD is processed and is still active when leaving the Data Map, the group header record is saved by the presentation services program. Whenever the same Data Map is re-invoked, this saved group header record is presented again if the first body record after re-invoking the Data Map selects a Body RCD that has the Group Indicator on.

RCDFlgs Flag bits.

Bits 0–5   For a definition of these flag bits see "Line
                 Descriptor (LND)" on page 104. LND flag bit 0–1 is
                 reserved in the RCD.

Bit 6   Field RCD.

        Value   **Description**

        B'0'   If this bit and bit 11 are both B'0', this is a
                 Record RCD.

        B'1'   This is a Field RCD. If both this bit and bit
                 11 are on, it is an error.

Bits 7–15   For a definition of these flag bits see "Line
                 Descriptor (LND)" on page 104. LND flag bits 9–10
                 are reserved in the RCD. The color for the data
                 presented by this RCD is always the color specified
                 by the Color Specification (X'4E') triplet, if
                 specified. If this triplet is not specified, the data is
                 presented in the presentation process default color.

Bit 16   New Page.

        Value   **Description**

        B'0'   This bit value has no effect on this RCD.

        B'1'   A logical page eject should be executed
                 before presenting any data for this RCD. If
                 this is a header or trailer RCD, the print
                 position is moved to the start of a new
                 page before this header or trailer becomes
                 the active header or trailer. (See "Logical
                 Page Eject Processing" on page 135.) This
                 bit is ignored on RCDs other than Record
                 RCDs.

Bit 17   Print Page Number. Indicates whether a page
                 number should be generated on the current page
                 before the page eject is performed.

        Value   **Description**

        B'0'   This bit has no effect on this RCD.

        B'1'   Print page number. The (I,B) position of the
                 RCD indicates the position of the page
                 number, which is maintained by the
                 presentation process. The page number is
                 rendered with the font selected by the
                 FntLID parameter. This bit is ignored on
                 RCDs other than Field RCDs.

            Note: The FntLID parameter must be
                 mapped to a font global identifier
                 with an MCF structured field in the
                 AEG of the Data Map. For the page
                 formatter to generate the page
                 number code points without
                 accessing the font object, it needs to
                 know the encoding scheme

Record Descriptor (RCD)

(EBCDIC, ASCII, or Unicode) for the font. This information is specified on the MCF with an Encoding Scheme ID (X'50') triplet. Only single-byte EBCDIC, double-byte EBCDIC, single-byte ASCII, and Unicode (UTF-16) encodings are supported for printing page numbers. The code points used for the page numbers are X'F0'–X'F9', X'42F0'–X'42F9', X'30'–X'39', and X'0030'–X'0039', respectively. If the encoding scheme is not specified on the MCF, single-byte EBCDIC encoding is assumed.

Bit 18   Reset Page Number.

Value	Description
B'0'	This bit has no effect on this RCD.
B'1'	The current page number is reset to the value specified in the Pgno parameter. This bit is ignored on RCDs other than Field RCDs.

Bit 19   Group Indicator.

Value	Description
B'0'	The input data associated with this RCD is not part of a group. If a group header record is active, it is deactivated and any saved group header input record is discarded.
B'1'	Indicates that the existing group header should continue to be saved and used for subsequent pages. This bit is ignored on RCDs other than Record RCDs.

Bit 20   Field Delimiter Size.

Value	Description
B'0'	Delimiter is 1 byte and is specified in the second byte of the field.
B'1'	Delimiter is 2 bytes.

Bit 21   Use Record ID. Selects the Record ID as the data field to be used for presentation.

Value	Description
B'0'	Do not select the Record ID.
B'1'	Select the Record ID.

This function is restricted to Field RCDs; it is ignored on all other RCDs.

All others   Reserved; must be B'000'.

IPos   Inline Position See "Line Descriptor (LND)" on page 104.

BPos   Baseline Position See "Line Descriptor (LND)" on page 104.

- *Relative Baseline Position for Record and Field RCDs.* If the baseline position is relative, the offset is measured as follows:
  - For Page Header RCDs, the offset is relative to the top of the page.
  - For Group Header and Body RCDs, the offset is relative to the last Group Header or Body RCD processed; if there is none, it is relative to the top margin.
  - For Field RCDs, it is relative to the last Field or Body RCD that was processed for print (whether or not data is printed).
  - For Page Trailer RCDs, it is relative to the last Record RCD processed; if there is none, it is relative to the top margin.

  Note that the actual location of "top" and "top margin" on a page is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76.

- *Overflow Processing for a Record RCD.* If the specified baseline position is relative, this Record RCD and any Field RCDs that are part of this Record RCD are scanned to determine the resulting absolute baseline position at the end of processing this Record RCD. This computed baseline position is checked to see if it overflows into the bottom margin area. If not, this Record RCD is processed with the current input record. If it overflows into the bottom margin, a logical page eject is executed. See the logical page eject section for page eject processing details. If the data in a single Record RCD (with its chained Field RCDs) is too large to fit within the top and bottom margins on a page, an error is generated. Note that the actual location of "top margin" and "bottom margin" on a page is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76.

TxtOrent   Text (I,B) Orientation. See "Line Descriptor (LND)" on page 104.

FntLID   Primary Font Local Identifier. See "Line Descriptor (LND)" on page 104.

FLDrcd   The RCD number of a Field RCD. A non-zero value in this parameter on a Record RCD indicates that field processing is to be performed on the current input data record. This parameter specifies the RCD number of a Field RCD. Multiple Field RCDs can be chained to a Record RCD using this parameter. The last Field RCD in the chain has a value of X'0000' in this parameter.

SupName   Suppression token name. See "Line Descriptor (LND)" on page 104.

SOLid   Shift-out Font Local Identifier. See "Line Descriptor (LND)" on page 104.

Datastrt   Data Start Position. See "Line Descriptor (LND)" on page 104.

Datalgth   Data Length. See "Line Descriptor (LND)" on page 104.

CONDrcd   The RCD number of a Conditional Processing RCD. A non-zero value in this parameter on a Record RCD means that conditional processing is to be performed on the current input data record. This parameter specifies the relative RCD number of a Conditional

Record Descriptor (RCD)

	Processing RCD. Multiple Conditional Processing RCDs can be chained to a Record RCD using this parameter. The last Conditional Processing RCD in the chain has a value of X'0000' in this parameter. A Field RCD has a value of X'0000' in this parameter.

See "Line Descriptor (LND)" on page 104.

SubPgID	Subpage Identifier. See "Line Descriptor (LND)" on page 104.
CCPID	CCP Identifier. See "Line Descriptor (LND)" on page 104.
Pgno	Page Number. Specifies the starting page number that is used when RCDFlgs specifies reset page number. This parameter is ignored on RCDs other than Field RCDs.
ESpac	End Space.

Value	Description
X'0000'	No check is made for End Space.
All others	A check is initiated to verify that the remaining body space on the page (distance between the starting print position of this RCD and the bottom margin area) is greater than or equal to the baseline value specified in this parameter. If the remaining body space is less than the value specified, a logical page eject is executed. This can be used, for example, on a Group Header RCD (specifying space for the first data record of the group) to ensure that a group header does not print at the end of a page without the first data record of the group. This parameter is ignored on a Page Header or Page Trailer RCD and on RCDs other than Record RCDs. Note that the actual location of the bottom margin area is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76.

Align	Field Alignment.

Value	Description
X'00'	Field is left aligned to the position specified in IPos.
X'01'	Field is right aligned to the position specified in IPos. This parameter is ignored for Field RCDs that specify a Barcode or Graphics Descriptor Triplet and on RCDs other than Field RCDs.

FldDelim	Field Delimiter. This is a 1-byte or a 2-byte parameter, depending on the encoding used for the field. RCDFlg bit 20 indicates the size of this parameter. A 1-byte parameter is specified in the second byte of the field. A value of X'0000' indicates that this parameter is not specified. For Record RCDs, this parameter specifies a 1-byte or 2-byte code that delimits all of the input record fields (excluding the 10 byte Record Format ID) used with this RCD and any Field and Conditional Processing RCDs chained to this RCD. The delimiter is specified at the end of the field it delimits. A delimiter may be specified after the Record ID but is ignored. A Field Number parameter is used rather than Data Start and Data Length

parameters to specify the location of input fields on Conditional Processing and Field RCDs that are chained to this Record RCD. Data Start and Data Length parameters are still used to select bytes in the field or in Fixed Data Text. This parameter is ignored on all non-Record RCDs. For comparisons, any input record fields used with a CCP are assumed to be padded on the right out to the CCP Comparison String Length.

Fldno   Field Number. Specifies the number of the field to be processed. The first field (field number 1) in the record is followed by the first delimiter; the second field is followed by the second delimiter; and so on. Any delimiter specified after the Record ID is ignored. The DataStrt and DataLgth parameters are applied to the selected field to select specific bytes in the field for processing. Specifying DataStrt = X'00000000' and DataLgth = X'FFFF' selects the whole field for processing. This parameter is used only if the Record RCD that this Field or Conditional Processing RCD is chained to specifies a Field Delimiter (other than X'00').

AdBIncr   Additional baseline increment. Specifies the additional baseline increment that is to be added to the current baseline position after a group header or data record is presented. This parameter is only processed for Group Header RCDs, and for Body RCDs that are record RCDs. It is ignored on all other RCDs. Note that this increment is not used when positioning MO:DCA objects with respect to the current RCD in mixed mode.

Triplets See "Fully Qualified Name (X'02') Triplet" on page 113, "Extended Resource Local Identifier (X'22') Triplet" on page 137, "Color Specification (X'4E') Triplet" on page 138, "Bar Code Symbol Descriptor (X'69') Triplet" on page 138, "Additional Bar Code Parameters (X'7B') Triplet" on page 138, "Resource Object Include (X'6C') Triplet" on page 139, "Graphics Descriptor (X'7E') Triplet" on page 139 and "Object Reference Qualifier (X'89') Triplet" on page 146.

Logical Page Eject Processing

A logical page eject can be caused by the following:

- Any Record RCD with a specification of New Page
- A relative baseline overflow, that is, a Body or Group Header RCD with a relative baseline position value that when processed against the current input record causes an overflow of the current print position into the bottom margin (see Baseline Position). Note that the actual location of "bottom margin" is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76
- A Data Map change or Medium Map change, or, in mixed-mode, a Begin Document or Begin Page structured field

When a logical page eject occurs, the following actions are taken in the following order.

For the *current* page:

1. If this is the start of a line data document (no previous page ejects, group header records or body records have been processed with this PageDef), no header or trailer processing is performed.

2. If an active page header record was in effect prior to this RCD, that record is presented on the current page using the matching RCD. Otherwise, if the active Data Map contains a default Page Header RCD, that RCD is used to present a page header.

3. If an active page trailer record was in effect prior to this RCD, that record is presented on the current page using the matching RCD. Otherwise, if the active Data Map contains a default Page Trailer RCD, that RCD is used to present a page trailer.

For the *new* page:

1. The current print position is moved to the top of the new page. If the Data Map is changed, the new Data Map's Margin Definition and RCDs are used for subsequent processing.

2. The baseline position is offset from the top of the new page by the top margin.

3. If the RCD causing the page eject is a Page Header, Group Header, or Page Trailer RCD, the input record causing the page eject is saved as the active page header, group header or page trailer record.

4. If an active group header record exists for this Data Map, that record is presented on the new page using the matching Record RCD. Note that the group header is not actually printed and causes no action until a Body RCD with Group Indicator (RCDFlgs bit 19) set to B'1' is processed for the page. If the RCD specifies relative positioning, the baseline position of the RCD is offset from the top of the page by the top margin plus the RCD BPos value.

5. If the page eject was caused by a Body RCD, the input record causing the page eject is presented on the new page using the RCD referenced by the record. If the RCD specifies relative positioning and is preceded on the page by a group header, the baseline position is relative to the last printed line of the group header. If the RCD specifies relative positioning and is not preceded on the page by a group header, the baseline position of the RCD is offset from the top of the page (0 position on the B axis) by the top margin plus the RCD BPos value.

Note that the actual location of "top of page" and "top margin" is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76.

RCD Triplets

Fully Qualified Name (X'02') Triplet

The Fully Qualified Name (X'02') triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

This triplet is optional and may occur once. If this triplet is specified more than once, only the first is used. The Fully Qualified Name type that may appear is X'01'—*Replace First GID Name*. This GID overrides the Record Descriptor's *RecID* field and is used as the Record Descriptor ID.

Triplet X'02' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	5–254	Length of the triplet, including Tlength	M
1	CODE	Tid	X'02'	Identifies the Fully Qualified Name triplet	M

Offset	Type	Name	Range	Meaning	M/O
2	CODE	FQNType	X'01'	Replace First GID name	M
3	CODE	FQNFmt	X'00'	GID format is character string	M
4–n		FQName		GID of the Record Descriptor. Can be up to 250 bytes in length.	M

Triplet X'02' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Fully Qualified Name triplet.

FQNType   Specifies the how the fully qualified name is to be used.

Value	Description
X'01'	This GID replaces the *RecID* parameter in the structured field.

> **Note:** The GID (Global Identifier) that overrides the ten-byte *RecID* field has the same semantics as the ten-byte *RecID* parameter.

All others	Reserved

FQNFmt   Specifies the format of the Global Identifier.

Value	Description
X'00'	The GID is a character-encoded name which means the data type is CHAR.
All others	Reserved

FQName   Contains the Global Identifier (GID) to be used to override the *RecID* parameter.

> **Note:** To be able to find a matching Record Descriptor ID, the encoding of the identifier specified in this parameter must match the encoding of the input data.

Fully Qualified Name (X'02') Triplet

This triplet is optional and may occur one or more times when a Bar Code Symbol Descriptor (X'69') triplet or a Graphics Descriptor (X'7E') triplet is specified on the RCD. The Fully Qualified Name type that may appear is X'DE'—*Data Object External Resource Reference*. The FQN triplet specifies the external identifier of a Color Management Resource (CMR) object that is used for the Bar Code object or Graphics object being generated. The identifier is used by the presentation system to locate the resource object in the resource hierarchy. The identifier is a character-encoded name which must be specified using FQNFmt = X'00'. The encoding for the external identifier of the CMR must be UTF-16BE.

Extended Resource Local Identifier (X'22') Triplet

This triplet is optional. It may occur one or more times to reference an IOB structured field in the PageDef. This triplet is ignored if the Graphics Descriptor (X'7E') triplet is specified on the RCD.

Color Specification (X'4E') Triplet

This is an optional triplet that specifies the color for text processed by this RCD, bar code generated by this RCD, and for graphics generated by this RCD. If this triplet is not specified, the record is presented in the presentation process default color. On an RCD that processes text or generates bar code, this triplet may occur once. If this triplet is specified more than once, only the first is used. On a Field RCD with a Graphics Descriptor Triplet, this color triplet can be specified once or twice.

- If it is specified once, the color applies to all graphics constructs.
- If it is specified twice, then:
  - If the RCD generates a graphics area, such as a full arc or box, the first color applies to the fill pattern of the graphics area, and the second color applies to the boundary lines of the graphics area (if drawn).
  - If the RCD generates a graphics line, the second color overrides the first color and applies to the graphics line.
- If it is specified more than twice, only the first two are used.

If this RCD generates the start of a graphic primitive which is completed by a succeeding RCD, the colors specified on the "start" RCD determine the color for the complete primitive.

With this triplet a color is specified by selecting a color space, an encoding for the components of the color value in that space, and a color value. The color spaces supported are:
- RGB
- CMYK
- Highlight
- CIELAB
- Standard OCA color space

The selected encoding defines the number of bits used to specify each component. For example, with the RGB color space, one supported encoding is 8 bits per component, which maps the component intensity range 0 to 1 to the binary values 0 to 255. The color value specifies the color. With the RGB color space and an 8 bit per component encoding, the color value (255,255,255) specifies full intensity for each component, which defines the color white.

The Color Specification triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Bar Code Symbol Descriptor (X'69') Triplet

This is an optional triplet. It may occur once. If this triplet is specified more than once, only the first is used. This triplet is ignored if the Graphics Descriptor (X'7E') triplet is specified on the RCD.

See "Bar Code Symbol Descriptor (X'69') Triplet" on page 116. Note that the LND/RCD parameters used by this triplet may be at different offsets in the LND and RCD.

Additional Bar Code Parameters (X'7B') Triplet

This is an optional triplet that may occur one or more times when a Bar Code Symbol Descriptor (X'69') triplet is specified. If this triplet is specified more than once, the data from each triplet is concatenated in the order it is received.

See "Additional Bar Code Parameters (X'7B') Triplet" on page 121.

Resource Object Include (X'6C') Triplet

This is an optional triplet that identifies an overlay or page segment object to be presented on the page at a specified position. Multiple Resource Object Include triplets may be specified on the same RCD. This triplet is ignored if the Graphics Descriptor (X'7E') triplet is specified on the RCD.

See "Resource Object Include (X'6C') Triplet" on page 122.

Graphics Descriptor (X'7E') Triplet

Architecture Note: The Graphics Descriptor triplet is registered in the MO:DCA architecture as a private-use triplet because it is used only in the PageDef object, which is not a MO:DCA object.

This is an optional Field RCD triplet. It may occur once. If this triplet is specified more than once, only the first is used. Text input and fixed data text are ignored on a Field RCD that specifies a Graphics Descriptor Triplet. When present, the Graphics Descriptor Triplet specifies the generation of a graphics primitive. The triplet may specify the complete primitive, or the start of the primitive, or the end of the primitive.

This triplet is ignored on RCDs other than Field RCDs. This triplet specifies primitives and their parameters as defined by the AFP GOCA architecture. For more information, see the *IBM Graphics Object Content Architecture for Advanced Function Presentation*, S544-5498. Not all PSFs support this triplet. See Appendix D, "System Support Information," on page 183 to determine if this triplet is supported in your environment.

Triplet X'7E' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	20 or 35	Length of the triplet, including Tlength	M
1	CODE	Tid	X'7E'	Identifies the Graphics Descriptor triplet	M
2	CODE	ParmSpc	X'01'–X'03'	Parameter specification: **X'01'** Triplet specifies all parameters for graphics primitive **X'02'** Triplet specifies start parameters for graphics primitive **X'03'** Triplet specifies end parameters for graphics primitive	M
3–4	UBIN	Graphid	X'0000'–X'FFFE'	ID for matching Start/End graphic pairs	M

Record Descriptor (RCD)

Offset	Type	Name	Range	Meaning	M/O
5	CODE	GrPrim	X'01'–X'05'	Graphics primitive: **X'01'** Horizontal line at current position **X'02'** Vertical line at current position **X'03'** Diagonal line at current position **X'04'** Full Arc at current position **X'05'** Box at current position	M
6				Reserved; must be zero	M
7	BITS	GraFlgs			m
Bit 0			B'0'	Reserved; must be zero	
Bit 1		Fill	B'0'–B'1'	**B'0'** Interior of primitive is not filled **B'1'** Interior of primitive is filled	
Bit 2		Boundary	B'0'–B'1'	**B'0'** Boundary of primitive is not drawn **B'1'** Boundary of primitive is drawn	
Bit 3–7			B'00000'	Reserved; must be zero	
8–9	UBIN	Iend	0 to page extent minus 1	I-coordinate for primitive end point	M
10–11	UBIN	Bend	0 to page extent minus 1	B-coordinate for primitive end point	M
12–13	UBIN	HAXIS	0–32767	Length of ellipse X-axis (parallel to I-axis) for rounded corner on box	M
14–15	UBIN	VAXIS	0–32767	Length of ellipse Y-axis (parallel to B-axis) for rounded corner on box	M
16	UBIN	MH	X'00'–X'FF'	Integer multiplier for radius of full arc	M
17	UBIN	MFR	X'00'–X'FF'	Fractional multiplier for radius of full arc	M
18–19	CODE	DescID	X'0001'–X'FFFE'	Identifies a graphics descriptor	M
20	CODE	FGMix	X'02'	Foreground mixing rule: **X'02'** Overpaint	O
21				Reserved; must be zero	O
22	CODE	LineTpe	See AFP GOCA.	Line type for primitive boundary	O
23	CODE	LineWMH	See AFP GOCA.	Line width for primitive boundary: integral multiplier	O
24	CODE	LineWMFR	See AFP GOCA.	Line width for primitive boundary: fractional multiplier	O
25	CODE	PattSet	X'00'	Pattern set for primitive fill	O
26	CODE	PattSymb	See AFP GOCA.	Pattern symbol for primitive fill	O
27–28	SBIN	XMAJ	See AFP GOCA.	I-coordinate of arc major axis end point	O
29–30	SBIN	YMIN	See AFP GOCA.	B-coordinate of arc minor axis end point	O

Offset	Type	Name	Range	Meaning	M/O
31–32	SBIN	XMIN	See AFP GOCA.	I-coordinate of arc minor axis end point	O
33–34	SBIN	YMAJ	See AFP GOCA.	B-coordinate of arc major axis end point	O

Triplet X'7E' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Graphics Descriptor triplet.

ParmSpc   Specifies whether the triplet defines a complete primitive, or whether it defines only the starting parameters or only the ending parameters.

Value   **Description**

X'01'   All parameters. The triplet specifies all the parameters required to generate the primitive. For horizontal lines, the RCD IPos/Bpos parameters specify the start point and the Iend parameter specifies the ending inline position. For vertical lines, the RCD IPos/Bpos parameters specify the start point and the Bend parameter specifies the ending baseline position. For diagonal lines, the RCD IPos/Bpos parameters specify the start point and the Iend/Bend parameters specify the end point. For boxes, the RCD IPos/Bpos parameters specify the left top corner of the box and the Iend/Bend parameters specify the right bottom corner. For horizontal lines, the ParmSpc value is always assumed to be X'01'.

X'02'   Start parameters. For vertical lines, the RCD IPos/Bpos parameters specify the start point and the ending baseline position (BPos) is specified by a Graphics Descriptor triplet with ParmSpc = X'03' on an ensuing RCD with a matching Graphid. For boxes, the RCD IPos/Bpos parameters specify the left edge of the box and the Iend parameter specifies the right edge of the box. The ending BPos parameter is specified by a Graphics Descriptor triplet with ParmSpc = X'03' on an ensuing RCD with a matching Graphid.

For diagonal lines, both the start point and the ending inline position are specified in the start parameters. The RCD IPos/Bpos parameters specify the start point and the Iend value specifies the ending inline position. The ending baseline position is specified by a Graphics Descriptor triplet with ParmSpc = X'03' on an ensuing RCD with a matching Graphid.

If a logical page eject is processed while any lines or boxes are active (have been started but not ended), these lines and/or boxes are ended. The bottom margin is used as the ending baseline position. Note that the actual location of "bottom margin" on a page is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76.

X'03'   End parameters. The triplet specifies the ending baseline

Record Descriptor (RCD)

position for lines or boxes started by Graphics Descriptor Triplets on previous RCDs. The Graphid parameter specifies the lines or boxes ended by this triplet. The RCD BPos specifies the ending baseline position for lines and boxes.

All others
Reserved

Graphid   Specifies an identifier that is used to tie one or more start (ParmSpc = X'02') and end (ParmSpc = X'03') Graphics Descriptor Triplets together. An ending triplet ends all of the active lines and boxes that were started with a matching Graphid. Note that the start and end triplets must have the same orientation.

GrPrim   Specifies the graphics primitive that is to be generated. The primitives and their parameters are specified based on definitions in the AFP GOCA architecture. See the *IBM Graphics Object Content Architecture for Advanced Function Presentation*, S544-5498.

