Data Stream and Object Architectures

Mixed Object Document Content Architecture Reference

SC31-6802-06

Data Stream and Object Architectures

Mixed Object Document Content Architecture Reference

SC31-6802-06

Note!

Before using this information and the product it supports, read the information in "Notices" on page 565.

Seventh Edition (January 2004)

This edition applies to the IBM Mixed Object Document Content Architecture. It replaces and makes obsolete the previous edition, SC31-6802-05. This edition remains current until a new edition or Technical Newsletter is published.

Technical changes are indicated by a vertical bar to the left of the change. Editorial changes that have no technical significance are not noted. For a detailed list of changes, see "Summary of Changes" on page 563.

Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality. If you request publications from the address given below, your order will be delayed because publications are not stocked there. Many of the IBM Printing Systems Division publications are available from the web page listed below.

Internet

Visit our home page at: `http://www.ibm.com/printers`

A Reader's Comments form is provided at the back of this publication. If the form has been removed, you can send comments by fax to 1-800-524-1519 (USA only) or 1-303-924-6873; by E-mail to `printpub@us.ibm.com`; or by mail to:

IBM Printing Systems Division
Department H7FE Building 004N
Information Development
PO Box 1900
Boulder CO 80301-9191 USA

IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Preface

This book describes the functions and services associated with the MO:DCA™ architecture.

This book is a reference, not a tutorial. It complements individual product publications, but does not describe product implementations of the architecture.

Who Should Read This Book

This book is for systems programmers and other developers who need such information to develop or adapt a product or program to interoperate with other presentation products in an IBM® mainframe or workstation environment.

How to Use This Book

This book is divided into seven chapters, five appendixes, and a glossary.

- "Chapter 1, "A Presentation Architecture Perspective"" introduces the IBM presentation architectures and positions the MO:DCA architecture as a strategic presentation data stream architecture.
- "Chapter 2, "Introduction to the MO:DCA Architecture"" introduces the concepts that form the basis of the MO:DCA architecture.
- "Chapter 3, "MO:DCA Overview"" provides an overview of MO:DCA data structures and their use.
- "Chapter 4, "MO:DCA-P Objects"" provides the structure definitions for MO:DCA-P objects.
- "Chapter 5, "MO:DCA Structured Fields"" provides the syntax and semantics for MO:DCA structured fields.
- "Chapter 6, "MO:DCA Triplets"" provides the syntax and semantics for MO:DCA triplet data structures.
- "Chapter 7, "MO:DCA Interchange Sets"" provides complete descriptions of the MO:DCA interchange sets and describes how products can become valid generators and receivers of the MO:DCA architecture.
- "Appendix A, "Color Resources"" provides information on color resources and on color to grayscale conversion.
- "Appendix B, "Resource Access Table (RAT)"" defines the Resource Access Table, which is used to locate and process resources such as TrueType and OpenType fonts.
- "Appendix C, "MO:DCA Migration Functions"" provides the syntax and semantics for MO:DCA migration structured fields, triplets, parameters, and provides the structure definitions for MO:DCA migration objects.
- "Appendix D, "MO:DCA Registry"" provides a registry for object type identifiers, media type identifiers, and color profile identifiers.
- "Appendix E, "Cross-References"" provides tables of MO:DCA structured fields and triplets sorted by identifier and by name.
- The "Glossary" defines some of the terms used within this book.

How to Read the Syntax Diagrams

Throughout this book, syntax is described using the following formats. The syntax of the structured field, the principal MO:DCA data structure, is shown with a horizontal representation, followed by a table that lists the data elements contained in the structured field. The syntax of the triplet, the secondary MO:DCA data structure, is shown using the table only. Six basic data types are used in the syntax descriptions:

CODE	Architected constant
CHAR	Character string, which may consist of any code points
BITS	Bit string
UBIN	Unsigned binary
SBIN	Signed binary
UNDF	Undefined type

Structured Field Introducer

Structured Field Introducer				
SF Length (2B)	ID = **X'D3TTCC'**	Flags (1B)	Reserved X'0000'	Structured Field Data

The meanings of the elements of the horizontal representation are as follows:

- The Structured Field Introducer, which identifies the length and the function or type of the structured field, is composed of the following parameters:

Element	**Meaning**
SF Length	The total length of the structured field including the length of the SF Length element.
ID = X'D3TTCC'	The structured field identifier—consisting of the structured field class, type, and category codes—that uniquely identifies each MO:DCA structured field.
Flags	The set of bits or flags that identify if the structured field is segmented of if a structured field extender or padding is to be used.

- The Structured Field Data, which provides the structured field's effect, is contained in the set of parameters described in the table.

For a detailed discussion of the data elements comprising MO:DCA structured fields, see "MO:DCA Structured Field Syntax" on page 20.

Data

The syntax for a MO:DCA data structure is as follows:

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
The field's byte offset.	The field's data type.	Name of field, if applicable.	Range of valid values, if applicable.	Meaning or purpose of the data element.	M or O	Code

A blank entry in the Range column indicates that there are no restrictions on the acceptable values.

Certain fields may be denoted in the Meaning column as *reserved*. A reserved field is a parameter that has no functional definition at the current time, but may have at some time in the future. All bytes in any field that the MO:DCA architecture defines as a reserved field should be given a value of zero by generating applications. Receiving applications should ignore any values contained in a reserved field.

Additional columns appear to the right of the Meaning column. These columns are:

M/O Mandatory or optional

Exc Exception code for the exception conditions that are possible for the data element. See "Exception Conditions" on page 63 for further information concerning exception conditions.

The following is an example of the MO:DCA 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	Exc
0–7	CHAR	OvlyName		Name of the overlay resource.	M	X'06'
8–10	SBIN	XolOset	0–32767	X axis origin for the page overlay	M	X'06'
11–13	SBIN	YolOset	0–32767	Y axis origin for the page overlay	M	X'06'
14–15	CODE	OvlyOrent	X'0000', X'2D00', X'5A00', X'8700'	The overlay's X axis rotation from the X axis of the including page coordinate system: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	O	X'02'
16–*n*		Triplets		See "IPO Semantics" on page 194 for triplet applicability.	O	X'10'

Related Publications

Several other publications may help you understand the licensed programs used with the data streams described in this book.