Value	Description
X'01'–X'03'	Line at current position. A straight line is generated between two points. The line is defined by the line type (LineTpe) and line width (LineWMH, LineWMFR) parameters in the descriptor. The color of the line is determined by the Color Specification (X'4E') triplet on the RCD. If two X'4E' triplets are specified, it is determined by the second triplet. If a X'4E' triplet is not specified, it is the presentation process default color.
	If the GrPrim value is X'01' (horizontal), the line is parallel to the I axis and ParmSpc is assumed to be X'01'. If the GrPrim value is X'02' (vertical), the line is perpendicular to the I axis. If the GrPrim value is X'03' (diagonal), the line is diagonal to the I axis. See ParmSpc for a description of the parameters used to draw the line.
X'04'	Full Arc at current position. A circle or ellipse is generated with center at the current RCD position. The ParmSpc parameter is ignored for this primitive.
	The color of the boundary line for the arc, if drawn, is determined by the *second* Color Specification (X'4E') triplet on the RCD, or by the first Color Specification triplet if only one is specified. If a X'4E' triplet is not specified, the color of the boundary line is the presentation process default color. The color of the fill pattern for the interior of the arc is determined by the *first* Color Specification (X'4E') triplet on the RCD; if a X'4E' triplet is not specified, the color of the fill pattern is the presentation process default color.
X'05'	Box at current position. A box is either specified by a single RCD or by a begin/end pair of RCDs. The box is generated with square corners or rounded

corners, depending on the value of the HAXIS and VAXIS parameters. If either parameter is zero, square corners are generated. If they are non-zero but equal, the corners are quadrants of a circle whose diameter is HAXIS. If they are non-zero and not equal, the corners are quadrants of an ellipse whose full axes are HAXIS and VAXIS.

The color of the boundary line for the box, if drawn, is determined by the *second* Color Specification (X'4E') triplet on the RCD, or by the first Color Specification triplet if only one is specified. If a X'4E' triplet is not specified, the color of the boundary line is the presentation process default color. The color of the fill pattern for the interior of the box is determined by the *first* Color Specification (X'4E') triplet on the RCD; if a X'4E' triplet is not specified, the color of the fill pattern is the presentation process default color.

See ParmSpc for more detail on how the box is drawn.

All others	Reserved

GraFlgs   Flags that specify how the primitive is generated. GraFlgs bits have the following definitions:

Bit 0   Fill. Indicates whether the interior of the primitive (the inside of the circle/ellipse or box) is to be filled with a colored pattern. The pattern is specified by the pattern set (PatSet) and pattern symbol (PatSymb) parameters in the descriptor and corresponds to one of the pattern symbols in the default pattern set defined in the AFP GOCA architecture. The color of the fill pattern is determined by the *first* Color Specification (X'4E') triplet on the RCD; if a Color Specification triplet is not specified, it is the presentation process default color.

This bit is ignored for the line primitive.

Value	Description
B'0'	Do not fill the interior of the primitive.
B'1'	Fill the interior of the primitive.

Bit 1   Boundary. Indicates whether the boundary of the primitive (circle/ellipse or box) is to be drawn with a line. The line is specified by the line type (LineTpe) and line width (LineWMH, LineWMFR) parameters in the descriptor.

Value	Description
B'0'	Do not draw the boundary of the primitive.
B'1'	Draw the boundary of the primitive.

Record Descriptor (RCD)

<table>
<tr><td></td><td>If drawn, the color of the boundary is determined by the <i>second</i> Color Specification (X'4E') triplet on the RCD, or by the first Color Specification triplet if only one is specified. If a X'4E' triplet is not specified, the color of the boundary line is the presentation process default color.</td></tr>
<tr><td></td><td>This bit is ignored for the line primitive, which is always drawn.</td></tr>
<tr><td>2–7</td><td>Reserved; all bits must be B'0'.</td></tr>
</table>

Iend
Specifies the I position of the end point for the primitive. This parameter is ignored if ParmSpc does not equal X'01' or X'02'.

Bend
Specifies the B position of the end point for the primitive. This parameter is ignored if ParmSpc does not equal X'01'. This parameter specifies a relative baseline position if the RCD specifies a relative baseline position.

HAXIS
Used only for the Box primitive to specify the length of the ellipse X-axis (parallel to the I-axis).

VAXIS
Used only for the Box primitive to specify the length of the ellipse Y-axis (parallel to the B-axis).

MH
Specifies the integer portion of the scale factor that is applied to the circle or ellipse defined by the XMAJ, XMIN, YMAJ, YMIN parameters in the descriptor.

MFR
Specifies the fractional portion of the scale factor that is applied to the circle or ellipse defined by the XMAJ, XMIN, YMAJ, YMIN parameters in the descriptor. A decimal point is implied between MH and MFR. The fractional portion of the scale factor is calculated by dividing MFR by 256. For example, if MFR=X'40', its decimal value is 64, which, divided by 256 results in a fractional component for the scale factor of 1/4.

For a circle, the radius is (MH•R + MFR•R), where R is the radius of the circle defined by the current arc parameters. For an ellipse, the major and minor axes are (MH•MAJ + MFR•MAJ) and (MH•MIN + MFR•MIN), where MAJ and MIN are the major and minor axes of the ellipse.

DescID
Specifies the ID of a graphics descriptor. The descriptor is defined by bytes 20–34 of this triplet. If the ID matches the ID of a graphics descriptor defined previously on an RCD for this page, the previous descriptor is used regardless of whether bytes 20–34 are specified in the current descriptor. If the ID does not match a previously-defined ID, bytes 20–34 must be specified and define the graphics descriptor that is to be identified with this ID and that is to be used to generate this primitive. If this Graphics Triplet specifies Parmspc = X'03', this parameter and all bytes in the graphics descriptor are ignored. The valid range for the ID is X'0001'—X'FFFE'.

For a given page, the presentation services program collects all graphics primitives that have the same graphics descriptor ID and the same orientation and groups them into a single graphics object.

The origin for the graphics object area is one of the four corners of the page as determined by the text orientation specified in the

RCD (TxtOrent) and is therefore coincident with the current (I,B) origin. The rotation of the graphics object area about the page X_p-axis matches the rotation of the current text (I,B) coordinate system. For example, with a (90°,180°) text orientation, the object area rotation is 90°. The extents of the object area are determined by the extents of the page.

The position of the graphics primitive in the (I,B) coordinate system therefore maps to the same position in the object area (X_{oa},Y_{oa}) coordinate system. This position in turn is projected to a graphics window whose upper left corner is at the graphics presentation space (GPS) origin, and whose extents match those of the object area. The upper left corner of the graphics presentation space window is therefore also coincident with the current (I,B) origin. The mapping between graphics window and object area is *position and trim*.

For example, if the RCD specifies a (90°,180°) text orientation, the upper left corner of the graphics window is at the top-right corner of the page, and graphics primitives in this object are rotated 90° with respect to the page X_p axis. The X-extent of the graphics window is the Y_p-extent of the page, and the Y-extent of the graphics window is the X_p-extent of the page.

The units of measure for the graphics presentation space and for the graphics object area are the same as those defined on the page (X_p,Y_p) presentation space in the PGD structured field of the Active Environment Group (AEG) for the Data Map.

FGMix Specifies how the graphics primitive mixes with underlying data. The only mixing supported is X'02' (Overpaint). This parameter is specified in an AFP GOCA object with the GDD Set Current Defaults instruction and the Set MIX drawing order.

LineTpe Specifies the type of line that is drawn. For supported values, see the *IBM Graphics Object Content Architecture for Advanced Function Presentation*, S544-5498. This parameter is specified in an AFP GOCA object with the GDD Set Current Defaults instruction and the Set Line Type drawing order.

LineWMH Specifies the width of the line that is drawn by defining the integer portion of the normal line width multiplier. For supported values, see the *IBM Graphics Object Content Architecture for Advanced Function Presentation*, S544-5498. This parameter is specified in an AFP GOCA object with the GDD Set Current Defaults instruction, the Set Line Width drawing order, and the Set Fractional Line Width drawing order.

LineWMFR Specifies the width of the line that is drawn by defining the fractional portion of the normal line width multiplier. For supported values, see the *IBM Graphics Object Content Architecture for Advanced Function Presentation*, S544-5498. This parameter is specified in an AFP GOCA object with the Set Fractional Line Width drawing order.

PatSet Specifies the pattern set that contains the pattern symbols used to fill the interior of a primitive. The only value supported is X'00'—default pattern set. This parameter is specified in an AFP GOCA object with the GDD Set Current Defaults instruction and the Set Pattern Set drawing order.

PatSymb Specifies the pattern symbol within the current pattern set that is used to fill the interior of a primitive. For supported values, see the *IBM Graphics Object Content Architecture for Advanced Function Presentation*, S544-5498. This parameter is specified in an AFP GOCA object with the GDD Set Current Defaults instruction and the Set Pattern Symbol drawing order.

XMAJ, XMIN, YMAJ, YMIN
Together these parameters define a circle or ellipse on the (I,B) coordinate system. The center of the arc is at the (I,B) origin. When this triplet is used to generate a circle or ellipse, the center is moved to the (I,B) position specified by the RCD, and the arc is scaled by the MH.MFR scale factor. Specifically, the four parameters specify the following:

XMAJ I coordinate of major axis endpoint
YMAJ B coordinate of major axis endpoint
XMIN I coordinate of minor axis endpoint
YMIN B coordinate of minor axis endpoint

These parameters are specified in an AFP GOCA object with the GDD Set Current Defaults instruction and the Set Arc Parameters drawing order.

Notes:

1. The last 15 bytes (bytes 20–34) in this triplet are optional as a group. That is, either they are all specified or none are specified. If the descriptor ID is intended to match a previously-defined descriptor ID, these bytes should not be specified.

2. The X'22', X'69', and X'6C' triplets are ignored when this triplet is specified on an RCD.

Object Reference Qualifier (X'89') Triplet
The Object Reference Qualifier triplet is used to specify whether the name of an object is retrieved from the input data or retrieved using normal methods. If the name is to be retrieved from the input data, that name overrides any ObjName field and any Fully Qualified Name (type X'01') triplet that would normally be used to select an object. This triplet may occur once on an RCD structured field.

Color Management Resource Descriptor (X'91') Triplet
The Color Management Resource Descriptor triplet specifies the processing mode and scope for a Color Management Resource (CMR). This triplet is mandatory when the RCD references a Color Management Resource (CMR) with the FQN type X'DE' triplet, in which case this triplet specifies the processing mode for the CMR, and must occur once for each FQN type X'DE' specified. It is ignored in all other cases. This triplet must immediately follow the FQN type X'DE' triplet that specifies the CMR name.

Rendering Intent (X'95') Triplet
The Rendering Intent (X'95') triplet is a MO:DCA triplet. For the formal definition of this triplet, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

The Rendering Intent triplet specifies the rendering intent parameter, which is used to modify the final appearance of color data. This parameter is based on the rendering intents defined by the International Color Consortium (ICC). This triplet is optional and may occur once when a Graphics Descriptor (X'7E') triplet is specified on the RCD. If this triplet is specified more than once, only the first is used. This triplet specifies the rendering intent that is to be used when presenting the Graphics object that is generated with this structured field. Only the rendering intent that applies to the object type of the referenced object is used; the other rendering intents are ignored.

Triplet X'95' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	10	Length of the triplet, including Tlength	M
1	CODE	Tid	X'95'	Identifies the Rendering Intent triplet	M
2-3				Reserved; must be zero	M
4-6			X'FFFFFF'	Not used	M
7	CODE	GOCARI	X'00'-X'03', X'FF'	Rendering intent for AFP GOCA objects: **X'00'** perceptual **X'01'** media-relative colorimetric **X'02'** saturation **X'03'** ICC-absolute colorimetric **X'FF'** not specified	M
8-9				Reserved; must be zero	M

Triplet X'95' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the Rendering Intent triplet.

GOCARI   Specifies the rendering intent for GOCA objects. Valid values are the following.

Value	Description
X'00'	**Perceptual**. Gamut mapping is vendor-specific, and colors are adjusted to give a pleasing appearance. This intent is typically used to render continuous-tone images.
X'01'	**Media-relative colorimetric**. In-gamut colors are rendered accurately, and out-of-gamut colors are mapped to the nearest value within the gamut. Colors are rendered with respect to the source white point and are adjusted for the media white point. Therefore colors printed on two different media with different white points do not match colorimetrically, but may match visually. This intent is typically used for vector graphics.
X'02'	**Saturation**. Gamut mapping is vendor-specific, and

Record Descriptor (RCD)

		colors are adjusted to emphasize saturation. This intent results in vivid colors and is typically used for business graphics.
	X'03'	**ICC-absolute colorimetric**. In-gamut colors are rendered accurately, and out-of-gamut colors are mapped to the nearest value within the gamut. Colors are rendered only with respect to the source white point, and are not adjusted for the media white point. Therefore colors printed on two different media with different white points should match colorimetrically, but may not match visually. This intent is typically used for logos.
	X'FF'	The rendering intent is not specified.
	All others	Reserved

XML Descriptor (XMD)

The XML Descriptor structured field contains information, such as data position, text orientation, font selection, field selection, and conditional processing identification, used to format XML data that consists of text delimited by start and end tags.

Note: The XMDs in a Data Map are numbered sequentially, starting with 1.

XMD (X'D3A68E') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A68E'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O
0	CODE	ElmType	X'00'–X'03'	Element Type	M
1–3	BITS	XMDFlgs			M
Bit 0–1:			B'00'	Reserved; must be zero	
Bit 2:			B'0'–B'1'	Generate Inline Position	
Bit 3:			B'0'–B'1'	Generate Baseline Position	
Bit 4:			B'0'–B'1'	Generate Font Change	
Bit 5:			B'0'–B'1'	Generate Suppression	
Bit 6:			B'0'–B'1'	Field XMD	
Bit 7:			B'0'–B'1'	Use Fixed Data	
Bit 8–9:			B'00'	Reserved; must be zero	
Bit 10:			B'0'–B'1'	Attribute XMD	
Bit 11:			B'0'–B'1'	Conditional Processing XMD	
Bit 12:			B'0'–B'1'	Relative Inline Position	
Bit 13:			B'0'–B'1'	Relative Baseline Position	
Bit 14–15:			B'00'	Reserved, must be zero	
Bit 16:			B'0'–B'1'	New Page	
Bit 17:			B'0'–B'1'	Print Page Number	
Bit 18:			B'0'–B'1'	Reset Page Number	
Bit 19:			B'0'–B'1'	Group Indicator	
Bit 20:			B'0'–B'1'	Field Delimiter Size	
Bit 21:			B'0'–B'1'	Use Start Tag	
Bit 22:			B'0'	Reserved, must be zero	
Bit 23:			B'0'–B'1'	Header/Trailer Continued	
4				Reserved; must be zero	M
5–6	UBIN	IPos	0 to page extent minus 1	Absolute inline position	M
	SBIN		X'8000'–X'7FFF'	Relative inline position	
7–8	UBIN	BPos	0 to page extent minus 1	Relative baseline position	M
	SBIN		X'8000'–X'7FFF'		
9–12	CODE	TxtOrent		Text (I,B) Orientation	M
			X'0000 2D00'	0,90 degrees	
			X'2D00 5A00'	90,180 degrees	
			X'5A00 8700'	180,270 degrees	
			X'8700 0000'	270,360 degrees	

XML Descriptor (XMD)

Offset	Type	Name	Range	Meaning	M/O
13	CODE	FntLID	X'01'–X'FE'	Primary font local ID	M
			X'FF'	Presentation system default font	
14–15	UBIN	FLDxmd	X'0000'–X'FFFF'	Field XMD Pointer	M
16–17				Reserved; must be zero	M
18–25	CHAR	SupName		Suppression token name. A value of all X'F's	M
26				Reserved; must be zero	M
27–29	UBIN	DataStrt	X'00000000'–X'00007FFF'	Data start position	M
31–32	UBIN	DataLgth	X'0000'–X'FFFE'	Data length	M
			X'FFFF'	Place remaining bytes	
33–34	UBIN	CONDxmd	X'0000'–X'FFFF'	Conditional Processing XMD Pointer	M
35	CODE	SubpgID	X'00'	Subpage ID (Always X'00' for XMDs)	M
36–37	CODE	CCPID	X'0000'–X'FFFF'	CCP Identifier	M
38–39	UBIN	Pgno	X'0001'–X'FFFF'	Starting page number	M
			X'0000'	Not specified	
40–41	UBIN	ESpac	0 to page extent minus 1	End Space	M
42	CODE	Align	X'00'–X'01'	Field Alignment	M
43–44	CODE	FldDelim	X'0000'–X'FFFF'	Field Delimiter	M
45–46	UBIN	Fldno	X'0001'–X'FFFF'	Field Number	M
			X'0000'	Not specified	
47–48	UBIN	AdBIncr	X'0000'–X'FFFF'	Additional baseline increment	M
49–61				Reserved; must be zero	M
62–*n*		Triplets		See "XMD Semantics" for triplet applicability.	O

XMD Semantics

The XMD uses many parameters that are defined for the LND and/or RCD. The definition of such parameters is deferred to the LND and/or RCD. When such definitions are applied to the XMD, the term "LND" and/or "RCD" should be read as "XMD", and the byte offsets of the parameters should be adjusted to the XMD.

There are four types of XMDs:

- Element XMDs, which define the processing of the data content of the XML element or define a default page header or traile.
- Field XMDs, which define the processing for an input field or specify constant text or graphics.
- Attribute XMDs, which define the processing for attributes specified in an XML start tag. Attribute XMDs are a special type of Field XMD.
- Conditional Processing XMDs, which specify the Conditional Processing associated with an input element.

The Field XMDs, Attribute XMDs, and Conditional Processing XMDs associated with an Element XMD are chained to that XMD using XMD number pointers. An

XMD is assumed to be an Element XMD if neither the Field XMD, Attribute XMD nor the Conditional Processing XMD bits are on in the XMDFlgs byte.

ElmType　　　Element Type. This parameter is ignored on XMDs other than Element XMDs.

Value　**Description**

X'00'　Body Element: This XMD does not have any special header or trailer properties associated with it and is used to format any input elements with a matching Qualified Tag.

X'01'　Page Header Element: This XMD is used to automatically print a header (such as the current customer name) at the beginning of each new page. The baseline position of this XMD can be anywhere on a logical page and can be specified as relative. If an input element is received with a matching Qualified Tag, that input element is not presented on receipt but is saved as the active page header. If no Qualified Tag is specified for an XMD that has the Page Header element type, it is assumed to be a default Page Header XMD (Only one default Page Header XMD can be specified in a Data Map and no input element data is processed with a default XMD). See the logical page eject processing section for page header and New Page processing details.

Note: Once a Page Header XMD is processed, the header element is saved for the duration of the document by the presentation services program. Whenever the same Data Map is re-invoked for that document, this saved header element is always presented with each page generated with the Data Map.

X'02'　Page Trailer Element: This XMD is used to automatically print a trailer (for example, a footnote and/or page number) at the end of each page. The baseline position of this XMD can be anywhere on a logical page and can be specified as relative. If an input element is received with a matching Qualified Tag, that input element is not presented on receipt but is saved as the active page trailer. If no Qualified Tag is specified for an XMD that has the Page Trailer element type, it is assumed to be a default Page Trailer XMD (Only one default Page Trailer XMD can be specified in a Data Map, and no input element data is processed with a default XMD). See the logical page eject processing section for page trailer and New Page processing details.

Note: Once a Page Trailer XMD is processed, the trailer element is saved for the duration of the document by the presentation services program. Whenever the same Data Map is re-invoked for that document, this saved trailer element is always presented with each page generated with the Data Map.

X'03'　Group Header Element: This XMD is used to automatically print a group header (for example, column headings for a group of elements) on a page. Note that the group header

XML Descriptor (XMD)

is not actually printed and causes no action until a Body Element XMD with Group Indicator (XMDFlgs bit 19) set to B'1' is processed for the page. The baseline position of this XMD can be specified as relative. If an input element is received with a matching Qualified Tag, that input element is saved as the active group header and then presented. If that input element and/or XMD causes a page eject, that input element is used as the active group header for the new page. See the logical page eject processing section for active group header and New Page processing details.

Note: Once a Group Header XMD is processed and is still active when leaving the Data Map, the group header element is saved by the presentation services program. Whenever the same Data Map is re-invoked, this saved group header element is presented again if the first body element after re-invoking the Data Map selects a Body Element XMD that has the Group Indicator on.

Note: The formation of the Page Header, Group Header, or Page Trailer may require element content from more than one element. This is accomplished by the use of XMDFlgs bit 23 (Header/Trailer Continued). Refer to the description of the Header/Trailer Continued flag for more information about continued headers and trailers.

XMDFlgs   Flag bits

Bits 0–5   For a definition of these flag bits see "Line Descriptor (LND)" on page 104. LND flag bits 0–1 are reserved in the XMD.

Bit 6   Field XMD

Value	Description
B'0'	If this bit and bits 10 and 11 are all B'0', this is an Element XMD.
B'1'	This is a Field XMD. If this bit and bit 11 is on, it is an error.

Bits 7–9   For a definition of these flag bits see "Line Descriptor (LND)" on page 104. LND flag bit 9 is reserved in the XMD.

Bit 10   Attribute XMD

Value	Description
B'0'	If this bit and bits 6 and 11 are all B'0', this is an Element XMD.
B'1'	This is an Attribute XMD. If this bit is on, bit 6 must also be on since an Attribute XMD is a special type of Field XMD. If this bit and bit 11 is on, it is an error.

When processing an attribute of the XML Start tag, the processor matches the attribute name to the

name specified in the XML Name triplet which was specified for this XMD. Once a match is found, the attribute value is used as the data to be processed by this XMD.

Note: The attribute value is the quoted string not containing the quotation mark used as a delimiter for that string.

Note: The color for the data presented by this XMD is always the color specified by the Color Specification (X'4E') triplet, if specified. If this triplet is not specified, the data is presented in the presentation process default color.

Bit 11 For a definition of this flag bit see "Line Descriptor (LND)" on page 104.

Bit 12 Relative Inline Position

Indicates whether the inline position specified in bytes 5–6 of this XMD is an absolute position or a relative position. If an absolute inline position is specified, it is measured as a positive offset in the inline direction from the current text (I,B) coordinate system origin. If a relative position is specified, it is measured as a positive or negative offset from the current inline position using the current text (I,B) coordinate system, which is defined by the text orientation specified in bytes 9–12.

Value	Description
B'0'	The inline position specified in bytes 5–6 is an absolute position.
B'1'	The inline position specified in bytes 5–6 is a relative position.

The following restriction applies to relative inline positioning:
• The text orientation of an XMD that specifies relative inline positioning must be the same as the text orientation of the XMD that defines the inline position from which the relative offset is measured.

Bits 13–15 For a definition of these flag bits see "Line Descriptor (LND)" on page 104.

Bits 16–20 For a definition of these flag bits see "Record Descriptor (RCD)" on page 127.

Bit 21 Use Start Tag

Selects the Start tag (including the angle brackets '<' and '>') as the data field to be used for presentation.

XML Descriptor (XMD)

Value	Description
B'0'	Do not select the Start Tag.
B'1'	Select the Start Tag.

This function is restricted to Field XMDs; it is ignored on all other XMDs.

Bit 22   Reserved; must be zero

Bit 23   Header/Trailer Continued

Indicates that this XMD is a continuation of a Page Header, Group Header or Page Trailer definition. The formation of the Page Header, Group Header or Page Trailer may require the element content from more than one element. This is accomplished by specifying this continuation indicator along with specifying the appropriate ElmType parameter to identify which type of header or trailer that is to be continued. If the header or trailer has not started, this header or trailer XMD starts the header or trailer regardless of the setting of this flag.

For Page Headers and Page Trailers, the elements that are used to build the continued header or trailer do not need to be contiguous in the XML data, but do need to be on the same page. If a page break occurs before a continued header or trailer is reached, the continued header or trailer acts as a new header or trailer. If the continued header or trailer is on the same page, the XMD structured fields used to print the various pieces of the continued header or trailer are processed as though the elements were specified contiguously. This means any relative positioning specified is relative to data placed using a previous header or trailer XMD, if any.

For Group Headers, the elements that are used to build the continued header do not need to be contiguous, but cannot have body elements placed between the pieces of the Group Header. If body elements are placed between the pieces of a continued Group Header, the Group Header is not continued but acts as a new Group Header.

Value	Description
B'0'	Not a continuation XMD.
B'1'	Continued Header or Trailer definition. If the header or trailer has started, data continues to be collected to form the header or trailer.

This function is ignored on Body Element XMDs.

IPos   Inline Position

Relative Inline Position for Element, Field and Attribute XMDs: If the inline position is relative, the offset is relative to the current inline position. If there is no prior XMD, the relative inline position is relative to the left margin.

> **Note:** Data must not exceed the boundaries of the page, which are defined in the Page Descriptor (PGD) structured field. If the new print position is outside these boundaries, printing of the page stops.

Note that the actual location of the 'left margin' on a page is affected by the text orientation; see "Margin Definition (X'7F') Triplet" on page 76.

BPos
Baseline Position

See "Record Descriptor (RCD)" on page 127.

TxtOrent
Text (I,B) Orientation

See "Line Descriptor (LND)" on page 104.

FntLID
Primary Font Local Identifier

See "Line Descriptor (LND)" on page 104.

FLDxmd
The XMD number of a Field XMD.

See "Record Descriptor (RCD)" on page 127.

SupName
Suppression token name

See "Line Descriptor (LND)" on page 104.

Datastrt
Data Start Position

See "Line Descriptor (LND)" on page 104.

Datalgth
Data Length

See "Line Descriptor (LND)" on page 104.

CONDxmd
The XMD number of a Conditional Processing XMD.

See "Record Descriptor (RCD)" on page 127.

SubPgID
Subpage Identifier

See "Line Descriptor (LND)" on page 104.

CCPID
CCP Identifier

See "Line Descriptor (LND)" on page 104.

Pgno
Page Number.

See "Record Descriptor (RCD)" on page 127.

ESpac
End Space.

See "Record Descriptor (RCD)" on page 127.

Align
Field Alignment.

See "Record Descriptor (RCD)" on page 127.

FldDelim
Field Delimiter.

See "Record Descriptor (RCD)" on page 127.

> **Fldno** Field Number.
>
> See "Record Descriptor (RCD)" on page 127.
>
> **AdBIncr** Additional baseline increment.
>
> See "Record Descriptor (RCD)" on page 127.
>
> **Triplets** See "Extended Resource Local Identifier (X'22') Triplet" on page 157, "Color Specification (X'4E') Triplet" on page 157, "Bar Code Symbol Descriptor (X'69') Triplet" on page 157, "Additional Bar Code Parameters (X'7B') Triplet" on page 157, "Resource Object Include (X'6C') Triplet" on page 158, "Graphics Descriptor (X'7E') Triplet" on page 158, and "XML Name (X'8A') Triplet."

Logical Page Eject Processing

See "Record Descriptor (RCD)" on page 127.

XMD Triplets

Fully Qualified Name (X'02') Triplet

This triplet is optional and may occur one or more times when a Bar Code Symbol Descriptor (X'69') triplet or a Graphics Descriptor (X'7E') triplet is specified on the XMD. The Fully Qualified Name type that may appear is X'DE'- *Data Object External Resource Reference*. The FQN triplet specifies the external identifier of a Color Management Resource (CMR) object that is used for the Bar Code object or Graphics object being generated. The identifier is used by the presentation system to locate the resource object in the resource hierarchy. The identifier is a character-encoded name which must be specified using FQNFmt = X'00'. The encoding for the external identifier of the CMR must be UTF-16BE.

See "Fully Qualified Name (X'02') Triplet" on page 113.

XML Name (X'8A') Triplet

Architecture Note: The XML Name triplet is registered in MO:DCA as a private-use triplet since it is used only in the PageDef object, which is not a MO:DCA object.

This triplet is used to build a Qualified Tag. A Qualified Tag is built by concatenating the names specified on each XML Name triplet in the order the triplets are specified. Each XML Name used in the concatenation is separated by a single space character.

Note: Multiple XML Name triplets do not have to be contiguous.

This triplet is mandatory on Body and Group Header Element XMDs and must occur at least once to build a Qualified Tag.

This triplet is optional for Page Header and Page Trailer Element XMDs. If omitted, this Page Header or Page Trailer XMD is the default Page Header or Page Trailer XMD.

This triplet is mandatory on Attribute XMDs and must occur once to identify the name of an attribute specified on an XML start tag. If this triplet occurs more than once on an Attribute XMD, only the first occurrence is used.

This triplet is ignored on XMDs other than Element XMDs and Attribute XMDs.

The name specified in this triplet must be encoded using the encoding specified on the Encoding Scheme ID (X'50') triplet of the BDM structured field.

Triplet X'8A' Syntax:

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	Tlength	5–254	Length of the triplet, including Tlength	M
1	CODE	Tid	X'8A'	Identifies the XML Name triplet	M
2–3				Reserved; must be zero	M
4–*n*	CHAR	XMLName		Name of Start Tag or Attribute in XML data.	M

Triplet X'8A' Semantics:

Tlength   Contains the length of the triplet.

Tid   Identifies the XML Name triplet.

XMLName   Specifies the name of the Start tag or the name of an attribute of a Start tag contained in the XML data. This XMLName is used to build Qualified Tags when used on Element XMDs.

Extended Resource Local Identifier (X'22') Triplet

This triplet is optional and may occur one or more times to reference an IOB structured field in the PageDef. This triplet is ignored if the Graphics Descriptor (X'7E') triplet is specified on the XMD.

See "Extended Resource Local Identifier (X'22') Triplet" on page 114.

Color Specification (X'4E') Triplet

This is an optional triplet that specifies the color for text processed by this XMD, bar code generated by this XMD, and for graphics generated by this XMD.

See "Color Specification (X'4E') Triplet" on page 138.

Bar Code Symbol Descriptor (X'69') Triplet

This is an optional triplet and may occur once. If this triplet is specified more than once, only the first is used. This triplet specifies that the data selected by the descriptor is to be presented as a bar code symbol. This triplet is ignored if the Graphics Descriptor (X'7E') triplet is specified on the XMD.

See "Bar Code Symbol Descriptor (X'69') Triplet" on page 116. Note that the LND/RCD/XMD parameters used by this triplet may be at different offsets in the LND, RCD and XMD.

Additional Bar Code Parameters (X'7B') Triplet

This is an optional triplet that specifies additional parameters for non-linear bar code symbologies (for example, 2D bar codes). This triplet may occur one or more times when a Bar Code Symbol Descriptor (X'69') triplet is specified. This triplet is ignored if the Graphics Descriptor (X'7E') triplet is specified on the XMD.

See "Additional Bar Code Parameters (X'7B') Triplet" on page 121.

Resource Object Include (X'6C') Triplet

This is an optional triplet that identifies an overlay or page segment object to be presented on the page at a specified position. Multiple Resource Object Include triplets may be specified on the same XMD. This triplet is ignored if the Graphics Descriptor (X'7E') triplet is specified on the XMD.

Graphics Descriptor (X'7E') Triplet

This is an optional Field XMD triplet and may occur once. If this triplet is specified more than once, only the first is used. Text input and fixed data text are ignored on a Field XMD that specifies a Graphics Descriptor Triplet. When present, the Graphics Descriptor Triplet specifies the generation of a graphics primitive. The triplet may specify the complete primitive, or the start of the primitive, or the end of the primitive. This triplet is ignored on XMDs other than Field XMDs.

Color Management Resource Descriptor (X'91') Triplet

The Color Management Resource Descriptor triplet specifies the processing mode and scope for a Color Management Resource (CMR). This triplet is mandatory when the XMD references a Color Management Resource (CMR) with the FQN type X'DE' triplet, in which case this triplet specifies the processing mode for the CMR, and must occur once for each FQN type X'DE' specified. It is ignored in all other cases. This triplet must immediately follow the FQN type X'DE' triplet that specifies the CMR name.

Rendering Intent (X'95') Triplet

The Rendering Intent triplet specifies the rendering intent parameter, which is used to modify the final appearance of color data. This parameter is based on the rendering intents defined by the International Color Consortium (ICC). This triplet is optional and may occur once when a Graphics Descriptor (X'7E') triplet is specified on the XMD. If this triplet is specified more than once, only the first is used. This triplet specifies the rendering intent that is to be used when presenting the Graphics object that is generated with this structured field. Only the rendering intent that applies to the object type of the referenced object is used; the other rendering intents are ignored.

Appendix A. Corequisite and Other Publications

Several other publications may help you understand the licensed programs used with the data streams described in this book.

IBM Architecture Publications

Title	Order Number
Bar Code Object Content Architecture Reference	S544-3766
Color Management Object Content Architecture Reference	S550-0511
Font Object Content Architecture Reference	S544-3285
Image Object Content Architecture Reference	SC31-6805
Intelligent Printer Data Stream Reference	S544-3417
Graphics Object Content Architecture Reference	SC31-6804
Mixed Object Document Content Architecture Reference	SC31-6802
Presentation Text Object Content Architecture Reference	SC31-6803
Graphics Object Content Architecture for Advanced Function Presentation Reference	S544-5498
Character Data Representation Architecture Reference	SC09-2190

You can order any of these architecture publications separately, or order the first seven publications using SBOF-6179.

IBM Advanced Function Presentation Publications

Title	Order Number
Guide to Advanced Function Presentation. Contains a comprehensive overview of AFP and AFP concepts.	G544-3876
Advanced Function Presentation: Programming Guide and Line Data Reference	S544-3884
Advanced Function Presentation: Printer Information. Contains detailed characteristics about IBM's page printers.	G544-3290
IBM Advanced Function Presentation Fonts: Technical Reference for Code Pages	S544-3802
IBM Advanced Function Presentation Fonts: Technical Reference for IBM Expanded Core Fonts	S544-5228
IBM Advanced Function Presentation Fonts: Font Summary	G544-3810
Overlay Generation Language/370: User's Guide and Reference. Contains information about the OGL product that is used to create AFP overlays.	S544-3702
Page Printer Formatting Aid User's Guide and Reference. Contains information about the PPFA product that is used to create AFP page definitions and form definitions.	G544-3181
Advanced Function Presentation Workbench for Windows: Using the Viewer Application. Contains information about using it with AFP API.	G544-3813

Title	Order Number
Advanced Function Presentation Conversion and Indexing Facility: Application Programming Guide. Contains information about using ACIF.	G544-3824
Advanced Function Presentation: Toolbox for Multiple Operating Systems User's Guide	G544-5292
AFP API Programming Guide and Reference. Contains information for using the AFP Application Programming Interface.	S544-3872
Printing and Publishing Collection Kit. Contains the online, softcopy version of most of the books referred to in this chapter.	SK2T-2921

IBM ImagePlus® Publications

Title	Order Number
IBM Systems Application Architecture (SAA) ImagePlus Online Library (CD-ROM)	SK2T-2131
ImagePlus MVS/ESA General Information Manual	GC31-7537
AS/400 ImagePlus General Information Manual	GC38-2027
SAA ImagePlus/2 General Information Manual	GC28-8173

IBM Graphics and Image Publications

Title	Order Number
GDDM, 5748-XXH: General Information Manual. Contains a comprehensive overview of graphics and image support for OS/390, VM, VSE and OS400 systems.	GC33-0100
Introducing GDQF. Contains a comprehensive overview of Graphic Query and Display Facilities for complex manufacturing graphics, image, and publishing products.	GH52-0249
OS/2 Presentation Manager GPI. Contains a description of the PM Graphic Programming Interface.	G362-0005

Print Services Facility Publications

Title	Order Number
Print Services Facility/MVS: Application Programming Guide	S544-3673
Print Services Facility for z/OS: Introduction	G550-0430
Print Services Facility for z/OS: User's Guide	S550-0435
Print Services Facility/VM: Application Programming Guide	S544-3677
Print Services Facility/VSE: Application Programming Guide	S544-3666
Print Services Facility/2: Getting Started	G544-3767
IBM AIX Print Services Facility/6000: Print Services Facility for AIX Users	G544-3814
AS/400 Information Directory	GC21-9678
IBM eServer™ iSeries Information Center, "Printing" topic	http://publib. boulder.ibm.com /iseries/

Infoprint Manager Publications

Title	Order Number
Infoprint Manager for AIX Publications (CD-ROM)	SK2T-9266
Infoprint Manager for Windows Publications (CD-ROM)	SK2T-9288

Infoprint Server Publications

Title	Order Number
Infoprint Server for iSeries: Introduction and Planning Guide	G544-5744
Infoprint Server for iSeries: User's Guide	G544-5745
z/OS Infoprint Server: Introduction	S544-5742
z/OS Infoprint Server: User's Guide	S544-5746

Infoprint Font Publications

Title	Order Number
IBM Infoprint Fonts: Font Summary. Describes font concepts and contains font summary tables for Expanded Core Fonts, DBCS Core Fonts, and Simulation Fonts and a table listing code pages.	G544-5846
IBM Infoprint Fonts: Introduction to Type Transformer and Utilities for Windows	G544-5853
Printing Systems Information Center, "Infoprint Fonts" topic	http://publib. boulder.ibm.com /infocenter/ printer /v1r1/index.jsp

Related Publications

Appendix B. Document and Resource Object Diagrams

This appendix contains diagrams of the elements that make up a data stream accepted by PSF servers in the AFP environment. Some PSF servers may accept additional elements and objects. For more information, see Appendix D, "System Support Information," on page 183. For the formal definition of all valid AFP (MO:DCA-P) object structures, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Unless otherwise noted, the objects and structured fields are presented from left to right in the order in which they must appear.

The following symbols apply to the syntax structures in this appendix:

|	
*	An asterisk indicates optional structured fields or objects. Those not marked are required.
S	The letter "S" indicates structured fields or objects that can appear more than once in the document. Those not marked can appear only once in the document.
`xx object`	A shaded box indicates objects that contain structured fields or other objects.
Other	All other symbols are explained in the figures.

Diagrams
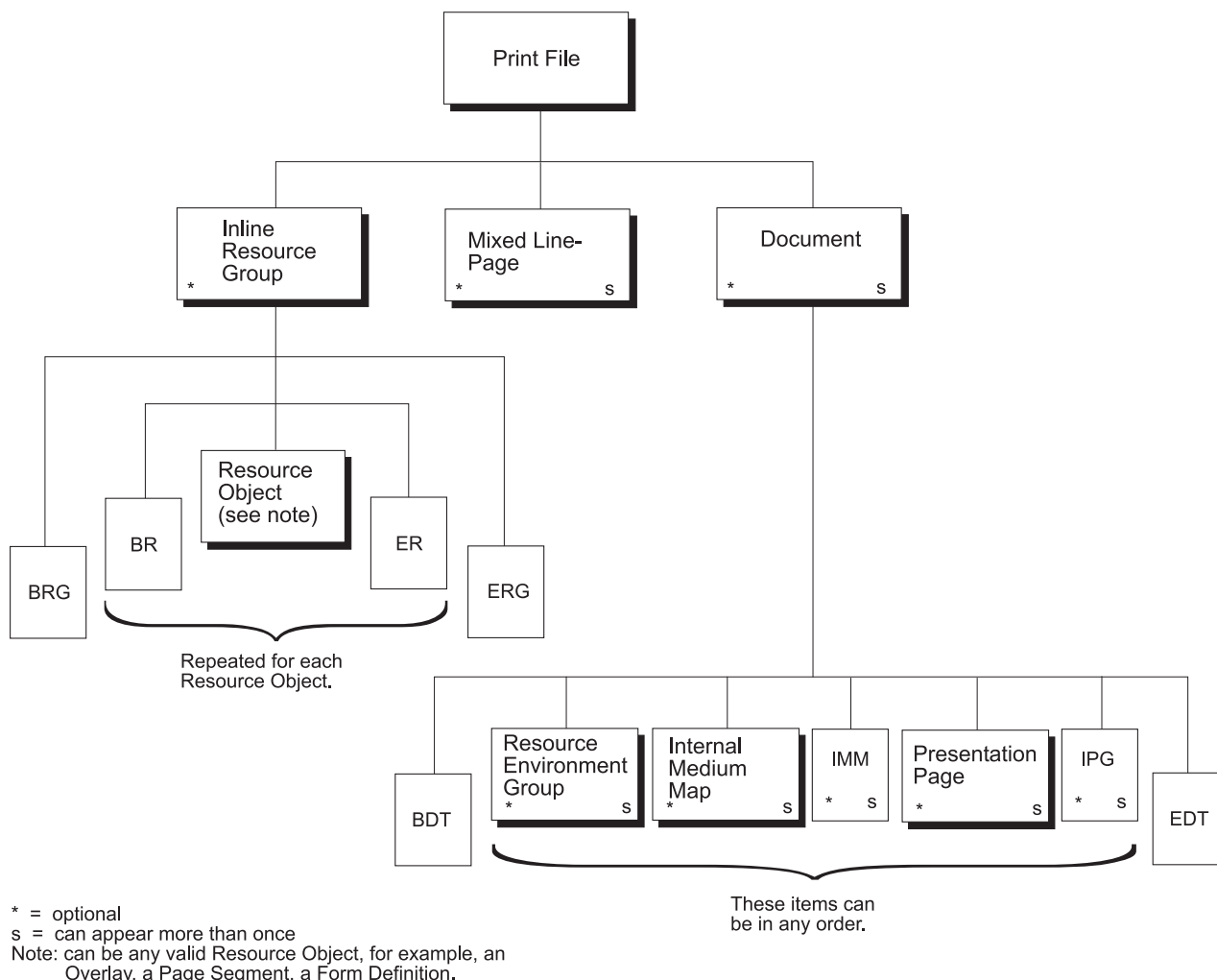
Figure 30. Structure of a Print File

Notes:

1. The mixed line-page documents and composed documents can occur in any order following the inline resource group.

2. Each AFP (MO:DCA-P) document may optionally be preceded by a single document index that is implicitly tied to the document and that indexes the document. For the formal definition of the MO:DCA-P document index see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

3. An AFP (MO:DCA-P) document may contain Link Logical Element (LLE) structured fields following the BDT, and may also group presentation pages into named page groups. MO:DCA-P page groups may in turn contain Tag Logical Element (TLE) structured fields following BNG. These structures do not affect the presentation of the document. To see which AFP print servers support these structures, see Appendix D, "System Support Information," on page 183. For the formal definition of these structures, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

4. If a Medium Map is included internal (inline) to the document, it is activated by immediately following it with an IMM that explicitly invokes it, otherwise the internal Medium Map is ignored. An IMM that does not follow an internal

Medium Map may not invoke an internal Medium Map elsewhere in the document and is assumed to reference a Medium Map in the current Form Definition.
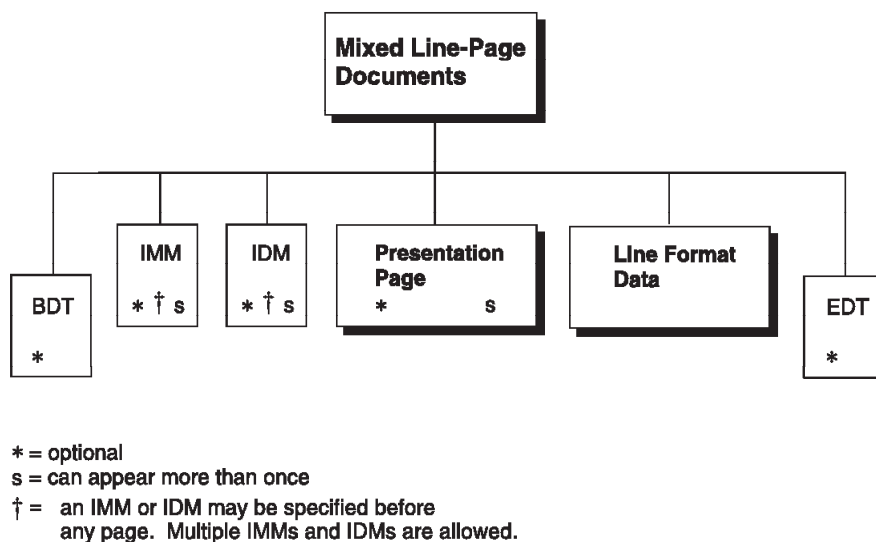

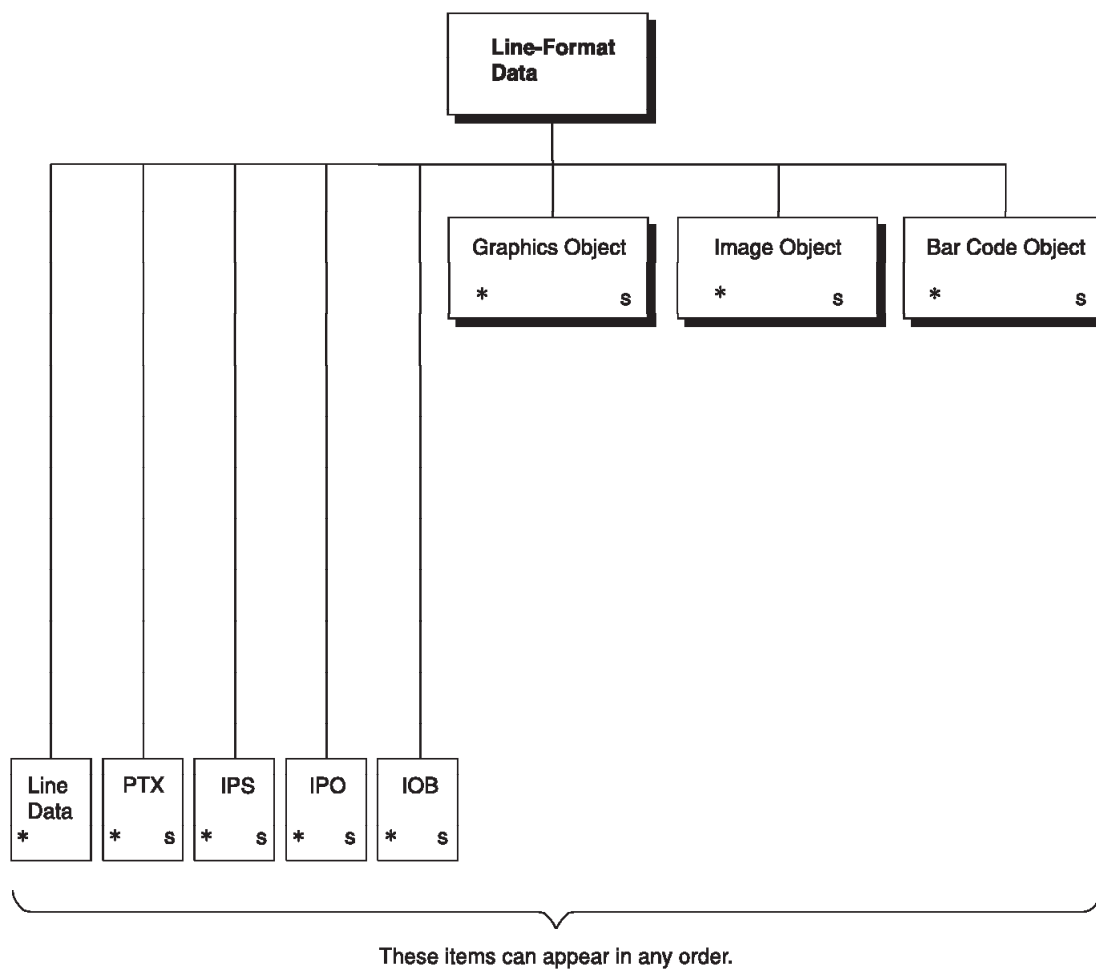*Figure 31. Structure of a Mixed Line-Page Document*

> **Note:** The No Operation (NOP) structured field may appear anywhere in a mixed document and thus is not listed in the structured field groupings.