IBM Architecture Publications

Table 1. IBM Architecture Publications

Title	Order Number
Bar Code Object Content Architecture Reference	S544-3766
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

Table 2. 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: Data Stream Reference	S544-3202
Advanced Function Presentation: Host Font Data Stream Reference	S544-3289
Advanced Function Presentation: Printer Information. Contains detailed characteristics about IBM's page printers.	G544-3290
Technical Reference for IBM Expanded Core Fonts	S544-5228
Technical Reference for Code Pages	S544-3802
Font Summary for AFP Font Collection	S544-5633
IBM Technical Reference for AFP Font Collection Japanese Fonts	S544-5685
IBM Technical Reference for AFP Font Collection Korean Fonts	S544-5686
IBM Technical Reference for AFP Font Collection Simplified Chinese Fonts	S544-5687
IBM Technical Reference for AFP Font Collection Traditional Chinese Fonts	S544-5688
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

Table 2. IBM Advanced Function Presentation Publications (continued)

Title	Order Number
Advanced Function Presentation Workbench for Windows: Using the Viewer Application. Contains information about using it with AFP API.	G544-3813
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 Content Manager Image Plus Publications

Table 3. IBM Content Manager Image Plus Publications

Title	Order Number
IBM ImagePlus Online Library CD-ROM	SK2T-2131
ImagePlus MVS/ESA™ General Information Manual	GC31-7537
ImagePlus VisualInfo™ for AS/400® Application Programming Guide and Reference	SC34-4586
IBM EDMSuite™ OnDemand User's Guide	SC26-9810

IBM Graphics and Image Publications

Table 4. IBM Graphics and Image Publications

Title	Order Number
GDDM, 5748-XXH: General Information Manual. Contains a comprehensive overview of graphics and image support for MVS™,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

Table 5. Print Services Facility Publications

Title	Order Number
Print Services Facility/MVS: Application Programming Guide	S544-3673
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

Infoprint Manager Publications

Table 6. Infoprint Manager Publications

Title	**Order Number**
Infoprint Manager for AIX Publications (CD-ROM)	SK2T-9266
Infoprint Manager for Windows Publications (CD-ROM)	SK2T-9288

Contents

Figures

Tables

Chapter 1. A Presentation Architecture Perspective

This chapter gives a brief overview of Presentation Architecture.

The Presentation Environment

Figure 1 shows today’s presentation environment.

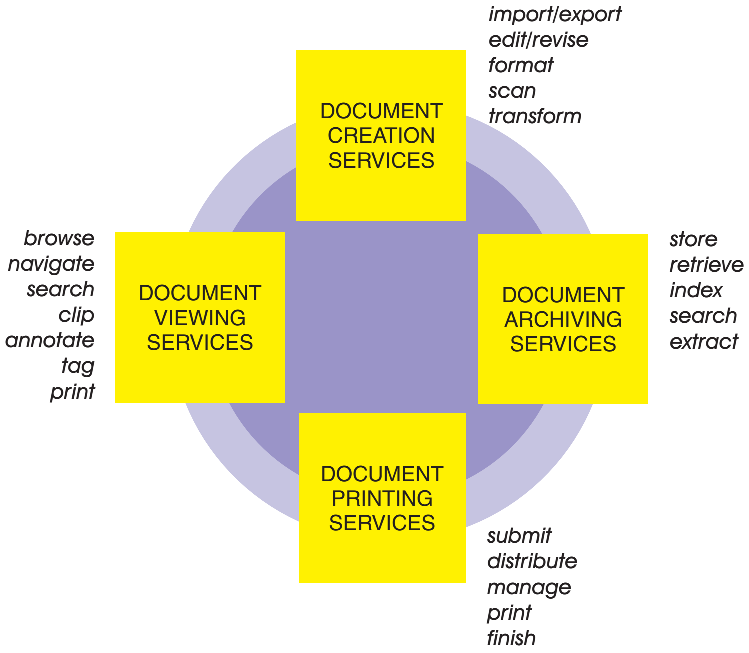


Figure 1. Presentation Environment. The environment is a coordinated set of services architected to meet the presentation needs of today’s applications.

The ability to create, store, retrieve, view and print data in presentation formats friendly to people is a key requirement in almost every application of computers and information processing. This requirement is becoming increasingly difficult to meet because of the number of applications, servers, and devices that must interoperate to satisfy today’s presentation needs.

The solution is a presentation architecture base that is both robust and open-ended, and easily adapted to accommodate the growing needs of the open system environment. IBM presentation architectures provide that base by defining interchange formats for data streams and objects that enable applications, services, and devices to communicate with one another to perform presentation functions. These presentation functions may be part of an integrated system solution or they may be totally separated from one another in time and space. IBM presentation architectures provide structures that support object-oriented models and client/server environments.

IBM presentation architectures define interchange formats that are system independent and are independent of any particular format used for physically transmitting or storing data. Where appropriate, IBM presentation architectures use industry and international standards, such as the ITU-TSS (formerly known as CCITT) facsimile standards for compressed image data.

Architecture Components

IBM presentation architectures provide the means for representing documents in a data format that is independent of the methods used to capture or create them. Documents may contain combinations of text, image, graphics and bar code objects in device- and resolution-independent formats. Documents may contain fonts, overlays and other resource objects required at presentation time to present the data properly. Finally, documents may contain resource objects, such as a document index and tagging elements supporting the search and navigation of document data, for a variety of application purposes.

In IBM, the presentation architecture components are divided into two major categories: *data streams* and *objects*.

Data Streams

A *data stream* is a continuous ordered stream of data elements and objects conforming to a given format. Application programs can generate data streams destined for a presentation service, archive library, presentation device or another application program. The strategic presentation data stream architectures are:

- *Mixed Object Document Content Architecture™ (MO:DCA)*
- *Intelligent Printer Data Stream™ (IPDS™) Architecture*

The MO:DCA architecture 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 database, then later retrieved, viewed, annotated and printed in local or distributed systems environments. Presentation fidelity is accommodated by including resource objects in the documents that reference them.

The IPDS architecture defines the data stream used by print server programs and device drivers to manage all-points-addressable page printing on a full spectrum of devices from low-end workstation and local area network-attached (LAN-attached) printers to high-speed, high-volume page printers for production jobs, shared printing, and mailroom applications. The same object content architectures carried in a MO:DCA data stream can be carried in an IPDS data stream to be interpreted and presented by microcode executing in printer hardware. The IPDS architecture defines bidirectional command protocols for query, resource management, and error recovery. The IPDS architecture also provides interfaces for document finishing operations provided by preprocessing and postprocessing devices attached to IPDS printers.

Other IBM data streams which use many of the presentation objects and concepts introduced in this chapter are:

- The *3270 Data Stream* used to transmit display data between applications and a nonprogrammable workstation
- The *Revisable-Form-Text Document Content Architecture (RFT:DCA)* used to interchange revisable-form text and non-text objects between application programs in an office environment.

Figure 2 on page 3 shows a system model relating MO:DCA and IPDS data streams to the presentation environment previously described. Also shown in the model are the object content architectures which apply to all levels of presentation processing in a system.