Diagrams

Figure 32. Structure of a Presentation Page Object

The diagram shows:

Top box: Presentation Page

Second row (children of Presentation Page), with these items able to be in any order: Active Environment Group, Presentation Text Object (* s), Image Object (* s), Graphics Object (* s), Bar Code (* s), Object Container (* s), IOB (* s), IPS (* s), IPG (*), IPO (* s)

BPG and EPG brackets enclose the group. "These items can be in any order."

Third row (children of Active Environment Group): BAG, PEC (*), MCF (* s), MDR (* s), MPG (*), MPO (* † s), MPS (* s), PGD, OBD (text) (*), OBP (text) (*), PTD (*), EAG

* = optional
s = can appear more than once
† = required for every IPO specified in a page

Note: An AFP (MO:DCA-P) presentation page can contain one or more Tag Logical Element (TLE) or Link Logical Element (LLE) structured fields following the AEG. These structures do not affect the presentation of the page. To see which AFP print servers support these structures, see Appendix D, "System Support Information," on page 183.. For the formal definition of these structures, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Figure 33. Structure of Line Format Data

Diagrams
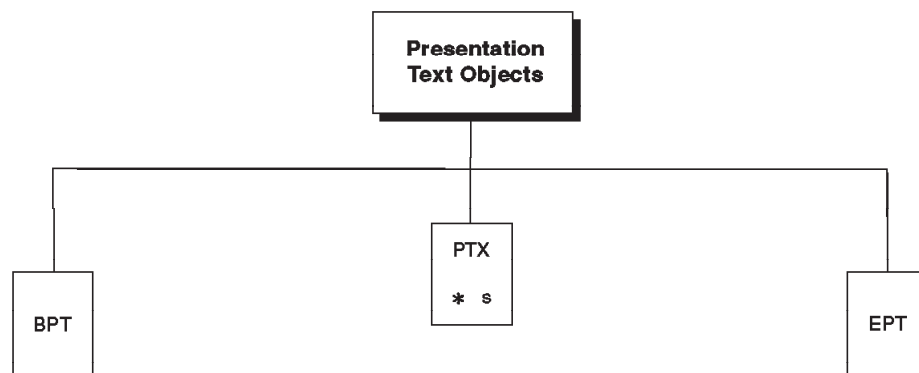
Figure 34. Structure of a Presentation Text Data Object
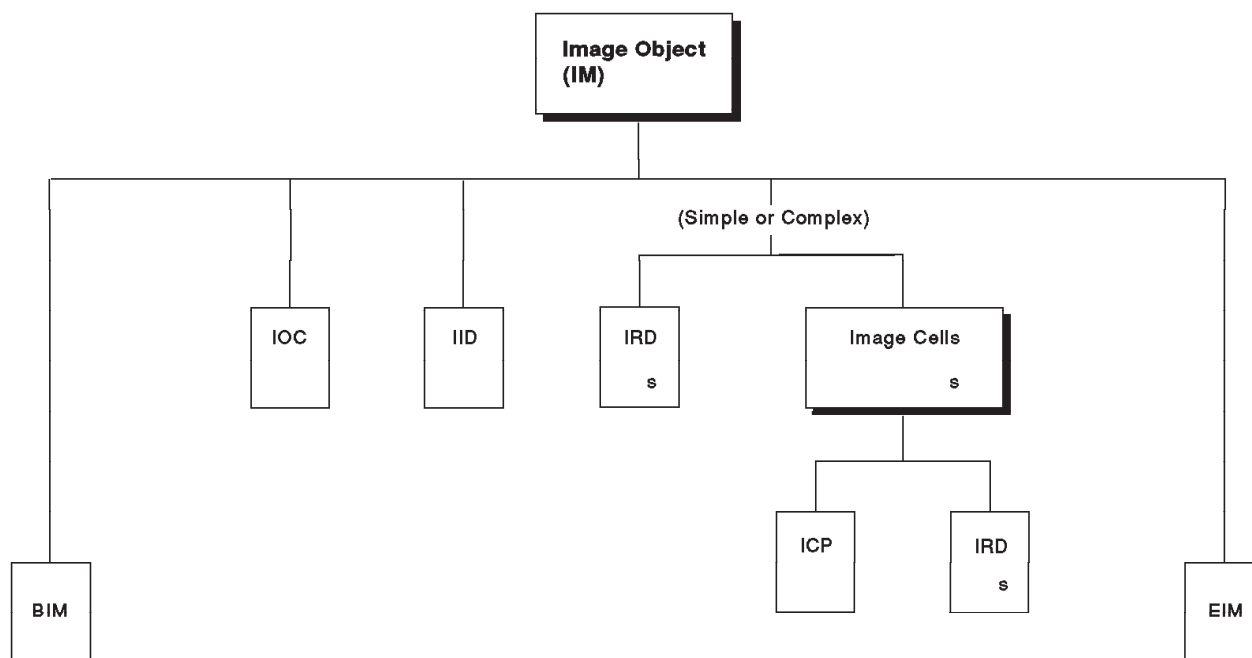
Figure 35. Structure of an IM Image Data Object

* = optional
s = can appear more than once
† = allowed in FS45 only

Figure 36. Structure of an IO Image Data Object

Diagrams


```
                         ┌─────────────────┐
                         │ Graphics Object │
                         └────────┬────────┘
                                  │
        ┌─────────────────────────┼──────────────────────────┐
        │              ┌──────────────────┐         ┌───────┐
        │              │ Object           │         │ GAD   │
        │              │ Environment      │         │ *   s │
        │              │ Group            │         └───────┘
        │              └────────┬─────────┘
        │                       │
        │    ┌──────┬──────┬────┴───┬──────┬──────┬──────┬──────┐
        │  ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐ ┌───┐
        │  │BOG│ │PEC│ │OBD│ │OBP│ │MGO│ │MCF│ │MDR│ │GDD│ │EOG│
        │  │   │ │ * │ │   │ │   │ │ * │ │ * │ │ * s│ │   │ │   │
        │  └───┘ └───┘ └───┘ └───┘ └───┘ └───┘ └───┘ └───┘ └───┘
    ┌───────┐                                                 ┌───────┐
    │ BGR   │                                                 │ EGR   │
    └───────┘                                                 └───────┘
```
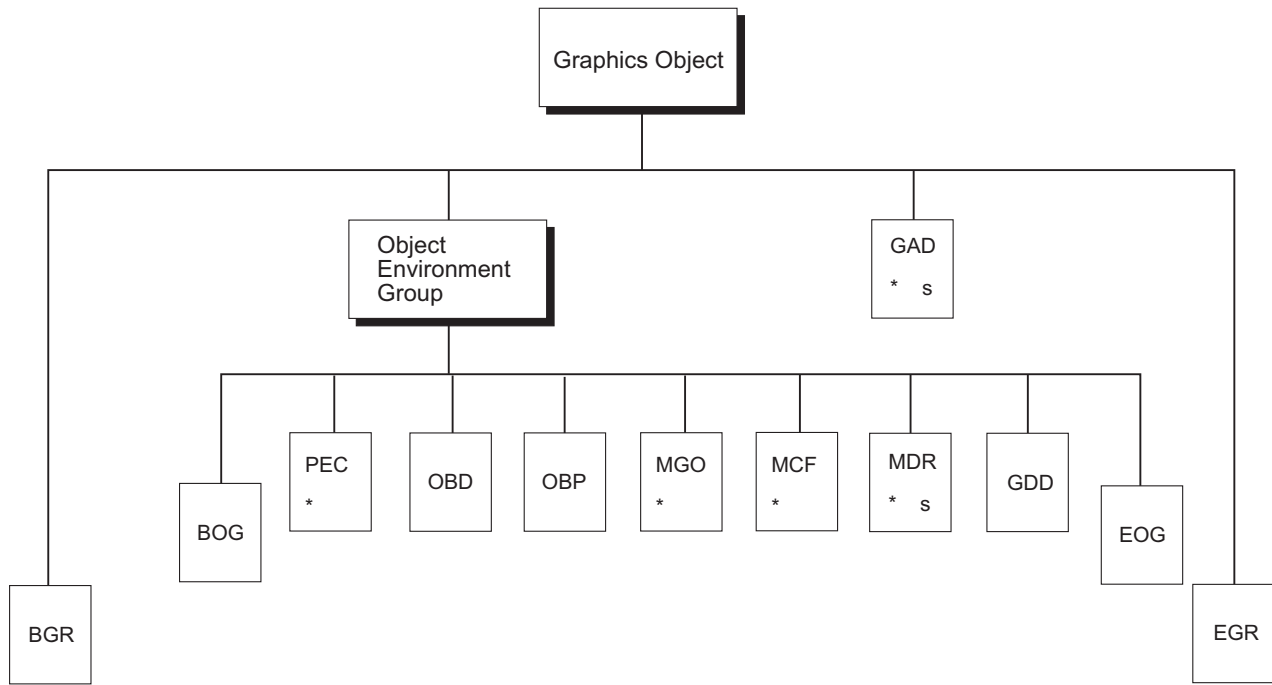
* = optional
s = can appear more than once

Figure 37. Structure of a Graphics Data Object
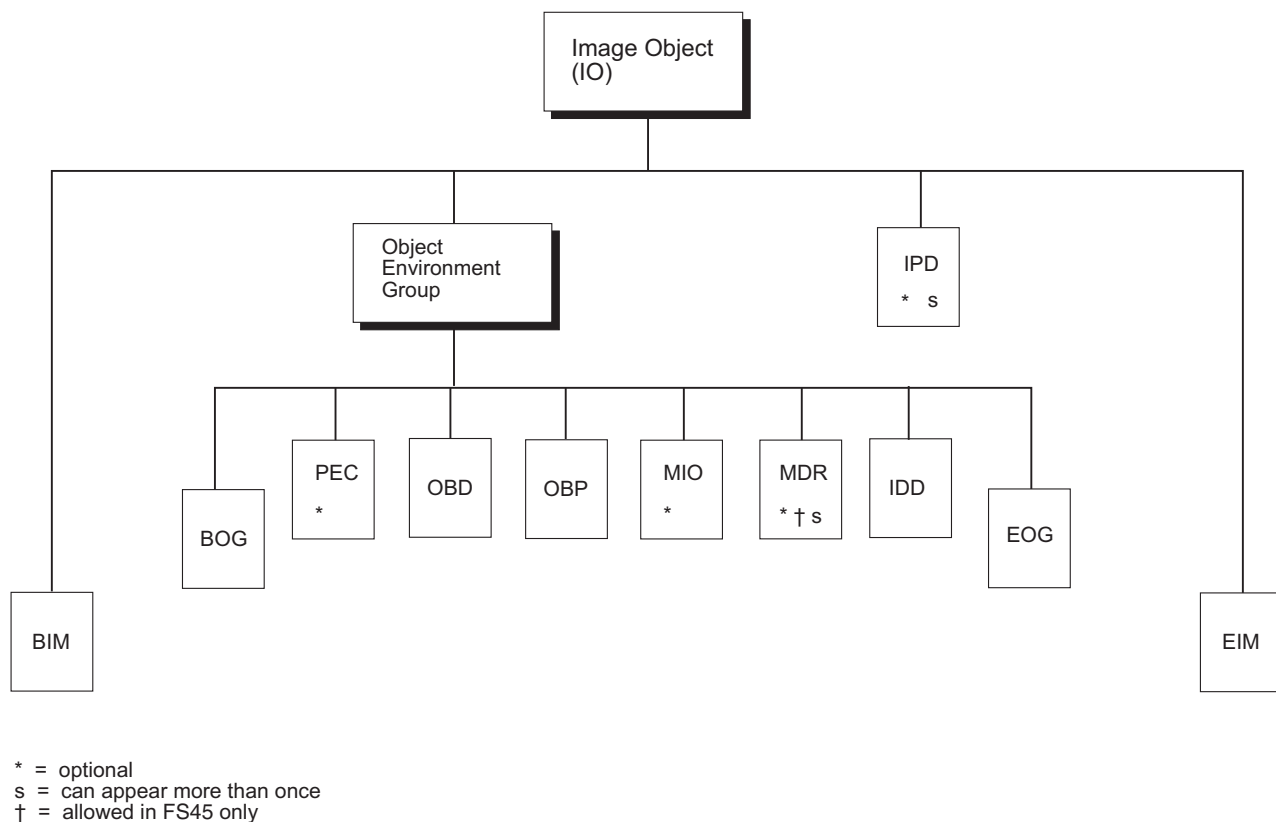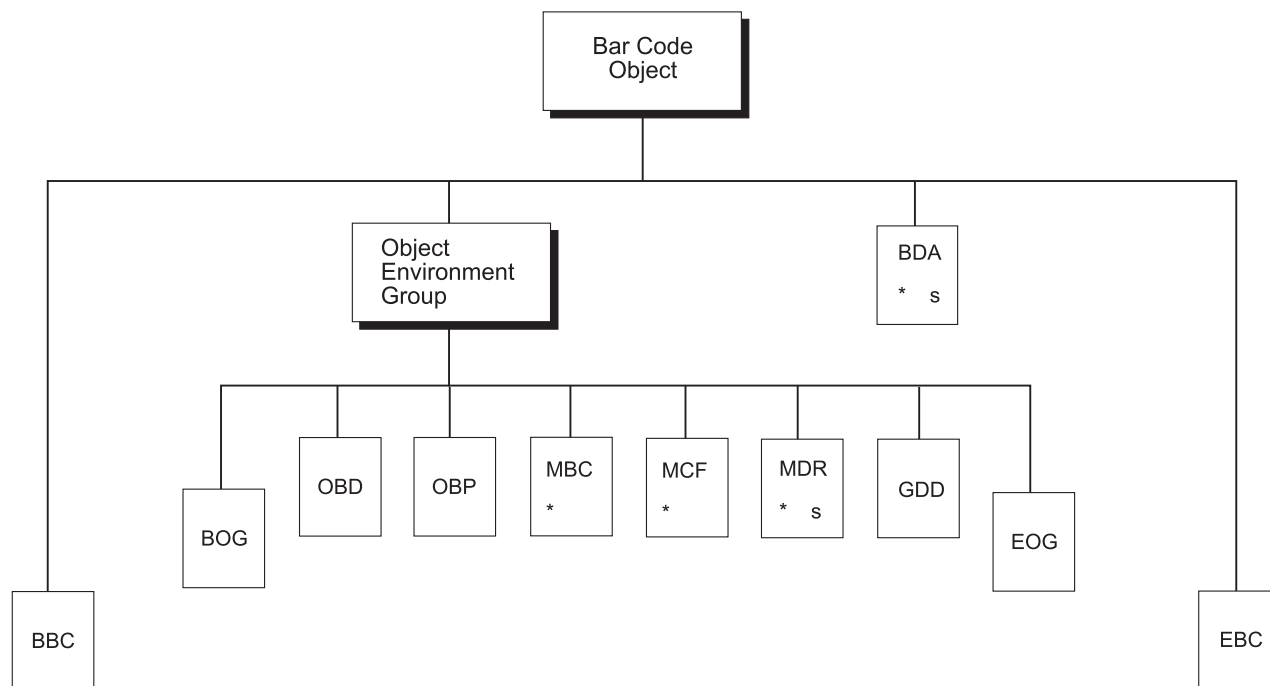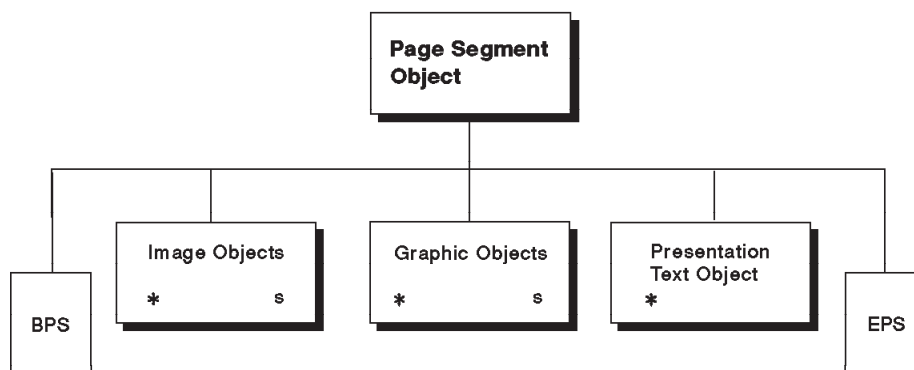
```
*  =  optional
s  =  can appear more than once
```

*Figure 38. Structure of a Bar Code Data Object*


```
* = optional
s = can appear more than once
```

Figure 39. Structure of a Page Segment Resource Object

Note: This is the structure of an AFP page segment. This structure is supported but is replaced strategically with the MO:DCA page segment. For more information, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Diagrams
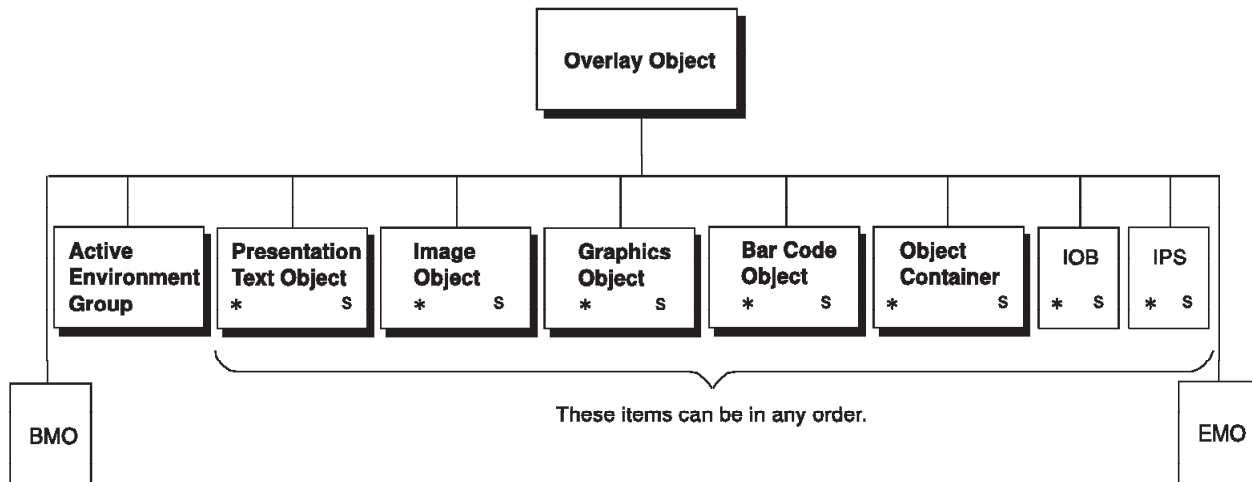
*= optional
S = can appear more than once

Figure 40. Structure of an Overlay Resource Object

Notes:

1. An AFP (MO:DCA-P) overlay object may contain one or more Tag Logical Element (TLE) or Link Logical Element (LLE) structured fields following the AEG. These structures do not affect the presentation of the overlay. To see which AFP print servers support these structures, see Appendix D, "System Support Information," on page 183. For the formal definition of these structures, see the *Mixed Object Document Content Architecture Reference*, SC31-6802.

2. The MPG and MPO structured fields are not supported in the AEG for an overlay.
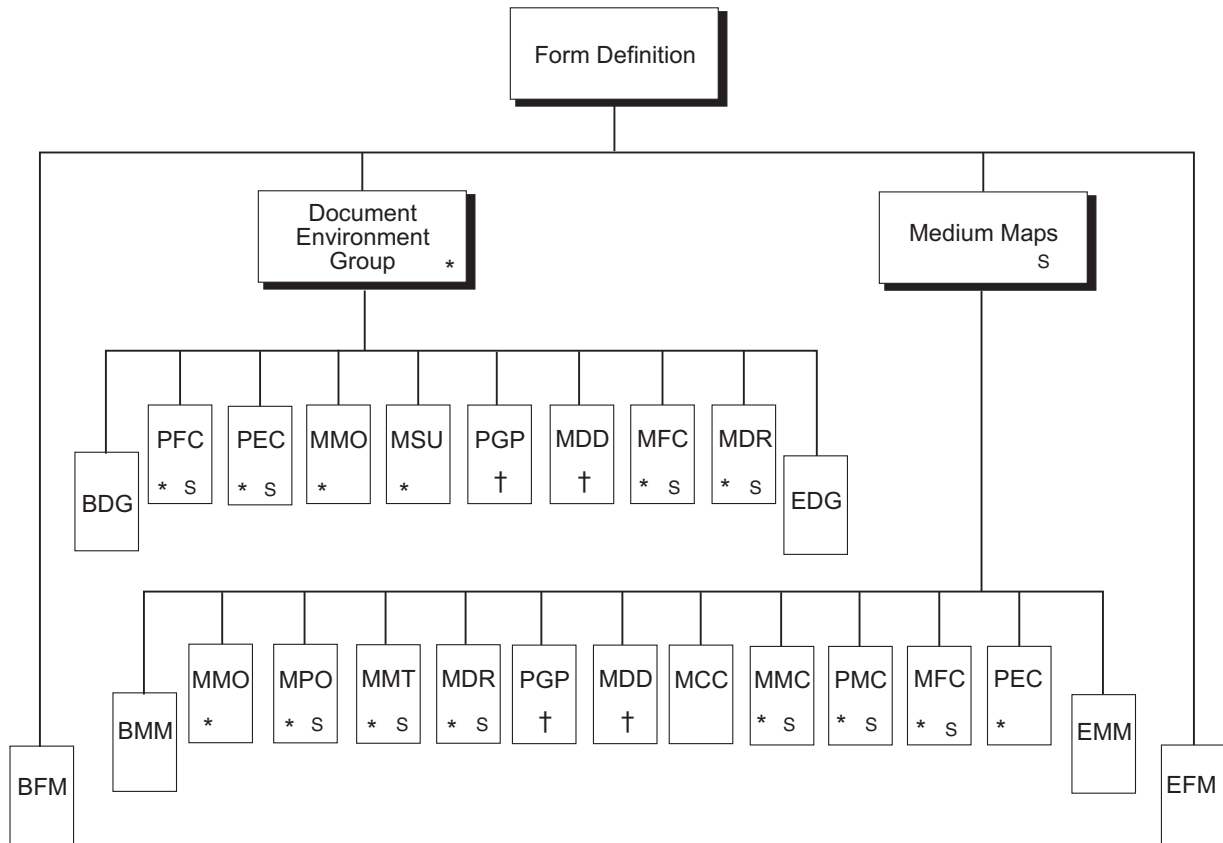
* = optional
s = can appear more than once
† = the structure field is required in either the
Document Environment Group or the Medium Map
Group

Figure 41. Structure of a Form Definition Resource Object

Diagrams

* = optional
s = can appear more than once
+ = the Data Map Transmission Subcase can contain LNDs, RCDs, or XMDs but not a mixture
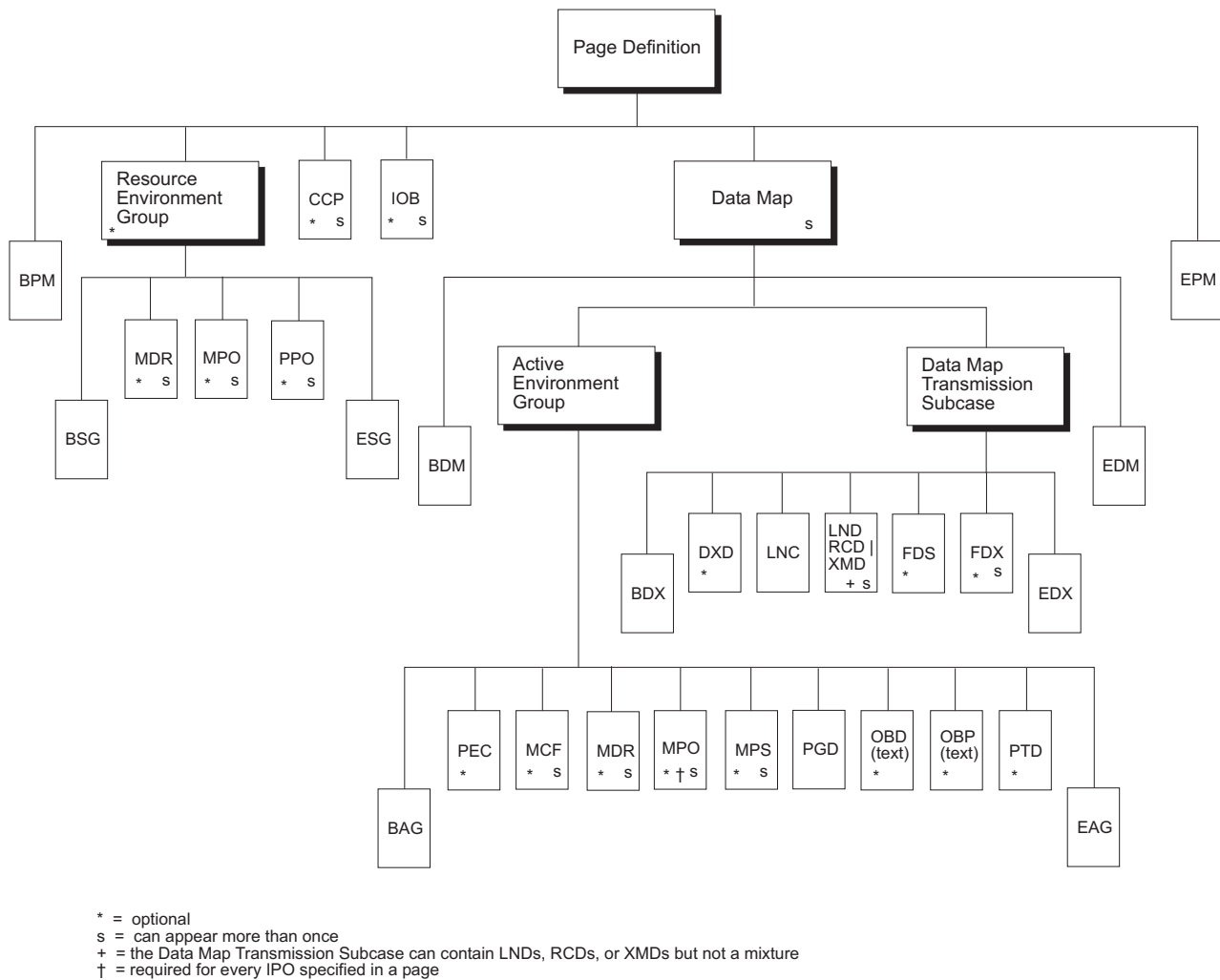† = required for every IPO specified in a page

Figure 42. Structure of a Page Definition Resource Object

Notes:

1. The Data Map Transmission Subcase may contain RCDs or XMDs instead of LNDs.

2. The Data Maps in a Page Definition must all contain LNDs, RCDs, or XMDs. A mixture is not allowed.

3. A Presentation Text Descriptor (PTD) is required in the AEG when a presentation text object is used on a page.

Appendix C. Cross-References

This appendix lists structured fields and PTOCA control sequences in alphabetical order by abbreviation name and in numerical order by hexadecimal code. This list includes not only the structured fields used in line-data and mixed data applications, but also the structured fields used in MO:DCA objects that are supported in AFP environments. For structured fields not described in this publication and for a complete list of MO:DCA structured fields, refer to the *Mixed Object Document Content Architecture Reference*.

Structured Fields Arranged Alphabetically

Table 11. Structured Fields Arranged Alphabetically

BAG	D3A8C9	Begin Active Environment Group
BBC	D3A8EB	Begin Bar Code Object
BCT	D3A89B	Begin Composed Text Object (renamed BPT)
BDA	D3EEEB	Bar Code Data
BDD	D3A6EB	Bar Code Data Descriptor
BDG	D3A8C4	Begin Document Environment Group
BDI	D3A8A7	Begin Document Index
BDM	D3A8CA	Begin Data Map
BDT	D3A8A8	Begin Document
BDX	D3A8E3	Begin Data Map Transmission Subcase
BFG	D3A8C5	Begin Form Environment Group (obsolete)
BFM	D3A8CD	Begin Form Map
BGR	D3A8BB	Begin Graphics Object
BII	D3A87B	Begin Image Object IM
BIM	D3A8FB	Begin Image Object IO
BMM	D3A8CC	Begin Medium Map
BMO	D3A8DF	Begin Overlay
BNG	D3A8AD	Begin Named Page Group
BOC	D3A892	Begin Object Container
BOG	D3A8C7	Begin Object Environment Group
BPG	D3A8AF	Begin Page
BPM	D3A8CB	Begin Page Map
BPS	D3A85F	Begin Page Segment
BPT	D3A89B	Begin Presentation Text Object
BR	D3A8CE	Begin Resource
BRG	D3A8C6	Begin Resource Group
BSG	D3A8D9	Begin Resource Environment Group
CCP	D3A7CA	Conditional Processing Control
CDD	D3A692	Container Data Descriptor

Cross-References

Table 11. Structured Fields Arranged Alphabetically (continued)

CTC	D3A79B	Composed Text Control (obsolete)
CTD	D3A69B	Composed Text Descriptor (renamed PTD Format 1)
CTX	D3EE9B	Composed Text Data (renamed PTX)
DXD	D3A6E3	Data Map Transmission Subcase Descriptor
EAG	D3A9C9	End Active Environment Group
EBC	D3A9EB	End Bar Code Object
ECT	D3A99B	End Composed Text Object (renamed EPT)
EDG	D3A9C4	End Document Environment Group
EDI	D3A9A7	End Document Index
EDM	D3A9CA	End Data Map
EDT	D3A9A8	End Document
EDX	D3A9E3	End Data Map Transmission Subcase
EFG	D3A9C5	End Form Environment Group (obsolete)
EFM	D3A9CD	End Form Map
EGR	D3A9BB	End Graphics Object
EII	D3A97B	End Image Object IM
EIM	D3A9FB	End Image Object IO
EMM	D3A9CC	End Medium Map
EMO	D3A9DF	End Overlay
ENG	D3A9AD	End Named Page Group
EOC	D3A992	End Object Container
EOG	D3A9C7	End Object Environment Group
EPG	D3A9AF	End Page
EPM	D3A9CB	End Page Map
DPS	D3A95F	End Page Segment
EPT	D3A99B	End Presentation Text
ER	D3A9CE	End Resource
ERG	D3A9C6	End Resource Group
ESG	D3A9D9	End Resource Environment Group
FDS	D3AAEC	Fixed Data Size
FDX	D3EEEC	Fixed Data Text
FGD	D3A6C5	Form Environment Group Descriptor (obsolete)
GAD	D3EEBB	Graphics Data
GDD	D3A6BB	Graphics Data Descriptor
ICP	D3AC7B	Image Cell Position
IDD	D3A6FB	Image Data Descriptor IO
IDM	D3ABCA	Invoke Data Map
IEL	D3B2A7	Index Element
IID	D3A67B	Image Input Descriptor IM
IMM	D3ABCC	Invoke Medium Map
IOB	D3AFC3	Include Object

Table 11. Structured Fields Arranged Alphabetically (continued)

IOC	D3A77B	Image Output Control IM
IPD	D3EEFB	Image Picture Data IO
IPG	D3AFAF	Include Page
IPO	D3AFD8	Include Page Overlay
IPS	D3AF5F	Include Page Segment
IRD	D3EE7B	Image Raster Data IM
LIN	------	Line Record

Note: Line Record is not a structured field. However, the abbreviation "LIN" can appear as a variable insert in a number of PSF messages that can also contain structured field abbreviations.

LLE	D3B490	Link Logical Element
LNC	D3AAE7	Line Descriptor Count
LND	D3A6E7	Line Descriptor
MBC	D3ABEB	Map Bar Code
MCC	D3A288	Medium Copy Count
MCD	D3AB92	Map Container Data
MCF	D3B18A	Map Coded Font (Format 1)
MCF	D3AB8A	Map Coded Font (Format 2)
MDD	D3A688	Medium Descriptor
MDR	D3ABC3	Map Data Resource
MFC	D3A088	Medium Finishing Control
MGO	D3ABBB	Map Graphic Object
MIO	D3ABFB	Map IO Image Object
MMC	D3A788	Medium Modification Control
MMO	D3B1DF	Map Medium Overlay
MMT	D3AB88	Map Media Type
MPG	D3ABAF	Map Page
MPO	D3ABD8	Map Page Overlay
MPS	D3B15F	Map Page Segment
MSU	D3ABEA	Map Suppression
NOP	D3EEEE	No Operation
OBD	D3A66B	Object Area Descriptor
OBP	D3AC6B	Object Area Position
OCD	D3EE92	Object Container Data
PEC	D3A7A8	Presentation Environment Control
PFC	D3B288	Presentation Fidelity Control
PGD	D3A6AF	Page Descriptor
PGP	D3ACAF	Page Position (Format 1)
PGP	D3B1AF	Page Position (Format 2)
PMC	D3A7AF	Page Modification Control
PPO	D3ADC3	Preprocess Presentation Object
PTD	D3A69B	Presentation Text Descriptor (Format 1)

Cross-References

Structured Fields Arranged Numerically by Hexadecimal Code

Table 12. Structured Fields Arranged Numerically by Hexadecimal Code

D3A088	MFC	Medium Finishing Control
D3A090	TLE	Tag Logical Element
D3A288	MCC	Medium Copy Count
D3A66B	OBD	Object Area Descriptor
D3A67B	IID	Image Input Descriptor
D3A688	MDD	Medium Descriptor
D3A68D	RCD	Record Descriptor
D3A68E	XMD	XML Descriptor
D3A692	CDD	Container Data Descriptor
D3A69B	CTD	Composed Text Descriptor (renamed PTD Format 1)
D3A69B	PTD	Presentation Text Descriptor (Format 1)
D3A6AF	PGD	Page Descriptor
D3A6BB	GDD	Graphics Data Descriptor
D3A6C5	FGD	Form Environment Group Descriptor (obsolete)
D3A6E3	DXD	Data Map Transmission Subcase Descriptor
D3A6E7	LND	Line Descriptor
D3A6EB	BDD	Bar Code Data Descriptor
D3A6FB	IDD	Image Data Descriptor IO
D3A77B	IOC	Image Output Control
D3A788	MMC	Medium Modification Control
D3A79B	CTC	Composed Text Control (obsolete)
D3A7AF	PMC	Page Modification Control
D3A7A8	PEC	Presentation Environment Control
D3A7CA	CCP	Conditional Processing Control
D3A85F	BPS	Begin Page Segment
D3A87B	BII	Begin Image Block
D3A892	BOC	Begin Object Container
D3A89B	BCT	Begin Composed Text Object (renamed BPT)
D3A89B	BPT	Begin Presentation Text Block
D3A8A7	BDI	Begin Document Index
D3A8A8	BDT	Begin Document
D3A8AD	BNG	Begin Named Page Group

Table 12. Structured Fields Arranged Numerically by Hexadecimal Code (continued)

D3A8AF	BPG	Begin Page
D3A8BB	BGR	Begin Graphics Object
D3A8C4	BDG	Begin Document Environment Group
D3A8C5	BFG	Begin Form Environment Group (obsolete)
D3A8C6	BRG	Begin Resource Group
D3A8C7	BOG	Begin Object Environment Group
D3A8C9	BAG	Begin Active Environment Group
D3A8CA	BDM	Begin Data Map
D3A8CB	BPM	Begin Page Map
D3A8CC	BMM	Begin Medium Map
D3A8CD	BFM	Begin Form Map
D3A8CE	BR	Begin Resource
D3A8D9	BSG	Begin Resource Environment Group
D3A8DF	BMO	Begin Overlay
D3A8E3	BDX	Begin Data Map Transmission Subcase
D3A8EB	BBC	Begin Bar Code Object
D3A8FB	BIM	Begin Image Object IO
D3A95F	EPS	End Page Segment
D3A97B	EII	End Image Block IM
D3A992	EOC	End Object Container
D3A99B	ECT	End Composed Text Object (renamed EPT)
D3A99B	EPT	End Presentation Text Object
D3A9A7	EDI	End Document Index
D3A9A8	EDT	End Document
D3A9AD	ENG	End Named Page Group
D3A9AF	EPG	End Page
D3A9BB	EGR	End Graphics Object
D3A9C4	EDG	End Document Environment Group
D3A9C5	EFG	End Form Environment Group (obsolete)
D3A9C6	ERG	End Resource Group
D3A9C7	EOG	End Object Environment Group
D3A9C9	EAG	End Active Environment Group
D3A9CA	EDM	End Data Map
D3A9CB	EPM	End Page Map
D3A9CC	EMM	End Medium Map
D3A9CD	EFM	End Form Map
D3A9CE	ER	End Resource
D3A9D9	ESG	End Resource Environment Group
D3A9DF	EMO	End Overlay
D3A9E3	EDX	End Data Map Transmission Subcase
D3A9EB	EBC	End Bar Code Object

Cross-References

Table 12. Structured Fields Arranged Numerically by Hexadecimal Code (continued)

D3A9FB	EIM	End Image Object IO
D3AAE7	LNC	Line Descriptor Count
D3AAEC	FDS	Fixed Data Size
D3AB88	MCF	Map Media Type
D3AB92	MCD	Map Container Data
D3ABAF	MPG	Map Page
D3ABBB	MGO	Map Graphic Object
D3ABC3	MDR	Map Data Resource
D3ABCA	IDM	Invoke Data Map
D3ABCC	IMM	Invoke Medium Map
D3ABD8	MPO	Map Page Overlay
D3ABEA	MSU	Map Medium Suppression
D3ABEB	MBC	Map Bar Code
D3ABFB	MIO	Map IO Image Object
D3AC6B	OBP	Object Area Position
D3AC7B	ICP	Image Cell Position
D3ACAF	PGP	Page Position (Format 1)
D3ADC3	PPO	Preprocess Presentation Object
D3AF5F	IPS	Include Page Segment
D3AFAF	IPG	Include Page
D3AFC3	IOB	Include Object
D3AFD8	IPO	Include Page Overlay
D3B15F	MPS	Map Page Segment
D3B18A	MCF	Map Coded Font (Format 1)
D3B19B	PTD	Presentation Text Descriptor (Format 2)
D3B1AF	PGP	Page Position (Format 2)
D3B1DF	MMO	Map Medium Overlay
D3B288	PFC	Presentation Fidelity Control
D3B2A7	IEL	Index Element
D3B490	LLE	Link Logical Element
D3EE7B	IRD	Image Raster Data
D3EE92	OCD	Object Container Data
D3EE9B	CTX	Composed Text Data (renamed PTX)
D3EEBB	GAD	Graphics Data
D3EE9B	PTX	Presentation Text Data
D3EEEB	BDA	Bar Code Data
D3EEEC	FDX	Fixed Data Text
D3EEEE	NOP	No Operation
D3EEFB	IPD	Image Picture Data IO

PTOCA Control Sequences Arranged Alphabetically

This is a list in alphabetical order by abbreviation name of text control sequences that can appear in the presentation text (PTX) structured field. An even function-type code indicates that the control sequence is followed by code points or by an unchained control sequence. An odd function-type code indicates that the control sequence is followed by a chained control sequence. Chaining control sequences reduces the number of bytes in the PTX structured field by eliminating the two-byte prefix and class code from each chained control sequence. See *Presentation Text Object Content Architecture Reference* for details of these control sequences.

Table 13. PTOCA Control Sequences Arranged Alphabetically

AMB	D2(D3)	Absolute Move Baseline
AMI	C6(C7)	Absolute Move Inline
BLN	D8(D9)	Begin Line
BSU	F2(F3)	Begin Suppression
DBR	E6(E7)	Draw Baseline Rule
DIR	E4(E5)	Draw Inline Rule
ESU	F4(F5)	End Suppression
NOP	F8(F9)	No Operation
OVS	72(73)	Overstrike
RMB	D4(D5)	Relative Move Baseline
RMI	C8(C9)	Relative Move Inline
RPS	EE(EF)	Repeat String
SBI	D0(D1)	Set Baseline Increment
SCFL	F0(F1)	Set Coded Font Local
SEC	80(81)	Set Extended Text Color
SIA	C2(C3)	Set Intercharacter Adjustment
SIM	C0(C1)	Set Inline Margin
STC	74(75)	Set Text Color
STO	F6(F7)	Set Text Orientation
SVI	C4(C5)	Set Variable Space Character Increment
TBM	78(79)	Temporary Baseline Move
TRN	DA(DB)	Transparent Data
USC	76(77)	Underscore

PTOCA Control Sequences Arranged Numerically

These are the PTOCA control sequences listed in numerical order by hexadecimal code.

72(73)	OVS	Overstrike
74(75)	STC	Set Text Color
76(77)	USC	Underscore
78(79)	TBM	Temporary Baseline Move

Cross-References

80(81)	SEC	Set Extended Text Color
C0(C1)	SIM	Set Inline Margin
C2(C3)	SIA	Set Intercharacter Adjustment
C4(C5)	SVI	Set Variable Space Character Increment
C6(C7)	AMI	Absolute Move Inline
C8(C9)	RMI	Relative Move Inline
D0(D1)	SBI	Set Baseline Increment
D2(D3)	AMB	Absolute Move Baseline
D4(D5)	RMB	Relative Move Baseline
D8(D9)	BLN	Begin Line
DA(DB)	TRN	Transparent Data
E4(E5)	DIR	Draw Inline Rule
E6(E7)	DBR	Draw Baseline Rule
EE(EF)	RPS	Repeat String
F0(F1)	SCFL	Set Coded Font Local
F2(F3)	BSI	Begin Suppression
F4(F5)	ESU	End Suppression
F6(F7)	STO	Set Text Orientation
F8(F9)	NOP	No Operation

Appendix D. System Support Information

Table 14 on page 184, Table 15 on page 188, Table 16 on page 193, Table 17 on page 199, and Table 18 on page 200 list the data stream structured fields and objects supported by current releases of AFP presentation servers. Table 19 on page 201 lists data stream functions, such as interchange set definitions, that are supported by current releases of AFP presentation servers. Table 20 on page 202 lists non-OCA object formats that are supported by current releases of AFP presentation servers.

An X indicates that an IBM component that implements the particular data stream element either has already been announced or is available for that system.

A dash (–) indicates that the data stream element is not supported by that system.

The following symbols apply to the syntax structures shown in this table:

[ ]	Brackets indicate optional structures.
...	Ellipsis marks indicate structures that may be repeated.
!	Exclamation marks indicate structures that may occur in any order relative to the other structures so marked within the same syntax structure. For example, in the Presentation Page object, Text, Image, and Graphics objects can appear in any order relative to each other.
Italics	Italics indicate that the syntax element itself is a complete syntax structure that is referenced in this section of the document.

Table 14. Document Structured Fields

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
Document										
Begin Document	X	X	X	X	X	X	X	X	X	X
[document index] (See Note 1)	X	X	X	X	X	X	X	X	X	—
[resource environment group]|...	X	X	—	—	—	—	—	X	X	—
[medium map]|... (See Note 2)	X	X	—	—	X	X	X	X	X	X
[Invoke Medium Map]|...	X	X	X	X	X	X	X	X	X	X
[Link Logical Element]|... (See Note 3)	X	X	X^4	X^4	X	X	X	X	X	X
[Include Page]|...	—	—	—	—	—	—	—	—	—	—
[presentation page]|...	X	X	X	X	X	X	X	X	X	X
[page group]|... (See Note 3)	X	X	X	X	X	X	X	X	X	X
End Document	X	X	X	X	X	X	X	X	X	X
Resource Environment Group										
Begin Resource Environment Group	X	X	—	—	—	—	—	X	X	X
[Map Data Resource]...	X	X	—	—	—	—	—	X	X	X
[Map Page Overlay]...	X	X	—	—	—	—	—	X	X	X^{10}
[Preprocess Presentation Object]...	X	X	—	—	—	—	—	X	X	X^{10}
End Resource Environment Group	X	X	—	—	—	—	—	X	X	X
Presentation Page										
Begin Page	X	X	X	X	X	X	X	X	X	X
Active Environment Group	X	X	X	X	X	X	X	X	X	X
[presentation text object]|...	X	X	X	X	X	X	X	X	X	X
[image object (IM)]|...	X	X	X	X	X	X	X	X	X	X
[image object (IO)]|...	X	X	X	X	X	X	X	X	X	X
[graphics object]|...	X	X	X	X	X	X	X	X	X	X
[bar code object]|...	X	X	X	X	X	X	X	X	X	—

Table 14. Document Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[object container]...	X	X	—	—	—	X	X	X	X	X
[Include Object]...	X	X	—	X⁴	X	X	X	X	X	X
[Include Page]...	—	—	—	—	—	—	—	X	X	—
[Include Page Segment]...	X	X	X	X	X	X	X	X	X	X
[LD Position Migration triplet]	X	X	X	X	X	X	X	X	X	X
[Include Page Segment]...	X	X	X	X	X	X	X	X	X	X
[Link Logical Element]... (See Note 2)	X	X	X⁴	X⁴	X	X	X	X	X	X
[Tag Logical Element]... (See Note 2)	X	X	X⁴	X⁴	X	X	X	X	X	X
End Page	X	X	X	X	X	X	X	X	X	X
Active Environment Group										
Begin Active Environment Group	X	X	X	X	X	X	X	X	X	X
[Map Coded Font, fmt 1]... (See Note 5)	X	X	X	X	X	X	X	X	X	X
[Map Coded Font, fmt 2]...	X	X	X	X	X	X	X	X	X	X
[Coded Font Name Reference]!	X	X	X	X	X	X	X	X	X	X
[Font Descriptor triplet]!	X	X	X⁴	X⁴	X	X	X	X	X	X
[Horizontal Scale Factor triplet]!	X	X	X⁴	X⁴	X	X	X	X	X	X
[Font Res & Met Tech triplet]!	X	X	—	—	—	X	X	X	X	X
[Map Data Resource]...	X	X	—	—	—	X	X	X	X	X
[Map Page] (See Note 6)	—	—	—	—	—	—	—	X	X	—
[Map Page Overlay]... (See Note 7)	X	X	X	X	X	X	X	X	X	X
[Map Page Segment]... (See Note 5)	X	X	X	X	X	X	X	X	X	X
[Page Descriptor]	X	X	X	X	X	X	X	X	X	X
[Color Specification triplet]!	X	X	—	X⁴	X	X	X	X	X	X

Table 14. Document Structured Fields *(continued)*

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[Reset Mixing triplet]!	X	X	—	X[4]	X	X	X	X	X	X
[Object Area Descriptor (text only)] (See Note 8)	X	X	X	X	X	X	X	X	X	X
[Object Area Position (text only)] (See Note 8)	X	X	X	X	X	X	X	X	X	X
[Presentation Text Descriptor, fmt 1] (See Note 9)	X	X	X	X	X	X	X	X	X	X
[Presentation Text Descriptor, fmt 2] (See Note 9)	X	X	X	X	X	X	X	X	X	X
End Active Environment Group	X	X	X	X	X	X	X	X	X	X
Page Group										
Begin Named Page Group (See Note 3)	X	X	X	X	X	X	X	X	X	X
[Tag Logical Element]... (See Note 3)	X	X	X[4]	X[4]	X	X	X	X	X	X
[Link Logical Element]... (See Note 3)	X	X	X[4]	X[4]	X	X	X	X	X	X
[medium map]!... (See Note 2)	X	X	—	—	X	X	X	X	X	X
[Invoke Medium Map]!...	X	X	X	X	X	X	X	X	X	X
[Include Page]!...	—	—	—	—	—	—	—	—	—	—
presentation page!....	X	X	X	X	X	X	X	X	X	X
[*page group*]!...	X	X	X	X	X	X	X	X	X	X
End Named Page Group (See Note 3)	X	X	X	X	X	X	X	X	X	X

Table 14. Document Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer

Notes:

1. A document index following BDT is . AFP products process a document index that is external to the document.

2. A Medium Map inside the document (internal Medium Map) must be followed by an IMM that invokes the Medium Map.

3. These structures are processed as NOPs for printing. They are processed for viewing.

4. Support of these structures is provided by PTFs to the release.

5. This structured field cannot be repeated with PSF/VSE 2.1.

6. Every page referenced with an IPG within a page must also be named in an MPG in the AEG of that page. The MPG is by Infoprint Manager, which downloads pages to be saved based on their presence in the inline resource group.