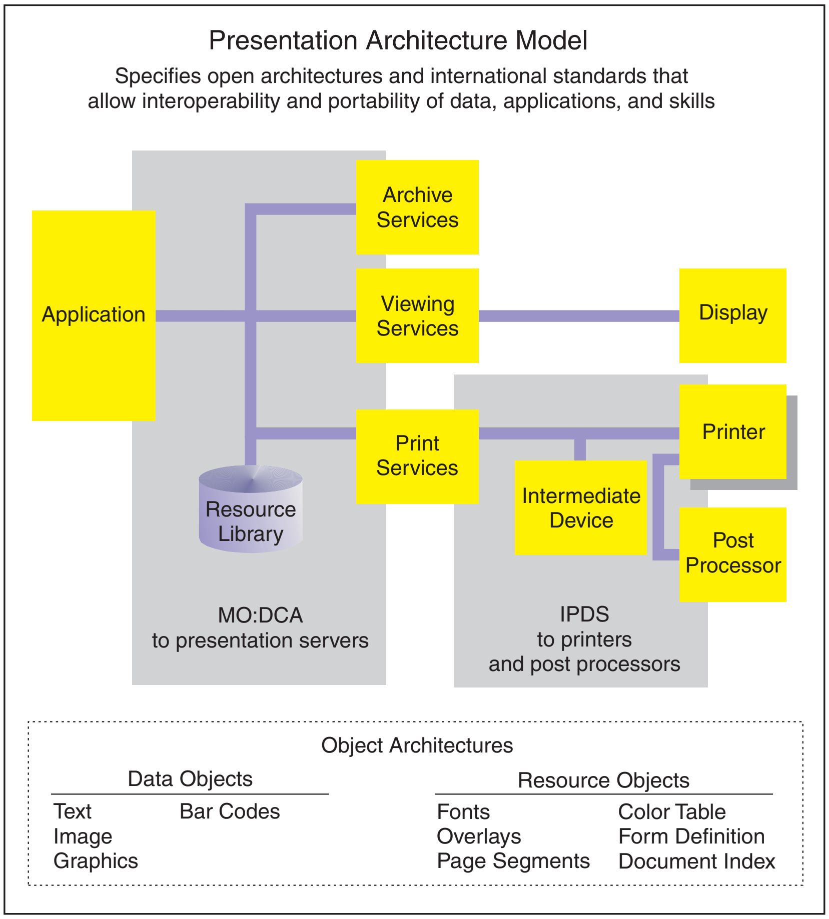


Figure 2. Presentation Model. This diagram shows the major components in a presentation system and their use of data stream and object architectures.

Objects

Documents can be made up of different kinds of data, such as text, graphic, image, and bar code. *Object content architectures* describe the structure and content of each type of data format that can exist in a document or appear in a data stream. Objects can be either *data objects* or *resource objects*.

A data object contains a single type of presentation data, that is, presentation text, vector graphics, raster image, or bar codes, and all of the controls required to present the data.

A resource object is a collection of presentation instructions and data. These objects are referenced by name in the presentation data stream and can be stored in system libraries so that multiple applications and the print server can use them.

All object content architectures (OCAs) are totally self-describing and independently defined. When multiple objects are composed on a page, they exist as peer objects, which can be individually positioned and manipulated to meet the needs of the presentation application.

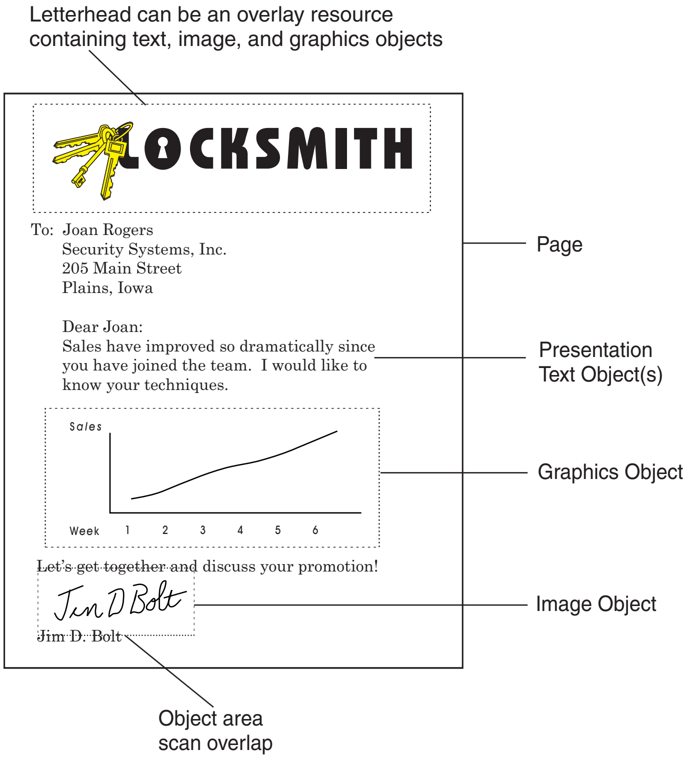
The IBM object content architectures are:

- *Presentation Text Object Content Architecture (PTOCA):* A data architecture for describing text objects that have been formatted for all-points-addressable presentations. Specifications of fonts, text color, and other visual attributes are included in the architecture definition.
- *Image Object Content Architecture (IOCA):* A data architecture for describing resolution-independent image objects captured from a number of different sources. Specifications of recording formats, data compression, color and gray-scale encoding are included in the architecture definition.
- *Graphics Object Content Architecture (GOCA):* A data architecture for describing vector graphic picture objects and line art drawings for a variety of applications. Specification of drawing primitives, such as lines, arcs, areas, and their visual attributes, are included in the architecture definition.
- *Graphics Object Content Architecture for Advanced Function Presentation™ (AFP GOCA):* A version of GOCA that is used in Advanced Function Presentation (AFP) environments.
- *Bar Code Object Content Architecture™ (BCOCA™):* A data architecture for describing bar code objects, using a number of different symbologies. Specification of the data to be encoded and the symbology attributes to be used are included in the architecture definition.
- *Font Object Content Architecture (FOCA):* A resource architecture for describing the structure and content of fonts referenced by presentation data objects in the document.

The MO:DCA architecture also supports data objects that are not defined by IBM object content architectures. Examples of such objects are Tag Image File Format (TIFF), Encapsulated PostScript (EPS), and Portable Document Format (PDF). Such objects may be carried in a MO:DCA envelope called an *object container*, or they may be referenced without being enveloped in MO:DCA structures.