7. Every overlay named in an IPO specified in a page must also be named in an MPO in the AEG of that page.

8. These structured fields are ignored by AFP print servers and the AFP Viewer.

9. This structured field is required if presentation text objects are present in the page; the AEG must have either fmt 1 or fmt 2, but cannot have both.

10. These structured fields are ignored by the AFP Viewer in a Resource Environment Group.

Table 15. Data Object Structured Fields

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
Presentation Text Object										
Begin Presentation Text Object	X	X	X	X	X	X	X	X	X	X
[Presentation Text data]...	X	X	X	X	X	X	X	X	X	X
PTOCA PT1 subset	X	X	X	X	X	X	X	X	X	X
PTOCA PT2 subset	X	X	X	X	X	X	X	X	X	X
PTOCA PT3 subset	X	X	—	X[1]	X	X	X	X	X	X
End Presentation Text Object	X	X	X	X	X	X	X	X	X	X
Image Object (IM)										
Begin Image Object (IM)	X	X	X	X	X	X	X	X	X	X
[LD Position Migration triplet]	X	X	X	X	X	X	X	X	X	—
Image Output Control	X	X	X	X	X	X	X	X	X	X
Image Input Descriptor	X	X	X	X	X	X	X	X	X	X
[IM image color]	X	X	—	X[1]	X	X	X	X	X	X
image data...										
Image Raster Data... OR	X	X	X	X	X	X	X	X	X	X
image cell...										
Image Cell Position	X	X	X	X	X	X	X	X	X	X
Image Raster Data ...	X	X	X	X	X	X	X	X	X	X
End Image Object (IM)	X	X	X	X	X	X	X	X	X	X
Image Object (IOCA)										
Begin Image Object (IO)	X	X	X	X	X	X	X	X	X	X
[LD Position Migration triplet]	X	X	X	X	X	X	X	X	X	—
Begin Object Environment Group	X	X	X	X	X	X	X	X	X	X
Object Area Descriptor	X	X	X	X	X	X	X	X	X	X
[Color Specification triplet]	X	X	—	X[1]	X	X	X	X	X	X
[Reset Mixing triplet]	X	X	—	X[1]	X	X	X	X	X	X

Table 15. Data Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
Object Area Position	X	X	X	X	X	X	X	X	X	X
[Map Image Object]	X	X	X	X	X	X	X	X	X	X
[Position and Trim]	X	X	X	X	X	X	X	X	X	X
[Center and Trim]	X	X	X	X	X	X	X	X	X	X
[Replicate and Trim]	X	X	—	X^1	X	X	X	X	X	X
[Scale to Fit]	X	X	X	X	X	X	X	X	X	X
[Scale to Fill]	X	X	—	—	X	X	X	X	X	X
[Image point-to-pel]	X	X	X	X	X	X	X	X	X	X
[Image point-to-pel double dot]	X	X	X	X	X	X	X	X	X	X
[Map Data Resource]...	X^4	X^4	—	—	—	X^4	X^4	X^4	X^4	—
Image Data Descriptor	X	X	X	X	X	X	X	X	X	X
[Set Bilevel Image Color SDF]	X	X	—	X^1	X	X	X	X	X	X
End Object Environment Group	X	X	X	X	X	X	X	X	X	X
[Image Picture Data]...	X	X	X	X	X	X	X	X	X	X
IOCA FS10 subset	X	X	X	X	X	X	X	X	X	X
IOCA FS11 subset	—	—	—	—	—	—	—	X	X	X
End Image Object (IO)	X	X	X	X	X	X	X	X	X	X
Graphics Object (GOCA)										
Begin Graphics Object	X	X	X	X	X	X	X	X	X	X
[LD Position Migration triplet]	X	X	X	X	X	X	X	X	X	—
Begin Object Environment Group	X	X	X	X	X	X	X	X	X	X
Object Area Descriptor	X	X	X	X	X	X	X	X	X	X
[Color Specification triplet]	X	X	—	X^1	X	X	X	X	X	X
[Reset Mixing triplet]	X	X	—	X^1	X	X	X	X	X	X
Object Area Position	X	X	X	X	X	X	X	X	X	X
[Map Graphics Object]	X	X	X	X	X	X	X	X	X	X

Table 15. Data Object Structured Fields *(continued)*

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[Position and Trim]	X	X	X	X	X	X	X	X	X	X
[Center and Trim]	X	X	X	X	X	X	X	X	X	X
[Replicate and Trim]	X	X	—	—	X	X	X	X	X	—
[Scale to Fit]	X	X	X	X	X	X	X	X	X	X
[Scale to Fill]	X	X	—	—	X	X	X	X	X	—
[Map Coded Font, fmt 2]... (See Notes 2 and 3)	X	X	X	X	X	X	X	X	X	X
[Coded Font Name Reference]	X	X	X	X	X	X	X	X	X	X
[Font Descriptor triplet]	X	X	X[1]	X[1]	X	X	X	X	X	X
[Horizontal Scale Factor triplet]	X	X	X[1]	X[1]	X	X	X	X	X	—
[Font Res & Met Tech triplet]	X	X	—	—	X	X	X	X	X	X
[Map Data Resource]...	X[5]	X[5]	—	—	—	X[5]	X[5]	X[5]	X[5]	—
Graphics Data Descriptor	X	X	X	X	X	X	X	X	X	X
End Object Environment Group	X	X	X	X	X	X	X	X	X	X
[Graphics Data]...	X	X	X	X	X	X	X	X	X	X
GOCA DR/2V0 subset	X	X	X	X	X	X	X	X	X	X
End Graphics Object	X	X	X	X	X	X	X	X	X	X
Bar Code Object (BCOCA)										
Begin Bar Code Object	X	X	X	X	X	X	X	X	X	—
[LD Position Migration triplet]	X	X	X	X	X	X	X	X	X	—
Begin Object Environment Group	X	X	X	X	X	X	X	X	X	—
Object Area Descriptor	X	X	X	X	X	X	X	X	X	—
[Color Specification triplet]	X	X	—	X[1]	X	X	X	X	X	—
[Reset Mixing triplet]	X	X	—	X[1]	X	X	X	X	X	—
Object Area Position	X	X	X	X	X	X	X	X	X	—
[Map Bar Code Object]	X	X	X	X	X	X	X	X	X	—
[Position]	X	X	X	X	X	X	X	X	X	—

Table 15. Data Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[Map Coded Font, fmt 2]... (See Note 2)	X	X	X	X	X	X	X	X	X	—
[Coded Font Name Reference]!	X	X	X	X	X	X	X	X	X	—
[Font Descriptor triplet]!	X	X	X^1	X^1	X	X	X	X	X	—
[Horizontal Scale Factor triplet]!	X	X	X^1	X^1	X	X	X	X	X	—
[Font Res & Met Tech triplet]!	X	X	—	—	X	X	X	X	X	—
[Map Data Resource]...	X^5	X^5	—	—	—	X^5	X^5	X^5	X^5	—
Bar Code Data Descriptor	X	X	X	X	X	X	X	X	X	—
End Object Environment Group	X	X	X	X	X	X	X	X	X	—
[Bar Code Data]...	X	X	X	X	X	X	X	X	X	—
BCOCA BCD1 subset	X	X	X	X	X	X	X	X	X	—
End Bar Code Object	X	X	X	X	X	X	X	X	X	—
Object Container										
Begin Object Container	X	X	—	—	—	X	X	X	X	X
[Begin Object Environment Group]	X	X	—	—	—	X	X	X	X	X
[Object Area Descriptor]	X	X	—	—	—	X	X	X	X	X
[Color Specification triplet]!	X	X	—	—	—	X	X	X	X	X
[Reset Mixing triplet]!	X	X	—	—	—	X	X	X	X	X
[Object Area Position]	X	X	—	—	—	X	X	X	X	X
[Map Container Data]	X	X	—	—	—	X	X	X	X	X
[Position and Trim]	X	X	—	—	—	X	X	X	X	X
[Center and Trim]	X	X	—	—	—	X	X	X	X	X
[Scale to Fit]	X	X	—	—	—	X	X	X	X	X
[Scale to Fill]	X	X	—	—	—	X	X	X	X	X
[Map Data Resource]...	X	X	—	—	—	X	X	X	X	X
[Container Data Descriptor]	X	X	—	—	—	X	X	X	X	X

Table 15. Data Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[End Object Environment Group]	X	X	—	—	—	X	X	X	X	X
[Object Container Data]	X	X	—	—	—	X	X	X	X	X
[End Object Container]	X	X	—	—	—	X	X	X	X	X

Notes:

1. Support of these structures is provided by PTFs to the release.

2. For PSF resource management, an MCF mapping the same font must be specified in the AEG whenever an MCF is specified in a graphics or bar code OEG. Local IDs need not match, ID X'FE' may be used on the MCF in the AEG.

3. MCFs are not allowed in graphics objects or bar code objects that are included in page segment objects.

4. TrueType/OpenType fonts are not supported on the MDR in this object.

5. Only TrueType/OpenType font MDR supported.

Table 16. Resource Object Structured Fields

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
Resource Group										
Begin Resource Group	X	X	X	X	X	X	X	X	X	X
resource object...	X	X	X	X	X	X	X	X	X	X
End Resource Group	X	X	X	X	X	X	X	X	X	X
Resource Object										
Begin Resource	X	X	X	X	X	X	X	X	X	X
page segment object	X	X	X	—	X	X	X	X	X	X
overlay object	X	X	X	—	X	X	X	X	X	X
Bar Code (BCOCA) object OR	X	X	—	—	X	X	X	X	X	—
Graphics (GOCA) object OR	X	X	—	—	X	X	X	X	X	X
Image (IOCA) object OR	X	X	—	—	X	X	X	X	X	X
Object Container OR	X	X	—	—	—	X	X	X	X	X
Form Definition object OR	X	X	X	X	X	X	X	X	X	X
Page Definition OR	X	X	X	X	X	X	X	X	X	—
font resource object (See Note 1)	X	X	X	—	X	X	X	X	X	X[2]
Document object OR	—	—	—	—	—	—	—	X	—	—
End Resource	X	X	X	X	X	X	X	X	X	X
Form Definition Object										
Begin Form Map	X	X	X	X	X	X	X	X	X	X
[*Document Environment Group*]	X	X	X	X	X	X	X	X	X	X
medium map...	X	X	X	X	X	X	X	X	X	X
End Form Map	X	X	X	X	X	X	X	X	X	X
Document Environment Group										
Begin Document Environment Group	X	X	X	X	X	X	X	X	X	X
[Presentation Fidelity Control]	X	X	—	—	—	—	—	X	X	—
[Color Fidelity triplet]	X	X	—	—	—	—	—	X	X	—

Table 16. Resource Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[Font Fidelity triplet]	X	X	—	—	—	—	—	X	X	—
[Map Medium Overlay]	X	X	X	X	X	X	X	X	X	X
[Map Suppression]	X	X	X	X	X	X	X	X	X	X
[Page Position, fmt 1] (See Note 6)	X	X	X	X	X	X	X	X	X	X
[Page Position, fmt 2] (See Note 6)	X	X	X	X	X	X	X	X	X	X
[Simple N-up]	X	X	X^4	X^4	X	X	X	X	X	X
[Enhanced N-up]	X	X	X^4	X^4	X	X	X	X	X	X
[Medium Descriptor] (See Note 6)	X	X	X	X	X	X	X	X	X	X
[Medium Finishing Control]	X	X	—	—	X	X	X	X	X	—
End Document Environment Group	X	X	X	X	X	X	X	X	X	X
Medium Map										
Begin Medium Map (See Note 6)	X	X	X	X	X	X	X	X	X	X
[Map Medium Overlay]	X	X	X	X	X	X	X	X	X	X
[Map Page Overlay]...	X	X	X	X	X	X	X	X	X	—
[Map Media Type]...	X	X	—	—	X	X	X	X^{11}	—	—
[Page Position, fmt 1] (See Note 6)	X	X	X	X	X	X	X	X	X	X
[Page Position, fmt 2] (See Note 6)	X	X	X	X	X	X	X	X	X	X
[Simple N-up]	X	X	X^4	X^4	X	X	X	X	X	X
[Enhanced N-up]	X	X	X^4	X^4	X	X	X	X	X	X
[Medium Descriptor] (See Note 6)	X	X	X	X	X	X	X	X	X	X
Medium Copy Count	X	X	X	X	X	X	X	X	X	X
[Medium Modification Control]...	X	X	X	X	X	X	X	X	X	X
[Fixed medium information]	X	X	—	X^4	—	—	—	X	X	—
[Media source, fmt 1]	X	X	X	X	X	X	X	X	X	—
[Media source, fmt 2]	X	X	X^4	X^4	X	X	X	X	X	—
[Media destination]	X	X	X^4	—	X	X	X	X	X	—

Table 16. Resource Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[N-up Control]	X	X	X^4	X^4	X	X	X	X	X	X
[Set-up Verification IDs]	X	X	—	X^4	X	X	X	X	X	—
[Page Modification Control]...	X	X	X^4	X^4	X	X	X	X	X	X
[Medium Finishing Control]	X	X	—	—	—	X	X	X	X	—
End Medium Map	X	X	X	X	X	X	X	X	X	X
AFP Page Segment Object										
Begin Page Segment	X	X	X	X	X	X	X	X	X	X
[image object-IM]!...	X	X	X	X	X	X	X	X	X	X
[image object-IO]!...	X	X	X	X	X	X	X	X	X	X
[graphics object]!... (See Note 3)	X	X	X	X	X	X	X	X	X	X
[presentation text object]	X	X	X	X	X	X	X	X	X	X
End Page Segment	X	X	X	X	X	X	X	X	X	X
MO:DCA Page Segment Object										
Begin Page Segment	X	X	X	X	X	X	X	X	X	X
[image object-IO]!...	X	X	X	X	X	X	X	X	X	X
[graphics object]!... (See Note 3)	X	X	X	X	X	X	X	X	X	X
[bar code object]!... (See Note 3)	X	X	X	X	X	X	X	X	X	—
End Page Segment	X	X	X	X	X	X	X	X	X	X
Overlay Object										
Begin Overlay	X	X	X	X	X	X	X	X	X	X
Active Environment Group	X	X	X	X	X	X	X	X	X	X
[presentation text object]!...	X	X	X	X	X	X	X	X	X	X
[image object-IM]!...	X	X	X	X	X	X	X	X	X	X
[image object-IO]!...	X	X	X	X	X	X	X	X	X	X
[graphics object]!...	X	X	X	X	X	X	X	X	X	X
[bar code object]!...	X	X	X	X	X	X	X	X	X	—

Table 16. Resource Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[object container]!...	X	X	—	—	—	X	X	X	X	X
[Include Object]!...	X	X	—	X⁴	X	X	X	X	X	X
[Include Page Segment]!...	X	X	X	X	X	X	X	X	X	X
[LD Position Migration triplet]	X	X	X	X	X	X	X	X	X	X
[Link Logical Element]!... (See Note 5)	X	X	X⁴	X⁴	X	X	X	X	X	X
[Tag Logical Element]!... (See Note 5)	X	X	X⁴	X⁴	X	X	X	X	X	X
End Overlay	X	X	X	X	X	X	X	X	X	X
Page Definition Object										
Begin Page Map	X	X	X	X	X	X	X	X	X	—
[resource environment group]	X	X	—	—	—	—	—	X	X	—
[Conditional Processing Control]!...	X	X	X	X	X	X	X	X	X	—
data map...	X	X	X	X	X	X	X	X	X	—
End Page Map	X	X	X	X	X	X	X	X	X	—
Resource Environment Group										
Begin Resource Environment Group	X	X	—	—	—	—	—	X	X	—
[Map Data Resource]...	X	X	—	—	—	—	—	X	X	—
[Map Page Overlay]...	X	X	—	—	—	—	—	X	X	—
End Resource Environment Group	X	X	—	—	—	—	—	X	X	—
Data Map										
Begin Data Map	X	X	X	X	X	X	X	X	X	—
[Data Format Parameter]!	X	X	—	—	X	X	X	X	X	—
[Margin Definition]!	X	X	—	—	X	X	X	X	X	—
Begin Active Environment Group	X	X	X	X	X	X	X	X	X	—
[Map Coded Font, fmt 1]...	X	X	X	X	X	X	X	X	X	—

Table 16. Resource Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[Map Coded Font, fmt 2]...	X	X	X	X	X	X	X	X	X	—
[Coded Font Name Reference]!	X	X	X	X	X	X	X	X	X	—
[Font Descriptor triplet]!	X	X	X^4	X^4	X	X	X	X	X	—
[Horizontal Scale Factor triplet]	X	X	X^4	X^4	X	X	X	X	X	—
[Map Data Resource]...	X	X	—	—	X	X	X	X	X	—
[Map Page Overlay]...	X	X	X	X	X	X	X	X	X	—
[Map Page Segment]...	X	X	X	X	X	X	X	X	X	—
Page Descriptor	X	X	X	X	X	X	X	X	X	—
[Object Area Descriptor (text only)] (See Note 8)	X	X	X	X	X	X	X	X	X	—
[Object Area Position (text only)] (See Note 8)	X	X	X	X	X	X	X	X	X	—
[Presentation Text Descriptor, fmt 1] (See Note 9)	X	X	X	X	X	X	X	X	X	—
[Presentation Text Descriptor, fmt 2] (See Note 9)	X	X	X	X	X	X	X	X	X	—
End Active Environment Group	X	X	X	X	X	X	X	X	X	—
Begin Data Map Transmission Subcase	X	X	X	X	X	X	X	X	X	—
Data Map Transmission Subcase Descriptor	X	X	X	X	X	X	X	X	X	—
Line Descriptor Count	X	X	X	X	X	X	X	X	X	—
Line Descriptor...	X	X	X	X	X	X	X	X	X	—
[DBCS coded font selection]	X	X	X^4	X^4	X	X	X	X	X	—
[Relative Baseline]	X	X	X^4	X^4	X	X	X	X	X	—
[Resource Object Include triplet]!	X	X	X^4	X^4	X	X	X	X	X	—
[Bar Code Descriptor triplet]!	X	X	X^4	X^4	X	X	X	X	X	—
[Color Specification triplet]!	X	X	—	X^4	X	X	X	X	X	—

Table 16. Resource Object Structured Fields (continued)

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
Record Descriptor...	X	X	—	—	X	X	X	X	X	—
XML Descriptor...	X	X	—	—	X	X	X	X	X	—
[Fixed Data Size] (See Note 10)	X	X	X	X	X	X	X	X	X	—
[Fixed Data Text]...	X	X	X	X	X	X	X	X	X	—
End Data Map Transmission Subcase	X	X	X	X	X	X	X	X	X	—
End Data Map	X	X	X	X	X	X	X	X	X	—

Notes:

1. See *Advanced Function Printing: Host Font Data Stream Reference*

2. Fonts in a resource group are by the AFP Viewer.

3. MCFs are not allowed in graphics objects or bar code objects that are included in page segment objects.

4. Support of these structures is provided by PTFs to the release.

5. These structures are processed as NOPs for printing. They are processed for viewing.

6. This structured field is required in a Document Environment group if it is not specified in each Medium Map in the Form Definition.

7. A structured field defined in a Medium Map takes precedence over the same structured field defined in a Document Environment Group.

8. These structured fields are ignored by AFP print servers and the AFP Viewer.

9. This structured field is required if presentation text objects are present in the page; the AEG must have either fmt 1 or fmt 2, but cannot have both.

10. Required structured field if Fixed Data Text structured fields appear in the Data Map.

11. Only character encoded names are supported on the MMT structured field.

Table 17. Structured Fields and Objects in Line Data

Functions	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5R2.0	i5/OS V5R3.0	i5/OS V5R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
[Presentation Text]...	X	X	X	X	X	X	X	X	X	—
[Include Page Segment]...	X	X	X	X	X	X	X	X	X	—
[Include Page Overlay]...	X	X	X	X	X	X	X	X	X	—
[Include Object]...	X	X	—	X[1]	X	X	X	X	X	—
image object (IM)]...	X	X	X	X	X	X	X	X	X	—
image object (IO)]...	X	X	X	X	X	X	X	X	X	—
graphics object]...	X	X	X	X	X	X	X	X	X	—
bar code object]...	X	X	X	X	X	X	X	X	X	—

Notes:

1. Support of these structures is provided by PTFs to the release.

System Support Information

Table 18. Document Index Structured Fields

Structured Fields	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
(See Note 1)										
Begin Document Index	X	X	X	X	X	X	X	X	X	X
[Index Element]!...	X	X	X	X	X	X	X	X	X	X
[Link Logical Element]!...	X	X	X	X	X	X	X	X	X	X
[Tag Logical Element]!...	X	X	X	X	X	X	X	X	X	X
End Document Index	X	X	X	X	X	X	X	X	X	X

Notes:

1. These structures are processed as NOPs for printing. They are processed for viewing.

Table 19. Data Stream Functions

Functions	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
MO:DCA IS/1	X	X	X	X	X	X	X	X	X	X
MO:DCA IS/2	—	—	—	—	—	—	—	—	—	X^1
Code page ID for Names (X'01' Triplet)	X^4	X^4	—	—	—	X^4	X^4	—	—	X
Token Name Override Using FQN type X'01' Triplet	X^4	X^4	—	X^3	X^2	X^4	X^4	X	X	—

Notes:

1. The AFP Viewer supports most of MO:DCA-P IS/2; it does not support BCOCA objects.
2. Supported only on the Include Object (IOB) structured field.
3. Supported with PTFs to the release, but only on the Include Object (IOB) structured field.
4. Supported only on the Include Object (IOB), Begin Resource (BRS), and Map Data Resource (MDR) structured fields.