In addition to supporting data objects, the MO:DCA architecture defines envelope architectures for other objects of common value in the presentation environment. Examples of these are *Form Definition* resource objects for managing the production of pages on the physical media, *overlay* resource objects that accommodate electronic storage of forms data, and *index* resource objects that support indexing and tagging of pages in a document.

Figure 3 on page 5 shows an example of an all-points-addressable page composed of multiple presentation objects.


Letterhead can be an overlay resource containing text, image, and graphics objects
LOCKSMITH
Page
Presentation Text Object(s)
Graphics Object
Sales
Week
1
2
3
4
5
6
Image Object
Jim D Bolt
Object area scan overlap

To: Joan Rogers
Security Systems, Inc.
205 Main Street
Plains, Iowa

Dear Joan:
Sales have improved so dramatically since you have joined the team. I would like to know your techniques.

Let's get together and discuss your promotion!

Jim D. Bolt

Figure 3. Presentation Page. This is an example of a mixed-object page that can be composed in a device-independent MO:DCA format and printed on an IPDS printer.

Relationship to Systems Application Architecture

Implementations of the data stream and object content architectures originally developed as part of Systems Application Architecture® Common Communications Support (SAA® CCS) now extend to other major application platforms, such as AIX® and Microsoft® Windows®. This is part of a continuous movement toward providing greater interoperability between presentation components in client/server and open systems environments.

Application-enabling Products

Some of the major application enabling products and application services using presentation interchange architectures are summarized below.

- *Advanced Function Presentation (AFP)*

  A set of licensed programs that use all-points-addressable concepts to present data on a wide variety of printer and display devices. AFP includes creating, formatting, viewing, retrieving, printing, and distributing information.

- *AFP Conversion and Indexing Facility (ACIF)*

  An AFP program for converting a line-data print file into a MO:DCA document and for indexing the document for later retrieval, viewing and selective printing of pages.

- *AFP Toolbox*

  AFP Toolbox provides application programmers with ease of use in formatting printed output. Without requiring knowledge of the AFP data stream, the AFP Toolbox provides access to sophisticated AFP functions through a callable C, C++, or COBOL interface. It is available on MVS, AIX, OS/2®, and AS/400 platforms.

With IBM AFP Toolbox you can:

- Combine variable data with electronic forms, electronic signatures, and images
- Define variable length paragraphs
- Precisely position and align text anywhere on a page using a wide variety of fonts
- Draw fixed or variable depth and width boxes
- Generate barcode objects
- Draw horizontal and vertical fixed or variable length lines
- Include indexing tags for use in efficient viewing and archival/retrieval
- Accent printed output with color and shading
- Dynamically control fonts, including user-defined fonts

- *AFP Workbench*

  A platform for the integration of AFP workstation enabling applications and services. The Viewer application is a Workbench application that runs under OS/2, WIN-OS/2®, or Microsoft Windows.

- *Advanced Function Printing™ Utilities/400*

  An IBM licensed program that includes a group of utilities that work together to provide Advanced Function Printing on AS/400.

- *Content Manager ImagePlus® for OS/390® and Content Manager for AS/400*

  A set of IBM licensed programs that are designed to work in conjunction with the ImagePlus Workstation Program to provide host support for Folder Applications and WorkFlow Management. Documents in the MO:DCA Interchange format are supported.

- *Content Manager OnDemand*

  An IBM licensed program that provides document capture, indexing, archive, retrieval and presentation services. Documents in the MO:DCA Interchange format are supported.

- *Graphical Data Display Manager (GDDM®)*

  An IBM licensed program containing utilities for creating, saving, editing, and displaying visual data such as page segments, charts, images, vector graphics, composites (text, graphics, image), and scanned data.

- *ImagePlus Workstation Program*

  An IBM licensed program designed to capture, view, annotate, print and manipulate text and image documents on a Windows 95, Windows 2000, Windows NT®, or OS/2 platform. Documents can be generated in the MO:DCA interchange format and can be transmitted to OS/390 and AS/400 hosts for folder management and archival storage by other Content Manager components.

- *Infoprint® Manager for AIX and Windows*.

  A print server that drives IPDS page printers. In addition to managing printer resources and providing error recovery for print jobs, Infoprint Manager provides data stream conversions to MO:DCA format for interoperability with other AFP products on AIX and other system platforms.

- *OS/2 Presentation Manager GPI*

  An extensive graphics programming interface (GPI) provided in OS/2 for creating, saving, editing and manipulating picture data composed of graphics primitives, such as lines, arcs, and areas with fill patterns. Metafiles created using the GPI can be archived for later retrieval in the MO:DCA-L interchange format.

- *Print Services Facility™ (PSF)*

The IBM software product that drives IPDS printers. PSF is supported under MVS, VSE, and VM and as a standard part of the operating system under OS/400®. PSF manages printer resources such as fonts and electronic forms, and provides error recovery for print jobs. Multiple data streams are accepted by PSF and are converted into an IPDS data stream for printing.

- *Print Services Facility/2 (PSF/2)*

  An OS/2-based print server that drives IPDS page printers and IBM PPDS and HP-PCL compatible printers. PSF/2 manages printer resources and provides error recovery for print jobs. PSF/2 supports distributed printing of MO:DCA print jobs from PSF/MVS, PSF/VM, PSF/VSE, and OS/400. It also supports printing from a wide range of workstation applications, including Microsoft Windows and the OS/2 Presentation Manager.

For more information on these and other products, see the publications listed in "Related Publications" on page vi.

Application-enabling Products

Chapter 2. Introduction to the MO:DCA Architecture

This chapter:
- Provides a definition of the MO:DCA architecture
- Describes the MO:DCA document component hierarchy

What is the Mixed Object Document Content Architecture?

A mixed object document is the collection of data objects that comprise the document's content, and the resources and formatting specifications that dictate the processing functions to be performed on the content. The term *Mixed* in the MO:DCA architecture refers both to the mixture of data objects and the mixture of document constructs that comprise the document's components. A MO:DCA document can contain a mixture of presentation data objects. Each data object type has unique processing requirements. An Object Content Architecture (OCA) has been established for each IBM data object to define its respective syntax and semantics. MO:DCA documents can contain data and data objects governed by the following OCAs:

- Bar Code Object Content Architecture (BCOCA), which is used to describe and generate bar code symbols.
- Font Object Content Architecture (FOCA), which is used to support the digital presentation of character shapes by defining their attributes, such as shape definitions, shape dimensions, and positioning information. Unlike the other OCAs, font objects are not carried inside the MO:DCA data stream. However, the MO:DCA architecture does provide and carry references to external font objects.
- Graphics Object Content Architecture (GOCA), which is used to represent pictures generated by a computer, commonly referred to as computer graphics.
- Image Object Content Architecture (IOCA), which is used to represent image information such as scanned pictures.
- Presentation Text Object Content Architecture (PTOCA), which is used to define text information.

MO:DCA documents can also contain or reference non-OCA data object types that are registered in the MO:DCA architecture. Such data object types may be carried in a generic MO:DCA object envelope called an *object container*. Examples of non-OCA data object types that can be included in MO:DCA documents are TIFF (Tag Image File Format), EPS (Encapsulated Postscript), and single-page PDF (Portable Document Format).

The MO:DCA architecture is designed to facilitate document *interchange* as well as document *exchange*. Interchange is the predictable interpretation of shared information in an environment where the characteristics of each process *need not be known* to all other processes. Exchange is 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.

The MO:DCA architecture is designed to integrate the different data object types into documents that can be interchanged as a single data stream. The MO:DCA architecture provides the data-stream structures needed to carry the data objects. It also provides syntactic and semantic rules governing their use to ensure that different applications process them in a consistent manner. Figure 4 illustrates the relationship of MO:DCA data structures to a presentation document composed of pages and data objects.

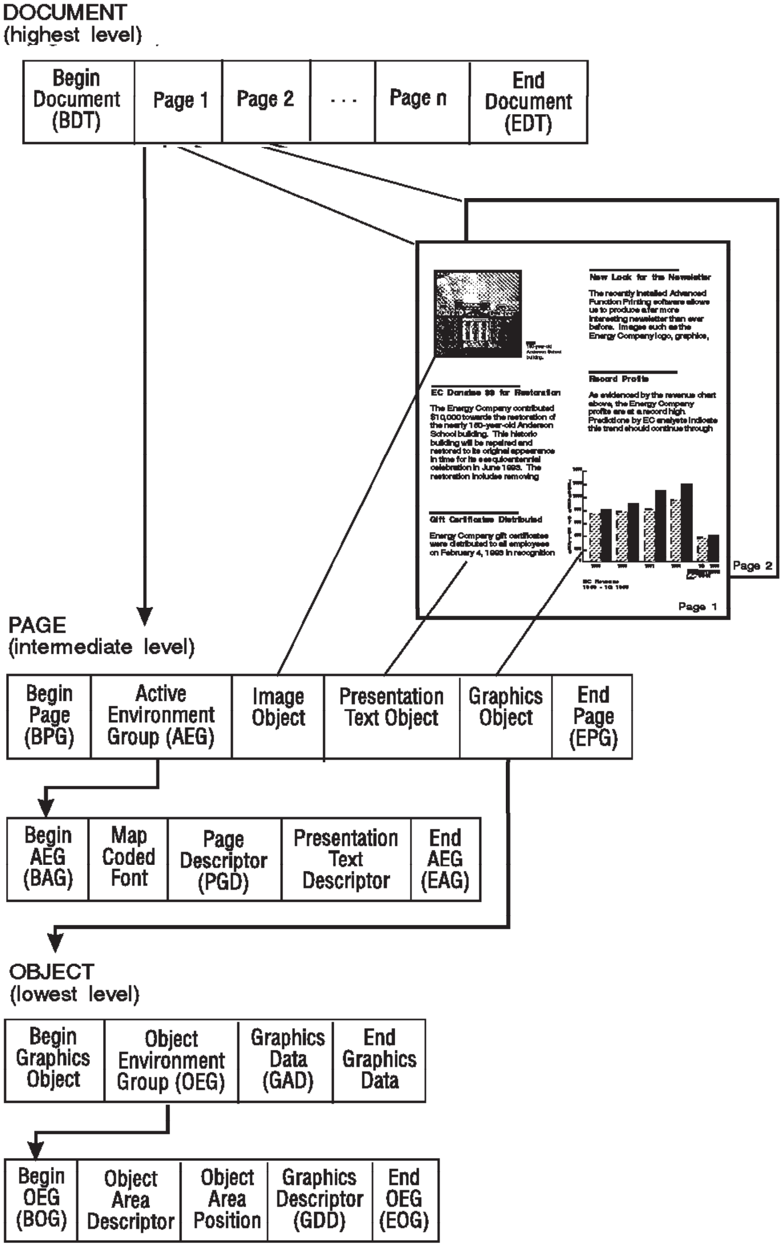


Figure 4. MO:DCA Presentation Document (MO:DCA-P) Components

In its simplest form, a MO:DCA document contains only data objects without any document composition structure. This form is called a MO:DCA resource (MO:DCA-L) document. In its most complex form, a MO:DCA document contains data objects along with data structures that define the document's layout and composition features. This form is called a MO:DCA presentation (MO:DCA-P) document.

MO:DCA components are defined with a syntax that consists of self-describing structures. Structured fields are the main MO:DCA structures and are used to encode MO:DCA commands. A structured field starts with an introducer that uniquely identifies the command, provides a total length for the command, and specifies additional control information such as whether padding bytes are present. The introducer is followed by up to 32,759 data bytes. Data may be encoded using fixed parameters, repeating groups, keywords, and triplets. Fixed parameters have a meaning only in the context of the structure that includes them. Repeating groups are used to specify a grouping of parameters that can appear multiple times. Keywords are self-identifying parameters that consist of a one-byte unique keyword identifier followed by a one-byte keyword value. Triplets are self-identifying parameters that contain a one-byte length, a one-byte unique triplet identifier, and up to 252 data bytes. Keywords and triplets have the same semantics wherever they are used. Together, these structures define a syntax for MO:DCA data streams that provides for orderly parsing and flexible extensibility.

Organization of the Architecture

The MO:DCA definition in this document is organized into three parts:
- Definition of the general architecture
- Definition of MO:DCA interchange sets
- Definition of MO:DCA migration functions

The general architecture is defined in Chapters Chapter 1 through Chapter 6. This includes the general architecture definition for structured fields in Chapter 5, “MO:DCA Structured Fields,” on page 103, the general architecture for triplets in Chapter 6, “MO:DCA Triplets,” on page 313, and the general architecture for MO:DCA-P object structure in Chapter 4, “MO:DCA-P Objects,” on page 67. The general architecture for MO:DCA-L object structure is defined by the MO:DCA-L interchange set.

MO:DCA interchange sets are defined in Chapter 7, “MO:DCA Interchange Sets,” on page 425. Interchange sets consist of structured field, triplet, and object structure specifications that are formal subsets of the general architecture. The purpose of interchange sets is to provide concise, complete document definitions with clear compliance rules that are agreed on and implemented by MO:DCA generators and receivers. It is strongly recommended that MO:DCA support includes compliance with an interchange set.