Table 20. Non-OCA Object Formats

Functions	PSF z/OS V3 R4	PSF z/OS V4 R1	PSF/VM V2 R1.1	PSF/VSE V2 R2.1	i5/OS V5 R2.0	i5/OS V5 R3.0	i5/OS V5 R4.0	Infoprint AIX V4 R2	Infoprint Windows V2 R2	AFP Viewer
(See Note 1)										
EPS (OID comp# 13)	X	X	—	—	—	X	X	X	X	X[3]
TIFF (OID comp# 14)	X[4]	X[4]	—	—	—	X[2]	X[2]	X	X[4]	X
COM Set-Up (OID comp# 15)	X	X	—	—	—	—	—	X	X	—
Tape Label Set-Up (OID comp# 16)	—	—	—	—	—	—	—	X	X	—
DIB-Windows (OID comp# 17)	—	—	—	—	—	—	—	—	—	X
DIB-OS/2 (OID comp# 18)	—	—	—	—	—	—	—	—	—	X
PCX (OID comp# 19)	—	—	—	—	—	—	—	—	—	X
Color Mapping Table (OID comp# 20)	X	X	—	—	—	X	X	X	X	—
GIF (OID comp# 22)	X[4]	X[4]	—	—	—	X[2]	X[2]	X	X[4]	X
JFIF (OID comp# 23)	X[4]	X[4]	—	—	—	X[2]	X[2]	X	X[4]	X
Anastak Control Record (OID comp# 24)	—	—	—	—	—	—	—	X	X	—
PDF Single-Page Object (OID comp# 25)	X	X	—	—	—	X	X	X	—	X[3]
PDF Resource Object (OID comp# 26)	X	X	—	—	—	X	X	X	—	X[3]
PCL Page Object (OID comp# 34)	—	—	—	—	—	X[2]	X[2]	X	—	—

Notes:

1. Some printing products will use a transform to convert the non-OCA object to an OCA object if the printer does not support the non-OCA object format. Please refer to your product documentation for more details.
2. Supported in Infoprint Server PDF Subsystem.
3. Supported for Infoprint Color 130 Plus.
4. Support of these structures is provided by PTFs to the release.

The complete registry for object OIDs, as well as a summary table for object OIDs that may be included with an IOB structured field is provided in the *Mixed Object Document Content Architecture Reference*, SC31-6802.

Summary of Changes

This fifth edition of the *AFP Programming Guide and Line Data Reference* contains the following changes:

- A new USPS Four-State Bar Code symbology
- A major extension to allow the use of Color Management Resources to be mapped and used in a page definition
- A clarification of the interaction between PTX structured fields imbedded in line data and the line descriptor processing performed for each PTX structured field processed
- Additional support for using extended color specifications with bar code objects

Technical changes are marked by revision bars in the left margin of a page.

Notices

This information was developed for products and services offered in the U.S.A.

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 rights 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 give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

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) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2–31 Roppongi 3–chome, Minato-ku
Tokyo 106, Japan

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
Department 11PA Building 002S
PO Box 1900
Boulder CO 80301 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 measurement 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 illustrates 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. 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 IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

For online versions of this book, we authorize you to:
- Copy, modify, and print the documentation contained on the media, for use within your enterprise, provided you reproduce the copyright notice, all warning statements, and other required statements on each copy or partial copy.
- Transfer the original unaltered copy of the documentation when you transfer the related IBM product (which may be either machines you own, or programs, if the program's license terms permit a transfer). You must, at the same time, destroy all other copies of the documentation.

You are responsible for payment of any taxes, including personal property taxes, resulting from this authorization.

Your failure to comply with the terms above terminates this authorization. Upon termination, you must destroy your machine readable documentation.

Trademarks

The following terms, used in this publication, are trademarks of the IBM Corporation in the United States or other countries or both:

Advanced Function Presentation
Advanced Function Printing
AIX
AFP
Application System/400
AS/400
Bar Code Object Content Architecture
BCOCA
CMOCA
Color Management Object Content Architecture
eServer
GDDM
IBM
ImagePlus

Infoprint
Intelligent Printer Data Stream
IPDS
iSeries
Mixed Object Document Content Architecture
MO:DCA
MVS
MVS/ESA
OS/2
OS/390
Operating System/2
Print Services Facility
S/390
SAA
System i5
System/370
System/390
Systems Application Architecture
z/OS
zSeries
z/VM
z/VSE

Microsoft and Windows are registered trademarks
of Microsoft Corporation in the United States,
other countries, or both.

Other company, product, or service names may be
the trademarks or service marks of others.

Glossary

Some of the terms and definitions that appear in this glossary have been taken from other source documents.

If you do not find the term that you are looking for, please refer to the *IBM Dictionary of Computing*, document number SC20-1699.

The following definitions are provided as supporting information only, and are not intended to be used as a substitute for the semantics described in the body of this reference.

A

absolute coordinate. One of the coordinates that identify the location of an addressable point with respect to the origin of a specified coordinate system. Contrast with *relative coordinate*.

absolute positioning. The establishment of a position within a coordinate system as an offset from the coordinate system origin. Contrast with *relative positioning*.

addressable position. A position in a presentation space or on a physical medium that can be identified by a coordinate from the coordinate system of the presentation space or physical medium. See also *picture element*.

Advanced Function Presentation (AFP). The IBM strategic environment for presentation.

AFP. See *Advanced Function Presentation*.

AFP data stream. A presentation data stream that is processed in AFP environments. MO:DCA-P is the strategic AFP interchange data stream. IPDS is the strategic AFP printer data stream.

AFPDS. A term formerly used to identify the composed page MO:DCA-based data stream interchanged in AFP environments. See also *MO:DCA-P* and *AFP data stream*.

all points addressable (APA). The capability to address, reference, and position data elements at any addressable position in a presentation space or on a physical medium. Contrast with character cell addressing, in which the presentation space is divided into a fixed number of character-size rectangles in which characters can appear. Only the cells are addressable. An example of all points addressability is the positioning of text, graphics, and images at any addressable point on the physical medium. See also *picture element*.

annotation. A process by which additional data or attributes, such as highlighting, are associated with a page or a position on a page. Application of this data or attributes to the page is typically under the control of the user. Common functions such as applying adhesive removable notes to paper documents or using a transparent highlighter are emulated electronically by the annotation process.

APA. See *all points addressable*.

application. The use to which an information system is put.

application program. A program written for or by a user that applies to the user's work.

architected. Identifies data that is defined and controlled by an architecture. Contrast with *unarchitected*.

aspect ratio. The ratio of the horizontal size of a picture to the vertical size of the picture.

attribute. A property or characteristic of one or more constructs.

B

background. The part of a presentation space that is not occupied with object data.

bar code. An array of parallel rectangular bars and spaces that together represent data elements or characters in a particular symbology. The bars and spaces 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.

bar code presentation space. A two-dimensional conceptual space in which bar code symbols are generated.

base LND. The first LND used to process an input line-data record. See also *reuse LND*.

baseline. A conceptual line with respect to which successive characters are aligned.

211

baseline direction (B). The direction in which successive lines of text appear on a logical page. Synonymous with *baseline progression* and *B-direction*.

baseline progression (B). Synonymous with *baseline direction* and *B-direction*.

BCOCA. See *Bar Code Object Content Architecture*.

B-direction (B). Synonymous with *baseline direction* and *baseline progression*.

BITS. A data type for architecture syntax, indicating one or more bytes to be interpreted as bit string information.

C

CCSID. See *Coded Character Set Identifier*.

CGCSGID. See *Coded Graphic Character Set Global Identifier*.

CHAR. A data type for architecture syntax, indicating one or more bytes to be interpreted as character information.

character. A member of a set of elements used for the organization, control, or representation of data. A character can be either a graphic character or a control character.

character baseline. A conceptual reference line that is coincident with the X-axis of the character coordinate system.

character increment. A character's character increment is the distance the inline coordinate is incremented when that character is placed in a presentation space or on a physical medium. Character increment is a property of each graphic character in a font and of the font's character rotation.

character rotation. The alignment of a character with respect to its character baseline, measured in degrees in a clockwise direction. Examples are 0°, 90°, 180°, and 270°. Zero-degree character rotation exists when a character is in its customary alignment with the baseline. Contrast with *rotation*.

character set. A finite set of different graphic or control characters that is complete for a given purpose. For example, the character set in ISO Standard 646, *7-bit Coded Character Set for Information Processing Interchange*.

character string. A sequence of characters.

clipping. Eliminating those parts of a picture that are outside of a clipping boundary such as a viewing window or presentation space. Synonymous with *trimming*.

CMOCA. Color Management Object Content Architecture.

CMR. Color management resource.

CODE. A data type for architecture syntax that indicates an architected constant to be interpreted as defined by the architecture.

Coded Character Set Identifier (CCSID). A 16-bit number identifying a specific set of encoding scheme identifier, character set identifiers, code page identifiers and other relevant information that uniquely identifies the coded graphic character representation used.

coded font. A resource containing elements of a code page and a font character set, used for presenting text, graphics character strings, and bar code HRI. See also *code page* and *font character set*.

coded font local identifier. A binary identifier that is mapped by the environment to a named resource to identify a coded font. See also *local identifier*.

coded graphic character. A graphic character that has been assigned one or more code points within a code page.

coded graphic character set. A set of graphic characters with their assigned code points.

Coded Graphic Character Set Global Identifier (CGCSGID). A four-byte binary or a ten-digit decimal identifier consisting of the concatenation of a GCSGID and a CPGID. The CGCSGID identifies the code point assignments in the code page for a specific graphic character set, from among all the graphic characters that are assigned in the code page.

code page. A set of assignments, each of which assigns a code point to a character. Each code page has a unique name or identifier. Within a given code page, a code point is assigned to one character. More than one character set can be assigned code points from the same code page. See also *code point* and *section*.

Code Page Global Identifier (CPGID). A unique code page identifier that can be expressed as either a two-byte binary or a five-digit decimal value.

code point. A unique bit pattern that can serve as an element of a code page or a site in a code table, to which a character can be assigned. The element is associated with a binary value. The assignment of a character to an element of a code page determines the binary value that will be used to represent each occurrence of the character in a character string. Code points are one or more bytes long. See also *section*.

color image. Images whose image data elements are represented by multiple bits or whose image data element values are mapped to color values. Constructs that map image data element values to color values are

look-up tables and image data element structure parameters. Examples of color values are screen color values for displays and color toner values for printers.

color management. The technology to calibrate the color of input devices (such as scanners or digital cameras), display devices and output devices (such as printers or offset presses). Parts of this technology are implemented in the operating system, the API, or directly in the application.

color management resource. An object that provides color management in presentation environments.

color model. See *color space*.

color space. The method by which a color is specified. For example, the RGB color space specifies color in terms of three intensities for red (R), green (G), and blue (B). Sometimes also referred to as *color model*.

color of medium. The color of a presentation space before any data is added to it. Synonymous with *reset color*.

color table. A collection of color element sets. The table can also specify the method used to combine the intensity levels of each element in an element set to produce a specific color. Examples of methods used to combine intensity levels are the additive method and the subtractive method. See also *color model*.

controlling environment. The environment in which an object is embedded, for example, the IPDS and MO:DCA data streams.

control sequence. A sequence of bytes that specifies a control function. A control sequence consists of a control sequence introducer and zero or more parameters.

coordinate system. A Cartesian coordinate system. An example is the image coordinate system that uses the fourth quadrant with positive values for the Y-axis. The origin is the upper left-hand corner of the fourth quadrant. A pair of (x,y) values corresponds to one image point. Each image point is described by an image data element.

coordinates. A pair of values that specify a position in a coordinate space. See also *absolute coordinate* and *relative coordinate*.

copy group. A set of copy subgroups that specify all copies of a sheet. In the MO:DCA architecture, a copy group is specified within a Medium Map. See also *copy subgroup*.

copy modification. The process of adding, deleting, or replacing data on selected copies of a presentation space.

copy subgroup. A part of a copy group that specifies a number of identical copies of a sheet and all modifications to those copies. Modifications include the media source, medium overlays to be presented on the sheet, text suppressions, and either simplex or duplex presentation. In the MO:DCA architecture, copy subgroups are specified by repeating groups in the Medium Copy Count structured field in a Medium Map. See also *copy group*.

CPGID. See *Code Page Global Identifier*.

D

data element. A unit of data that is considered indivisible.

Data Map. A print control object in a Page Definition that establishes the page environment and specifies the mapping of line data to the page. Synonymous with *Page Format*.

data stream. A continuous stream of data that has a defined format. An example of a defined format is a structured field.

default. A value, attribute, or option that is assumed when none has been specified and one is needed to continue processing.

default indicator. A field whose bits are all B'1' indicating that a hierarchical default value is to be used. The value may be specified by an external parameter.

device dependent. Dependent upon one or more device characteristics. An example of device dependency is a font whose characteristics are specified in terms of addressable positions of specific devices.

document. (1) A machine-readable collection of one or more objects which represent a composition, a work, or a collection of data. (2) A publication or other written material.

document component. A set of related structured fields that are bounded by *begin* and *end* structured fields. Examples are object, page, and overlay.

document content architecture. A family of architectures that define the syntax and semantics of document components. See also *document component* and *structured field*.

document element. A self-identifying, variable-length, bounded record, that can have a content portion that provides control information, data, or both. An application or device does not have to understand control information or data to parse a data stream when all the records in the data stream are document elements. See also *structured field*.

document formatting. A method used to determine where information is positioned in presentation spaces or on physical media.

document presentation. A method used to produce a visible copy of formatted information on physical media.

double-byte character set (DBCS). A character set that can contain up to 65536 characters.

double-byte coded font. A coded font in which the code points are two bytes long.

duplex. A method used to print data on both sides of a sheet. Normal-duplex printing occurs when the sheet is turned over the Y_m axis. Tumble-duplex printing occurs when the sheet is turned over the X_m axis.

duplex printing. A method used to print data on both sides of a sheet. Contrast with *simplex printing*.

E

EBCDIC. See *Extended Binary-Coded Decimal Interchange Code*.

element. A structured field in a document content architecture data stream.

Em square. A square layout space used for designing each of the characters of a font.

encoding scheme. A set of specific definitions that describe the philosophy used to represent character data. The number of bits, the number of bytes, the allowable ranges of bytes, the maximum number of characters, and the meanings assigned to some generic and specific bit patterns, are some examples of specifications to be found in such a definition.

Encoding Scheme Identifier (ESID). A 16-bit number assigned to uniquely identify a particular encoding scheme specification. See also *encoding scheme*.

ESID. See *Encoding Scheme Identifier*.

exception. An invalid or unsupported data-stream construct.

exception action. Action taken when an exception is detected.

exception condition. The condition that exists when a product encounters an invalid or unsupported construct.

exchange. The predictable interpretation of shared information by a family of system processes in an environment where the characteristics of each process must be known to all other processes. Contrast with *interchange*.

Extended Binary-Coded Decimal Interchange Code (EBCDIC). A coded character set consisting of eight-bit coded characters.

external parameter. A parameter for which the current value can be provided by the controlling environment, for example, the data stream, or by the application itself.

F

factoring. The movement of a parameter value from one state to a higher-level state. This permits the parameter value to apply to all of the lower-level states unless specifically overridden at the lower level.

FGID. See *Font Typeface Global Identifier*.

final form data. Data that has been formatted for presentation.

fixed medium information. Information that can be applied to a sheet by a printer or printer-attached device that is independent of data provided through the data stream. Fixed medium information does not mix with the data provided by the data stream and is presented on a sheet either before or after the text, image, graphics, or bar code data provided within the data stream. Fixed medium information can be used to create pre-printed forms, or other types of printing, such as colored logos or letterheads, that cannot be created conveniently within the data stream.

FOCA. See *Font Object Content Architecture*.

font. A set of graphic characters that have a characteristic design, or a font designer's concept of how the graphic characters should appear. The characteristic design specifies the characteristics of its graphic characters. Examples of characteristics are shape, graphic pattern, style, size, weight, and increment. Examples of fonts are fully described fonts, symbol sets, and their internal printer representations. See also *coded font* and *symbol set*.

font character set. A FOCA resource containing descriptive information, font metrics, and the digital representation of character shapes for a specified graphic character set.

Font Typeface Global Identifier (FGID). A unique font identifier that can be expressed as either a two-byte binary or a five-digit decimal value. The FGID is used to identify a type style and the following characteristics or parameters: posture, weight, and width.

font height (FH). Synonymous with *vertical font size*.

font metrics. Measurement information that defines individual character values such as height, width, and space, as well as overall font values such as averages

and maximums. Font metrics can be expressed in specific fixed units, such as pels, or in relative units that are independent of both the resolution and the size of the font.

font object.   A resource object which contains some or all of the description of a font.

Font Object Content Architecture (FOCA).   An architected collection of constructs used to describe fonts and to interchange those font descriptions.

font referencing.   A method used to identify or characterize a font. Examples of processes that use font referencing are document editing, formatting, and presentation.

font width (FW).   Synonymous with *horizontal font size*.

foreground.   The part of a presentation space that is occupied with object data.

form.   A physical entity on which information is printed. An example of a form is one piece of paper. Synonymous with *sheet*.

format.   The arrangement or layout of data on a physical medium or in a presentation space.

formatter.   A process used to prepare a document for presentation.

Formdef.   See *Form Definition*.

Form Definition (Formdef).   Synonymous with *Form Map*.

Form Map.   A print control object that contains an environment definition and one or more Medium Maps. Synonymous with *Form Definition*. See also *Medium Map*.

function set.   A collection of architecture constructs and associated values. Function sets can be defined across or within subsets.

FW.   See *font width*.

G

GCGID.   See *Graphic Character Global Identifier*.

GCSGID.   *Graphic Character Set Global Identifier*.

GID.   See *global identifier*.

Global Identifier (GID).   One of the following:
- A Code Page Global ID (CPGID)
- A Graphic Character Global Identifier (GCGID)
- A Font Typeface Global Identifier (FGID)
- A Graphic Character Set Global Identifier (GCSGID)

- An encoded graphic character string that, when qualified by the associated CGCSGID, provides a reference name for a document element

global resource identifier (GRID).   An eight-byte identifier that identifies a coded font resource. A GRID contains the following fields in the order shown:

1. GCSGID of a minimum set of graphic characters required for presentation. It can be a character set that is associated with the code page, or with the font character set, or with both.
2. CPGID of the associated code page.
3. FGID of the associated font character set.
4. Font width in 1440ths of an inch.

glyph.   A member of a set of symbols which represent data. Glyphs may be letters, digits, punctuation marks, or other symbols. Synonymous with *graphic character*.

GOCA.   See *Graphics Object Content Architecture*.

graphic character.   A member of a set of symbols which represent data. Graphic characters can be letters, digits, punctuation marks, or other symbols. Synonymous with *glyph*. See also *character*.

Graphic Character Global Identifier (GCGID).   An alphanumeric character string used to identify a specific graphic character. A GCGID can be from four bytes to eight bytes long.

Graphic Character Set Global Identifier (GCSGID).   A unique graphic character set identifier that can be expressed as either a two-byte binary or a five-digit decimal value.

graphics data.   Data containing lines, arcs, markers, and other constructs that describe a picture.

graphics object.   An object that contains graphics data. See also *object*.

Graphics Object Content Architecture (GOCA).   An architected collection of constructs used to interchange and present graphics data.

graphics presentation space.   A two-dimensional conceptual space in which a picture is generated. In this space graphics drawing orders are defined. The picture can then be mapped onto an output medium. All viewing transforms are completed before the picture is generated for presentation on an output medium.

grayscale image.   Images whose image data elements are represented by multiple bits and whose image data element values are mapped to more than one level of brightness through an image data element structure parameter or a look-up table.

GRID.   See *global resource identifier*.

H

hexadecimal. A number system with a base of sixteen. The decimal digits 0 through 9 and characters A through F are used to represent hexadecimal digits. The hexadecimal digits A through F correspond to the decimal numbers 10 through 15, respectively. An example of a hexadecimal number is X'1B', which is equal to the decimal number 27.

highlight color. A spot color that is used to accentuate or contrast monochromatic areas. See also *spot color*.

hollow font. A font design in which the graphic character shapes include only the outer edges of the strokes.

horizontal font size. (1) A characteristic value, parallel to the character baseline, that represents the size of all graphic characters in a font. Synonymous with *font width*. (2) In a font character set, nominal horizontal font size is a font-designer defined value corresponding to the nominal character increment for a font character set. The value is generally the width of the space character, and is defined differently for fonts with different spacing characteristics.
1. For fixed-pitch, uniform character increment fonts: the fixed character increment, which is also the space character increment
2. For PSM fonts: the width of the space character
3. For typographic, proportionally-spaced fonts: one third of the vertical font size, which is also the default size of the space character

The font designer can also define a minimum and maximum horizontal font size to represent the limits of scaling. :gd.In font referencing, the specified horizontal font size is the desired size of the font when the characters are presented. If this size is different from the nominal horizontal font size specified in a font character set, the character shapes and character metrics might need to be scaled prior to presentation.

horizontal scale factor. In outline-font referencing, the specified horizontal adjustment of the Em square. The horizontal scale factor is specified in 1440ths of an inch. When the horizontal and vertical scale factors are different, anamorphic scaling occurs. See also *vertical scale factor*.

I

ID. Identifier.

IDE. See *image data element*.

I-direction. Synonymous with *inline direction*.

image. An electronic representation of a picture produced by means of sensing light, sound, electron radiation, or other emanations coming from the picture or reflected by the picture. An image can also be generated directly by software without reference to an existing picture.

image content. Image data and its associated image data parameters.

image coordinate system. An X,Y Cartesian coordinate system using only the fourth quadrant with positive values for the Y-axis. The origin of an image coordinate system is its upper left hand corner. An X,Y coordinate specifies a presentation position which corresponds to one and only one image data element in the image content.

image data. Rectangular arrays of raster information that define an image.

image data element (IDE). A basic unit of image information. An image data element expresses the intensity of a signal at a corresponding image point. An image data element can use a look-up table to introduce a level of indirection into the expression of grayscale or color.

image distortion. Deformation of an image such that the original proportions of the image are changed and the original balance and symmetry of the image are lost.

image object. An object which contains image data. See also *object*.

Image Object Content Architecture (IOCA). An architected collection of constructs used to interchange and present images.

image point. A discrete X,Y coordinate in the image presentation space. See also *addressable position*.

image presentation space. A two-dimensional conceptual space in which an image is generated. It can then be mapped onto an output medium.

IM image. A migration image object that is resolution-dependent, bi-level, and that cannot be compressed or scaled. Contrast with *IO image*.

indexed object. An object in a MO:DCA document that is referenced by an Index Element structured field in a MO:DCA index. Examples of indexed objects are pages and page groups.

inline direction (I). The direction in which successive characters appear in a line of text. Synonymous with *I-direction*.

inline resource. A resource object carried in a resource group that preceeds all documents in an AFP print file.

Intelligent Printer Data Stream (IPDS). An architected host-to-printer data stream that contains both data and controls defining how the data is to be presented.

interchange. The predictable interpretation of shared information in an environment where the characteristics of each process need not be known to all other processes. Contrast with *exchange*.

interoperability. In SAA® usage, the ability to link SAA and non-SAA environments and use the combination for distributed processing. See also *SAA environments*.

IOCA. See *Image Object Content Architecture*.