MO:DCA migration objects, structured fields, triplets, parameters, and rules for processing these structures are defined in Appendix C, “MO:DCA Migration Functions,” on page 501. These constructs may appear in MO:DCA data streams, but they are not considered to be part of the formal architecture definition and may not be supported by all MO:DCA products.

Compliance with the Architecture

MO:DCA-compliant products do not necessarily support all of the structures and functions defined in this document. MO:DCA compliance may be based on document interchange, in which case a product must support one of the defined interchange sets in accordance with the rules specified in “Interchange Set Compliance Requirements” on page 426. MO:DCA compliance may also be based on document exchange, in which case a product must support a subset of the general architecture and must define, in its product documentation, which MO:DCA structures and functions are supported.

Application Note: The MO:DCA structures and functions that are supported by AFP print servers are specified in the *Advanced Function Presentation: Programming Guide and Line Data Reference*.

MO:DCA Concepts

The document is the highest level of the MO:DCA data-stream document component hierarchy. Documents can be made up of pages, and the pages, which are at the intermediate level, can be made up of objects. Objects are at the lowest level, and can be bar codes, graphics, images, and presentation text. The MO:DCA document component hierarchy for a document containing image, graphics, and presentation text objects is illustrated in Figure 4 on page 10. Multiple documents can be grouped together into a printfile.

At each level of the hierarchy certain sets of MO:DCA data structures, called *structured fields*, are permissible. The document, pages and objects are bounded by structured fields that define their beginnings and their ends. These structured fields, called *begin-end pairs*, provide an *envelope* for the data-stream components. This feature enables a processor of the data stream that is not fully compliant with the architecture to bypass those objects that are beyond its scope, and to process the data stream to the best of its abilities.

Documents

MO:DCA documents can belong to either of two categories: presentation documents or resource documents.

- A presentation document (MO:DCA-P document) is one that has been formatted and is intended for presentation, usually on a printer or display device. A data stream containing a presentation document should produce the same document content in the same format on different printers or display devices dependent, however, on the capabilities of each of the printers or display devices. A presentation document can reference resources that are to be included as part of the document to be presented.
- A resource document (MO:DCA-L document) is a collection of resource objects and data objects that can be stored in a library for later retrieval and use.

Pages

Pages contain the data objects that comprise a presentation document. Figure 4 on page 10 portrays the location of the page within the data-stream hierarchy. Each page has a set of data objects associated with it. Each page within a document is independent from any other page, and each must establish its own environment parameters.

The page is the level in the document component hierarchy that is used for printing or displaying a document's content. The data objects contained in the page envelope in the data stream are presented when the page is presented. Each data object has layout information associated with it that directs the placement and orientation of the data on the page. In addition, each page contains layout information that specifies the measurement units, page width, and page depth.

The presentation of a document by a presentation device is a process that consists of presenting the document's pages on a physical medium in accordance with the document's layout and formatting specifications. Examples of physical media are sides of a sheet of paper and display screens.

Overlays

Overlays are page-like resource objects that contain data objects and that define their own environment parameters. Overlays can be included directly on the medium presentation space using a keyword on the Medium Modification Control (MMC) structured field. Such overlays are positioned at the origin of the medium presentation space and are called *medium overlays*. Overlays may also be included on the logical page presentation space using the Include Page Overlay (IPO) and Page Modification Control (PMC) structured fields. Such overlays are positioned at an offset from the logical page origin that is defined by the IPO and PMC and are called *page overlays*. Page overlays that are included with a PMC are also referred to as *PMC overlays*. Note that the MMC and PMC are specified in a *Medium Map* print control object, whereas the IPO is specified directly in the data stream.

Page Segments

Page segments are resource objects that contain data objects but that do not define their own environment parameters. Page segments can be included on the logical page presentation space or on the overlay presentation space using the Include Page Segment (IPS) structured field, and inherit the environment parameters defined by the including page or overlay.

Objects

Objects contain the data that is to be presented. They also may contain environment information needed to establish the proper location and orientation for the data on the presentation surface. Objects in the data stream are bounded by delimiters that identify their type, such as graphics, image, or text. The MO:DCA architecture supports two categories of objects: data objects and resource objects.

Data Objects

In general, data objects consist of data to be presented and the directives required to present it. The content of each type of data object is defined by an object architecture that specifies presentation functions that can be used within its MO:DCA coordinate space. All data objects function as equals in the MO:DCA data-stream environment. Data objects are carried as separate entities with no dependencies on the MO:DCA layout structures or on the containing data-stream environment.

The *object area* is the space on a page that is used to present the data object. An object area is defined by layout information, such as the object area's origin, width, depth, and orientation on the page.

Resource Objects

Resource objects are named objects or named collections of objects that can be referenced from within the document. In general, the referenced resources can reside in a resource group or an external library and can be referenced repeatedly. They may be used in numerous places in a document or in several documents. They are characterized by an unchanging and often complex composition. It is inefficient, and thus undesirable, for applications to generate these objects each time they are required. Instead, the inclusion of these objects in a resource group or a library enables applications to retrieve them repeatedly as they are needed to obtain the desired presentation effect. Examples of resource objects are:

- Fonts that support presentation text and graphics objects
- Referenced data objects
- Page overlays that contain corporate logos, copyright notices, or other such material
- Color attribute tables

- Page segments
- Color mapping tables

Secondary Resource Objects

A data object that is processed as a resource may itself require additional resources for presentation. Such resources are called *secondary resources*. Examples of data objects and their secondary resources are:

- An IOCA FS45 image object that references a tile resource
- A single-page PDF object that requires a custom font
- An EPS object that is to be rendered with a SWOP or Euroscale color profile

A secondary resource may be referenced explicitly from a data object, such as a IOCA tile resource; or it may be tied implicitly to the data object, such as a color profile. A secondary resource must be mapped with an MDR that carries the *external* identifier of the resource in an FQN type X'DE' triplet. This identifier is used to identify the secondary resource in the data stream and in the presentation system. If there is also an explicit reference to the secondary resource from within the data object, the *internal* identifier is specified in an FQN type X'BE' triplet. The FQN type X'DE' and FQN type X'BE' triplets are paired at object include time (when the Include Object structured field that includes the data object is processed) to match up the internal and external identifiers.

Resource Object Mapping

The MO:DCA architecture defines Map structured fields for objects that are to be processed as resource objects. Examples are the Map Page Overlay (MPO), Map Page Segment (MPS), Map Coded Font (MCF), and Map Data Resource (MDR) structured fields. Map structured fields are specified in environment groups and indicate to the presentation server that the referenced object is to be processed as a resource object and will be required for presentation. They may also provide additional information, such as a mapping of the resource reference to a local identifier for the resource. The scope of the environment determines the scope of the mapping. For example, if a resource is mapped in the Active Environment Group (AEG) for a page, the scope of the mapping is the page. Any object that is to be treated as a resource must be mapped. For some objects like page segments, IOCA objects, and non-OCA data objects, treating the object as a resource is optional. Therefore for these objects, the mapping is optional. If a mapping is specified, the object is sent to the presentation device and may be used multiple times via an include command. In that case, the object is sometimes called a *hard* object. If a mapping is not specified, the object is sent to the presentation device as part of the page or overlay, and is sometimes called a *soft* object.

Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

Preloading and Preprocessing Resource Objects

The Resource Environment Group (REG) allows *preloading* of complex resources before printing for the first page is started. This can avoid device underruns that might occur if the resource downloading takes place between pages.

Resource *preprocessing* is an extension of the resource preload concept. It can be used with objects that have a long rasterization time, and causes this rasterization to be done after the resource is preloaded, but before printing of the first page is started. This can avoid device underruns that might occur if such rasterization takes place between pages. Examples of resource objects that might benefit from resource preprocessing are:

- Large IOCA FS45 image objects that need to be rotated, scaled, or trimmed
- Complex EPS and PDF objects

The penalty for underrunning a device is dependent on the device characteristics. For example, on a cut-sheet printer the penalty is normally a larger time delay between page printing, which may be acceptable. On a continuous-forms printer that can backhitch and recover from an underrun, the penalty is normally a loss of throughput and possibly increased printer maintenance. On a continuous-forms printer that cannot backhitch, the penalty is most severe in that unwanted blank sheets are generated during the underrun. These blank sheets must be accounted for and discarded by the post-processing system.

Resource preloading and preprocessing does come at a cost. The undesirable effect of resource preloading and preprocessing is that the time to first print is increased. To achieve optimum throughput, an application should be tuned to preload and preprocess only those resources whose downloading and processing between pages would cause an unacceptable device underrun.

Object Containers

An *object container* is an envelope for object data. The object data may or may not be defined by an IBM presentation architecture. The container consists of a mandatory Begin/End structured field pair, an optional Object Environment Group (OEG), and mandatory Object Container Data (OCD) structured fields. The Begin structured field specifies information about the object data such as object-type identifier, class, type, level, and structure, so that a MO:DCA receiver can determine whether it is an object that can be processed given its capabilities. The OCD structured fields are used to carry the object data.

If the object is to be carried in MO:DCA resource groups and interchanged, it must at a minimum be enveloped by a Begin/End pair, the Object Classification (X'10') triplet on the Begin structured field must specify the registered object-type identifier (object-type OID) for the object data format, and all object data must be partitioned into OCDs. If the object container is to appear directly in a page or overlay, the container must be structured in accordance with the MO:DCA–P syntax for data objects supported directly in pages and overlays, such as IOCA, GOCA, and BCOCA objects. For a definition of this structure, see “Object Containers” on page 100. Object containers can be included indirectly by name in a document using the Include Object (IOB) structured field.

If the object data is traditional time-invariant presentation data, it must be paginated, that is the presentation space within which the object data is presented must be restricted to a single page. However, the object data within the container is not constrained to be traditional presentation object data. Examples of presentation object data that can be carried in an object container are image objects in TIFF (Tag Image File Format), PCX (Paintbrush Picture Format), and DIB (Device Independent Bitmap) format. Examples of non-presentation object data that can be carried in an object container are COM Set-up Files and Color Mapping Tables.

Environment Groups

An *environment group* in the data stream is used to carry layout information and to identify mappings to resources for resource management. Environment groups can be specified at the object, page, or document level. An environment group consists of a set of MO:DCA structured fields enveloped in a begin-end pair.

Document Environment Groups

A *Document Environment Group* may be associated with a Form Map print control object. The document environment group contains presentation specifications such as resource mappings and medium information that apply to all Medium Maps in the Form Map. The scope of a document environment group is the scope of its containing Form Map.

Resource Environment Groups

A *Resource Environment Group* (REG) is associated with a document or a group of pages in a document. It is contained in the document's begin-end envelope in the data stream. 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 is the pages that follow, up to the next REG (which is a complete replacement for the current REG) or the end of the document, whichever occurs first. Specification of a REG is optional. Identifying a resource in a REG does not remove the need to map that resource in the environment groups for the pages and objects that use the resource.

Active Environment Groups

An *Active Environment Group* (AEG) is associated with each page, and is contained in the page's begin-end envelope in the data stream. Figure 4 on page 10 depicts the relationship of the active environment group to the page. The active environment group contains layout and formatting information that defines the measurement units and size of the page, and may contain resource information. Any objects that are required for page presentation and that are to be treated as resource objects must be mapped with a map structured field in the AEG. The scope of an active environment group is the scope of its containing page or overlay. In many cases the information contained in an active environment group can be inherited by objects contained in the page. See "Default Values" on page 31 for a discussion of defaults and inheritance.

Object Environment Groups

An *Object Environment Group* (OEG) may be associated with an object and is contained within the object's begin-end envelope. Figure 4 on page 10 depicts the relationship of the object environment group to its corresponding object. The object environment group defines the object's origin and orientation on the page, and can contain resource information.

Any objects that are required for object presentation and that are to be treated as resource objects must be mapped using a map structured field in the OEG.

Application Note: For PSF resource management, any mapping specified in the OEG for an object must also be specified in the AEG for the page or overlay that includes the object. This is sometimes referred to as *factoring* the resource mapping from OEG to AEG.

The scope of an object environment group is the scope of its containing object. An application that creates a data-stream document may omit some of the parameters normally contained in the object environment group, or it may specify that one or more default values are to be used. The values to be used may be:

- Inherited from the active environment group on the current page
- Supplied by default values defined by the MO:DCA architecture
- Supplied by default values defined by the application

See "Default Values" on page 31 for a discussion of defaults and inheritance.

Resource Groups

A *resource group* is used in the data stream to contain resources during transmission. The resources can be referenced or included at other locations within the data stream. A document can consist entirely of resource groups and can be used to pass any type of resource between products without any document composition overhead.

Resource groups can exist at the print file level, the document level, the page level, and the data object level. A resource group has the same scope as its container. That is, the contents of the resource group are available for referencing until the containing component is ended. For example, when a resource group is contained within a page, the contents of the resource group are available for referencing only within that page. Once the End Page structured field is encountered, the resources contained within that resource group are no longer available.

Although the MO:DCA architecture has several ways of referencing a resource object, ultimately they all result in matching a referenced identifier with the identifier used for the resource object. If the resource object is within a resource group in the data stream, the resource object's identifier is specified on the Begin structured field that defines the object. If the resource is in an external library, the resource object's identifier is the library name associated with the object. The MO:DCA architecture does not require that the library name be the same as the identifier specified on the Begin structured field.