IO image. An image object containing IOCA constructs. Contrast with *IM image*.

IPDS. See *Intelligent Printer Data Stream*.

K

keyword. A two-part self-defining parameter consisting of a one-byte identifier and a one-byte value.

L

landscape. A presentation orientation in which the X_m axis is parallel to the long sides of a rectangular physical medium. Contrast with *portrait*.

language. A set of symbols, conventions, and rules that is used for conveying information. See also *pragmatics*, *semantics*, and *syntax*.

LID. See *local identifier*.

local identifier (LID). An identifier that is mapped by the environment to a named resource.

location. A site within a data stream. A location is specified in terms of an offset in the number of structured fields from the beginning of a data stream, or in the number of bytes from another location within the data stream.

logical page. A presentation space. One or more object areas or data blocks can be mapped to a logical page. A logical page has specifiable characteristics, such as size, shape, orientation, and offset. The shape of a page is the shape of a rectangle. Orientation and offset are specified relative to a medium coordinate system.

logical unit. A unit of linear measurement expressed with a unit base and units per unit-base value. For example, in the MO:DCA and IPDS architectures, the following logical units are used:

- 1 logical unit = 1/1440 inch (unit base = 10 inches, units per unit base = 14400)
- 1 logical unit = 1/240 inch (unit base = 10 inches, units per unit base = 2400)

Synonymous with *L-unit*.

look-up table (LUT). A logical list of colors or intensities. The list has a name and can be referenced to select a color or intensity. See also *color table*.

L-unit. Synonymous with *logical unit*.

LUT. See *look-up table*.

M

meaning. A table heading for architecture syntax. The entries under this heading convey the meaning or purpose of a construct. A meaning entry can be a long name, a description, or a brief statements of function.

media. Plural of medium. See also *medium*.

media destination. The destination to which sheets are sent as the last step in the print process. Some printers support several media destinations to allow options such as print job distribution to one or more specific destinations, and routing output to a specific destination for security reasons. Contrast with *media source*.

media source. The source from which sheets are obtained for printing. Some printers support several media sources so that media with different characteristics (such as size, color, and type) can be selected when desired. Contrast with *media destination*.

medium. A two-dimensional conceptual space with a base coordinate system from which all other coordinate systems are either directly or indirectly derived. A medium is mapped onto a physical medium in a device-dependent manner. Synonymous with *medium presentation space*. See also *logical page*, *physical medium*, and *presentation space*.

Medium Map. A print control object in a Form Map that defines resource mappings and controls modifications to a form, page placement on a form, and form copy generation. See also *Form Map*.

medium presentation space. A two-dimensional conceptual space with a base coordinate system from which all other coordinate systems are either directly or indirectly derived. A medium presentation space is mapped onto a physical medium in a device-dependent manner. Synonymous with *medium*. See also *logical page*, *physical medium*, and *presentation space*.

Mixed Object Document Content Architecture (MO:DCA). An architected, device-independent data stream for interchanging documents.

MO:DCA. See *Mixed Object Document Content Architecture*.

MO:DCA-L. MO:DCA Resource Interchange Set. A subset of MO:DCA that defines an interchange format for resource documents. Contrast with *MO:DCA-P IS/1* and *MO:DCA-P IS/2*.

MO:DCA-P. The subset of MO:DCA that defines presentation documents.

MO:DCA-P IS/1. MO:DCA Presentation Interchange Set 1. A subset of MO:DCA-P that defines an interchange format for presentation documents. See also **MO:DCA-P IS/2**. Contrast with *MO:DCA-L*.

MO:DCA-P IS/2. MO:DCA Presentation Interchange Set 2. A subset of MO:DCA-P that defines an interchange format for presentation documents that is a superset of MO:DCA-P IS/1. See also *MO:DCA-P IS/1*. Contrast with *MO:DCA-L*.

N

name. A table heading for architecture syntax. The entries under this heading are short names that give a general indication of the contents of the construct.

named color. A color that is specified with a descriptive name. An example of a named color is "green".

nested resource. A resource that is invoked within another resource using either an Include command or a local ID. See also *nesting resource*.

nesting coordinate space. A coordinate space which contains another coordinate space. Examples of coordinate spaces are medium, overlay, page and object area.

nesting resource. A resource that invokes nested resources. See also *nested resource*.

no operation (NOP). A construct whose execution causes a product to proceed to the next instruction to be processed without taking any other action.

NOP. See *no operation*.

N-up. The presentation of a fixed number of pages on a side of a physical medium. For example, 4-up is the presentation of four pages on a side.

O

object. A collection of structured fields. The first structured field provides a begin-object function and the last structured field provides an end-object function. The object can contain one or more other structured fields whose content consists of one or more data elements of a particular data type. An object can be assigned a name, which can be used to reference the object. Examples of objects are image, graphics, text, page segment, and document index objects.

object area. A rectangular area in a presentation space into which a data object is mapped. The presentation space can be for a page or an overlay. Examples are a

graphics object area, an image object area, and a bar code object area. Synonymous with *data block*.

object data. A collection of related data elements that have been bundled together. Examples of object data include graphic characters, image data elements, and drawing orders.

offset. A table heading for architecture syntax. The entries under this heading indicate the numeric displacement into a construct. The offset is measured in bytes and starts with byte zero. Individual bits can be expressed as displacements within bytes.

orientation. The angular distance a presentation space or object area is rotated in a specified coordinate system, expressed in degrees and minutes. For example, the orientation of printing on a physical medium, relative to the X_m axis of the X_m,Y_m coordinate system.

origin. The point in a coordinate system where the axes intersect. An example of an origin is the addressable position in an X_m,Y_m coordinate system where both coordinate values are zero.

orthogonal. Intersecting at right angles. An example of orthogonal is the positional relationship between the axes of a Cartesian coordinate system.

outline font. A shape technology in which the graphic character shapes are represented in digital form by a series of mathematical expressions that define the outer edges of the strokes. The resultant graphic character shapes can be either solid or hollow.

overlay. (1) A resource object that can contain text, image, graphics, and bar code data. Overlays define their own environment, and are often used as electronic forms. (2) The final representation of such an object on a physical medium. Contrast with *page segment*.

P

page. (1) A data stream object delimited by a Begin Page structured field and an End Page structured field. A page can contain text, image, graphics, and bar code data. (2) The final representation of such an object on a physical medium.

PageDef. See *Page Definition*.

Page Definition (PageDef). Synonymous with *Page Map*.

Page Map. A print control object used to format line data into page data. A Page Map contains one or more Data Maps and may optionally specify conditional processing of the line data. Synonymous with *Page Definition*. See also *Data Map*.

Page Format. Synonymous with *Data Map*.

page group. A named group of sequential pages. A page group is delimited by a Begin Named Page Group structured field and an End Named Page Group structured field. A page group may contain nested page groups. All pages in the page group inherit the attributes and processing characteristics that are assigned to the page group.

page segment. (1) In MO:DCA, a resource object that can contain any mixture of bar code objects, graphics objects, and IOCA image objects. A page segment does not contain an active environment group. The environment for a page segment is defined by the active environment group of the including page or overlay. (2) The final representation of such an object on a physical medium. Contrast with *overlay:*.

parameter. A variable that is given a constant value for a specified application.

pel. The smallest printable or displayable unit on a physical medium. In computer graphics, the smallest element of a physical medium that can be independently assigned color and intensity. Pels per inch is often used as a measurement of presentation granularity. Synonymous with *picture element* and *pixel*.

physical medium. A physical entity on which information is presented. Examples of a physical medium are a sheet of paper and a display screen. See also *medium* and *medium presentation space*.

picture element. Synonymous with *pel*.

pixel. Synonymous with *pel*.

point. A unit of measure used mainly for measuring typographical material. There are seventy-two points to an inch.

portrait. A presentation orientation in which the X_m axis is parallel to the short sides of a rectangular physical medium. Contrast with *landscape*.

position. A position in a presentation space or on a physical medium that can be identified by a coordinate from the coordinate system of the presentation space or physical medium. See also *pel*. Synonymous with *addressable position*.

pragmatics. Information related to the usage of a construct. See also *semantics* and *syntax*.

presentation device. A device that produces character shapes, graphics pictures, images, or bar code symbols on a physical medium. Examples of physical medium are a display screen and a sheet of paper.

presentation space. A conceptual address space with a specified coordinate system and a set of addressable positions. The coordinate system and addressable positions can coincide with those of a physical medium. Examples of a presentation space are medium,

page, and object area. See also *bar code presentation space, graphics presentation space, image presentation space, logical page, medium presentation space*, and *text presentation space*.

presentation text object. An object that contains presentation text data. See also *object*.

Presentation Text Object Content Architecture (PTOCA). An architected collection of constructs used to interchange and present presentation text data.

print control object. A resource object that contains layout, finishing, and resource mapping information used to present a document on physical media. Examples of print control objects are *Form Maps* and *Medium Maps*.

process color. A color that is specified as a combination of the components, or primaries, of a color space. A process color is rendered by mixing the specified amounts of the primaries. An example of a process color is C=.1, M=.8, Y=.2, K=.1 in the cyan/magenta/yellow/black (CMYK) color space. Contrast with *spot color*.

Proportional Spacing Machine font (PSM font). A font originating with the electric typewriter and having character increment values that are integer multiples of the narrowest character width.

PSM font. See *Proportional Spacing Machine font*.

PTOCA. See *Presentation Text Object Content Architecture*.

R

range. A table heading for architecture syntax. The entries under this heading give numeric ranges applicable to a construct. The ranges may be expressed in binary, decimal, or hexadecimal. The range can consist of a single value.

raster pattern. A rectangular array of pels arranged in rows called scan lines.

record-format line data. A form of line data where each record is preceded by a 10-byte identifier. The record is presented by matching its ID to the ID specified on a Record Descriptor in the Data Map of a Page Definition.

redaction. The process of applying an opaque mask over a page so that a selected portion of the page is visible. Because this function is typically used to prevent unauthorized viewing of data, an associated security level is also provided.

relative coordinate. One of the coordinates that identify the location of an addressable point by means of a displacement from some other addressable point. Contrast with *absolute coordinate*.

relative positioning. The establishment of a position within a coordinate system as an offset from the current position. Contrast with *absolute positioning*.

repeating group. A group of parameter specifications that may be repeated.

reserved. Having no assigned meaning and put aside for future use. The content of reserved fields is not used by receivers, and should be set by generators to a specified value, if given, or to binary zeros. A reserved field or value can be assigned a meaning by an architecture at any time.

reset color. The color of a presentation space before any data is added to it. Synonymous with *color of medium*.

resolution. (1) A measure of the sharpness of an input or output device capability, as given by some measure relative to the distance between two points or lines that can just be distinguished. (2) The number of addressable pels per unit of length.

resource. An object that is referenced by a data stream or by another object to provide data or information. Resource objects may be stored in libraries. In MO:DCA, resource objects can be contained within a resource group. Examples of resources are fonts, overlays, and page segments.

retired. Set aside for a particular purpose, and not available for any other purpose. Retired fields and values are specified for compatibility with existing products and identify one of the following:

- Fields or values that have been used by a product in a manner not compliant with the architected definition
- Fields or values that have been removed from an architecture

reuse LND. An LND in a chain of LNDs, also called a *reuse chain*, where all LNDs process fields in the same line-data record. See also *base LND*.

rotation. The orientation of a presentation space with respect to the coordinate system of a containing presentation space. Rotation is measured in degrees in a clockwise direction. Zero-degree rotation exists when the angle between a presentation space's positive X-axis and the containing presentation space's positive X-axis is zero degrees. Contrast with *character rotation*.

row. A subarray that consists of all elements that have an identical position within the high dimension of a regular two-dimensional array.

S

SAA. See *Systems Application Architecture*®.

SAA environments. Those environments in which IBM intends to provide full implementation of applicable SAA architectural elements. See also *interoperability*.

SBCS. See *single-byte character set*.

SBIN. A data type for architecture syntax that indicates that one or more bytes be interpreted as a signed binary number, with the sign bit in the high-order position of the leftmost byte. Positive numbers are represented in true binary notation with the sign bit set to B'0'. Negative numbers are represented in twos-complement binary notation with a B'1' in the sign-bit position.

scaling. Making all or part of a picture smaller or larger by multiplying the coordinate values of the picture by a constant amount. If the same multiplier is applied along both dimensions, the scaling is uniform and the proportions of the picture are unaffected. Otherwise, the scaling is anamorphic and the proportions of the picture are changed.

section. A portion of a double-byte code page that consists of 256 entries. The first byte of a two-byte code point is the section identifier. See also *code page* and *code point*.

section identifier. A value that identifies a section. Synonymous with *section number*.

section number. A value that identifies a section. Synonymous with *section identifier*.

semantics. The meaning of the parameters of a construct. See also *pragmatics* and *syntax*.

sheet. A physical entity on which information is printed. An example of a sheet is one piece of paper. Synonymous with *form*.

side. A physical surface of a sheet. A sheet has a front side and a back side. See also *sheet*.

simplex printing. A method used to print data on one side of a sheet; the other side is left blank. Contrast with *duplex printing*.

single-byte character set (SBCS). A character set that can contain up to 256 characters.

single-byte coded font. A coded font in which the code points are one byte long.

spot color. A color that is specified with a unique identifier such as a number. A spot color is normally rendered with a custom colorant instead of with a combination of process color primaries. See also *highlight color*. Contrast with *process color*.

structured field. A self-identifying, variable-length, bounded record that can have a content portion that provides control information, data, or both. See also *document element*.

structured field introducer. The header component of a structured field which provides information that is common for all structured fields. Examples of information that is common for all structured fields are length, function type, and category type. Examples of structured field function types are begin, end, data, and descriptor. Examples of structured field category types are presentation text, image, graphics, and page.

subpage. A part of a logical page on which line data may be placed. A line data record is identified as belonging to a particular subpage with the subpage identifier byte in the Line Descriptor (LND) structured field. Conditional processing can be used with a Page Definition to select a new Data Map and/or Medium Map to take effect before or after the current subpage is printed.

subset. Within the base-and-towers concept, a portion of architecture represented by a particular level in a tower or by a base.

suppression. A method used to prevent presentation of specified data. Examples of suppression are the processing of text data without placing characters on a physical medium and the electronic equivalent of the "spot carbon", that prevents selected data from being presented on certain copies of a presentation space or a physical medium.

symbol. A visual representation of something by reason of relationship, association, or convention.

symbol set. A coded font that is usually simpler in structure than a fully-described font. Symbol sets are used where typographic quality is not required. Examples of devices that might not provide typographic quality are dot-matrix printers and displays.

syntax. The rules governing the structure of a construct. See also *pragmatics* and *semantics*.

Systems Application Architecture (SAA). A set of IBM software interfaces, conventions, and protocols that provide a framework for designing and developing applications that are consistent across systems.

T

text. A graphic representation of information. Text can consist of alphanumeric characters and symbols arranged in paragraphs, tables, columns, and other shapes.

text presentation. The transformation of document graphic character content and its associated font information into a visible form. An example of a visible form of text is character shapes on a physical medium.

text presentation space. A two-dimensional conceptual space in which text is generated for presentation on an output medium.

toned. Containing marking agents such as toner or ink. Contrast with *untoned*.

trimming. Eliminating those parts of a picture that are outside of a clipping boundary such as a viewing window. Synonymous with *clipping*.

triplet. A three-part self-defining variable-length parameter consisting of a length byte, an identifier byte, and one or more parameter-value bytes.

triplet identifier. A one-byte type identifier for a triplet.

tumble-duplex printing. A method used to simulate the effect of physically turning a sheet around the X_m axis.

type. A table heading for architecture syntax. The entries under this heading indicate the types of data present in a construct. Examples include BITS, CHAR, CODE, SBIN, UBIN, UNDF.

typeface. All characters of a single type family or style, weight class, width class, and posture, regardless of size; for example, Sonoran Serif, Bold, Normal, Italics, in any point size.

type family. All characters of a single design, regardless of attributes such as width, weight, posture, and size. Examples are Courier and Gothic.

type structure. Attributes of characters other than type family or typeface. Examples are solid shape, hollow shape, and overstruck.

type style. The form of characters within the same font, for example, Courier or Gothic.

type weight. A parameter indicating the degree of boldness of a typeface. A character's stroke thickness determines its weight class. Examples are light, medium, and bold. Synonymous with *weight class*.

type width. A parameter indicating a relative change from the font's normal width-to-height ratio. Examples are normal, condensed, and expanded. Synonymous with *width class*.

typographic font. A font with graphic characters that have varying character increments. Proportional spacing can be used to provide the appearance of even spacing between presented characters and to eliminate excess blank space around narrow characters. An example of a narrow character is the letter *i*. Contrast with *uniformly spaced font*.

U

UBIN. A data type for architecture syntax, indicating one or more bytes to be interpreted as an unsigned binary number.

unarchitected. Identifies data that is neither defined nor controlled by an architecture. Contrast with *architected*.

UNDF. A data type for architecture syntax, indicating one or more bytes that are undefined by the architecture.

uniformly spaced font. A font with graphic characters having a uniform character increment. The distance between reference points of adjacent graphic characters is constant in the escapement direction. The blank space between the graphic characters can vary. Contrast with *typographic font*.

untoned. Unmarked portion of a physical medium. Contrast with *toned*.

V

vertical font size. (1) A characteristic value, perpendicular to the character baseline, that represents the size of all graphic characters in a font. Synonymous with *font height*. (2) In a font character set, nominal vertical font size is a font-designer defined value corresponding to the nominal distance between adjacent baseline when character rotation is zero degrees and no external leading is used. This distance represents the baseline-to-baseline increment that includes the maximum baseline extent and the designers recommendation for internal leading. The font designer can also define a minimum and maximum vertical font size to represent the limits of scaling. (3) In font referencing, the specified vertical font size is the desired size of the font when the characters are presented. If this size is different from the nominal vertical font size specified in a font character set, the character shapes and character metrics might need to be scaled before presentation.

vertical scale factor. In outline-font referencing, the specified vertical adjustment of the Em square. The vertical scale factor is specified in 1440ths of an inch. When the horizontal and vertical scale factors are different, anamorphic scaling occurs. See also *horizontal scale factor*.

W

weight class. A parameter indicating the degree of boldness of a typeface. A character's stroke thickness determines its weight class. Examples are light, medium, and bold. Synonymous with *type weight*.

width class. A parameter indicating a relative change from the font's normal width-to-height ratio. Examples are normal, condensed, and expanded. Synonymous with *type width*.

X

X_m,Y_m **coordinate system.** The medium coordinate system.

X_{oa},Y_{oa} **coordinate system.** The object area coordinate system.

X_{ol},Y_{ol} **coordinate system.** The overlay coordinate system.

X_{pg},Y_{pg} **coordinate system.** The coordinate system of a page presentation space. This coordinate system describes the size, position, and orientation of a page presentation space. Orientation of an X_{pg},Y_{pg} coordinate system is relative to an environment-specified coordinate system. An example of an environment-specified coordinate system is the X_m,Y_m coordinate system.

Index

A

Active Environment Group (AEG)
   in a Data Map   20
   in a Page Definition   20
   structured field list   188
   syntax   188
Additional Bar Code Parameters (X'7B') triplet   138, 157
Additional Bar Code Parameters (X'7B') Triplet   121
AEG (Active Environment Group)
   in a Data Map   20
   in a Page Definition   20
   structured field list   188
   syntax   188
AFP Viewer   183
AIX printing environment   9
AS/400 printing environment   9

B

bar code
   See Bar Code Data object
Bar Code Data object
   structured field list   188
   syntax   188
Bar Code Object Content Architecture (BCOCA)   2
bar code printing   36
Bar Code Symbol Descriptor (X'69') triplet   138, 157
Bar Code Symbol Descriptor (X'69') Triplet   116
base LND   30
BCOCA (Bar Code Object Content Architecture)   2
BDM (Begin Data Map)   75
BDX (Begin Data Map Transmission Subcase)   82
Begin Data Map (BDM)   75
Begin Data Map Transmission Subcase (BDX)   82
Begin Form Environment Group (BFG)   175, 179
Begin Named Page Group (BNG)   68
Begin Page Map (BPM)   82
BFG (Begin Form Environment Group)   175, 179
BNG (Begin Named Page Group)   68
BPM (Begin Page Map)   82

C

carriage control   27
CCP   83
CCP (Conditional Processing Control)   83
CCP identifier   83
CHARS parameter   22
CMOCA (Color Management Object Content Architecture)   2
code page identification support   201
Color Management Object Content Architecture (CMOCA)   2
Color Management Resource   125
Color Management Resource Descriptor (X'91') Triplet   125, 146, 158
Color Specification (X'4E') triplet   138
Color Specification (X'4E') Triplet   115, 157
Color Specification Triplet   115, 138
Composed-Text Control (CTC)   176
Composed-Text Data (CTX)   176
Composed-Text Descriptor (CTD)   176

Conditional Processing Control (CCP)   83
Conditional Processing Control (CCP) structured field
   in a Page Definition   16
conditional processing in a PAGEDEF
   example of   32
   in a Page Definition   31
copy group
   invoked with IMM   49
   structured field list   193
   syntax   193
CTC (Composed-Text Control)   176
CTD (Composed-Text Descriptor)   176
CTX (Composed-Text Data)   176
current line position   58
current LND position   56
current position   56
current RCD position   56

D

data field in a structured field   71
Data Map
   Active Environment Group in one   20
   Data Map Transmission Subcase in one   19, 27
   in a Page Definition   17, 19
Data Map Transmission Subcase   30
   in a Data Map   27
   in a Page Definition   27
Data Map Transmission Subcase Descriptor (DXD)   87
data objects
   structured field list   188
data objects in line data   60
data stream definition table   183
   data object structured fields   188
   document structured fields   184
   index structured fields   200
   resource object structured fields   193
data stream objects
   print file   44
DBCS support   34
DCF (Document Composition Facility)   67
diagrams
   Bar Code data object   170
   Form Definition resource object   172
   Graphics data object   169
   IM Image data object   168
   IO Image data object   168
   line format data   166
   Master Environment Group   169
   Mixed Line-Page Document   164, 165
   Overlay resource object   171
   Page Definition resource object   173
   Page Segment resource object   171
   Presentation Page object   165
   Presentation Text data object   167
   Print File   164
document
   data stream functions   201
   non-OCA objects   202
   structured field list   184, 200
   syntax   184, 200

U

W

X

Readers' Comments — We'd Like to Hear from You

Advanced Function Presentation
Programming Guide and Line Data Reference

Publication No. S544-3884-04

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	☐	☐	☐	☐	☐

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	☐	☐	☐	☐	☐
Complete	☐	☐	☐	☐	☐
Easy to find	☐	☐	☐	☐	☐
Easy to understand	☐	☐	☐	☐	☐
Well organized	☐	☐	☐	☐	☐
Applicable to your tasks	☐	☐	☐	☐	☐

Please tell us how we can improve this book:

Thank you for your responses. May we contact you?   ☐ Yes   ☐ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you. IBM or any other organizations will only use the personal information that you supply to contact you about the issues that you state on this form.

Name

Address

Company or Organization

Phone No.

Fold and Tape   **Please do not staple**   Fold and Tape
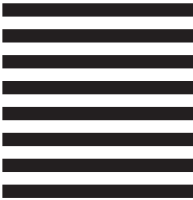
NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL   PERMIT NO. 40   ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
IBM Printing Systems Division
Department H7FE, Building 004N
Information Development
P.O. Box 1900
Boulder, CO USA   80301-9817

Fold and Tape   **Please do not staple**   Fold and Tape

IBM®

File Number: GENL-30

Printed in USA