In addition to matching the identifier, the resource object type must also match the reference. Thus, if a reference is made to a page overlay named ABCDEF and a color attribute table named ABCDEF is encountered in the resource group, it is not considered a valid match because the Begin structured field is of the wrong type.

Although the MO:DCA architecture permits objects of different resource types to have the same identifier, it requires that objects of the same resource type *within the same resource group* have unique identifiers. However, there is no restriction on having multiple objects of the same resource type and identifier in multiple resource groups.

The MO:DCA architecture defines the order in which resource groups must be searched when attempting to locate a specific resource. When searching for a resource, the first resource located that satisfies the search criteria ends the search. Thus, although two different versions of the same resource type with the same name may exist in different resource groups, the MO:DCA resource scope and search rules remove any uncertainty as to which of the resources will be selected.

When the reference is from within a data object, the MO:DCA search order is:

1. The current data object level resource group, if one exists
2. The current page level resource group, if one exists
3. The document level resource group, if one exists
4. The print file level resource group, if one exists

When the reference is from outside a data object, the MO:DCA search order is:

1. The current page level resource group, if one exists

2. The document level resource group, if one exists
3. The print file level resource group, if one exists

If no resource groups exist or if the referenced resource object is not found in any of the resource groups searched, the search is extended to an external library. The search convention does not include library access methods, since these are dependent upon the implementing system. For the formal definition of resource groups in MO:DCA-P data streams, see "Resource Groups" on page 78.

Page Groups

A *page group* is used in the data stream to define a named, logical grouping of sequential pages. Page groups are delimited by begin-end structured fields that carry the name of the page group. Page groups are defined so that the pages that comprise the group can be referenced or processed as a single entity. Examples of page group processing are:

- Assigning a set of common indexing attributes to the page group
- Retrieving the page group from an archive system for viewing

Print Control Objects

Print control objects are resource objects that contain formatting, layout, and resource-mapping information used to present the document's pages on physical media. Print control objects may be selected at the time of the print request, or they may be invoked directly from the document. There are two types of print control objects, *form maps*, also known as *form definitions* or *formdefs*, and *medium maps*. A form map print control object contains one or more medium map print control objects. Note that a medium map is also sometimes referred to as a *copygroup*.

Process Elements

Process elements are document structures that facilitate particular forms of document processing. A process element is defined by a structured field and does not contain any presentation specifications, that is, it does not affect the appearance of a document when the document is presented. An example of a process element is a Tag Logical Element (TLE), which specifies object attribute information that can be used to support attribute-based document indexing and attribute-based document navigation. Another example is a Link Logical Element (LLE), which specifies a linkage from a source document component to a target document component.

Chapter 3. MO:DCA Overview

This chapter:
- Describes the general syntax and semantics for MO:DCA structured fields
- Describes state, as defined by the MO:DCA architecture
- Describes the types and categories of MO:DCA parameters
- Describes conventions used in the MO:DCA architecture for coordinate systems, measurement units, and rotation units
- Describes MO:DCA mixing rules
- Describes font technologies used in MO:DCA documents
- Describes MO:DCA document indexing
- Describes other aspects of MO:DCA document presentation
- Describes and defines the MO:DCA exception conditions

MO:DCA Data Structures

Each component of a mixed object document is explicitly defined and delimited in the data stream that transmits it. This is accomplished through the use of MO:DCA data structures, called *structured fields*, that reside in the data stream. Structured fields are used to envelop document components and to provide commands and information to applications using the data stream. Structured fields may contain one or more parameters. Each parameter provides one value from a set of values defined by the architecture.

Notation Conventions

In addition to the information provided in “How to Read the Syntax Diagrams” on page iv, the following notation conventions apply throughout this document:

- Bytes are numbered from left to right beginning with byte zero, which is considered the high order (most significant) byte position. This is referred to as *big-endian* byte order. For example, a three-byte field would consist of byte zero, byte one, and byte two.
- Each byte is composed of eight bits.
- Bits in a single byte are numbered from left to right beginning with bit zero, the most significant bit, and continuing through bit seven, the least significant bit. This is referred to as big-endian bit order.
- When bits from multiple consecutive bytes are considered together, the first byte always contains bits zero to seven and the bits of the additional bytes are numbered eight to *n*, where *n* is equal to one less than the total number of bytes multiplied by eight. For example, a two-byte field would consist of bits zero to fifteen and a four-byte field would consist of bits zero to thirty-one.
- Negative numbers are expressed in two’s-complement form. See “Number” on page 34 for details.
- Field values are expressed in hexadecimal or binary notation:

```
B'01111110' = X'7E' = +126
X'7FFF' = +32767
X'8000'  = −32768 (when signed binary is used)
X'8000'  = +32768   (when unsigned binary is used)
```

MO:DCA Structured Field Syntax

MO:DCA structured fields consist of two parts: an introducer that identifies the length and type of the structured field, and data that provides the structured field's effect. The data is contained in a set of parameters, which can consist of other data structures and data elements. The maximum length of a structured field is 32,767 bytes. The general format for a structured field is as follows:

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

Structured Field Introducer

The MO:DCA Structured Field Introducer (SFI) introduces a structured field, and identifies its type and its length. SFIs have the following format:

SFI Syntax

Table 7. Structured Field Introducer (SFI)

Offset	Type	Name	Range	Meaning	M/O	Exc
0–1	UBIN	SFLength	8–32767	Total length of the structured field including the length of the introducer	M	X'82'
2–4	CODE	SFTypeID		A three-byte code that uniquely identifies the structured field. See “SFI Semantics” on page 21 for a description.	M	X'78'
5	BITS	FlagByte		Used to indicate whether an extension, segmentation, or padding is in use	M	X'82'
Bit 0		ExtFlag	B'0', B'1'	**B'0'** No SFI extension exists **B'1'** SFI extension is present		
Bit 1				Reserved; must be zero		
Bit 2		SegFlag	B'0', B'1'	**B'0'** Data is not segmented **B'1'** Data is segmented		
Bit 3				Reserved; must be zero		
Bit 4		PadFlag	B'0', B'1'	**B'0'** No padding data exists **B'1'** Padding data is present		
Bits 5–7				Reserved; must be zero		
6–7				Reserved; should be zero	M	X'82'
The following optional extension appears only if bit 0 of FlagByte is B'1':						
8	UBIN	ExtLength	1–255	Length of the extension including the length of ExtLength itself	O	X'82'

Table 7. Structured Field Introducer (SFI) (continued)

Offset	Type	Name	Range	Meaning	M/O	Exc
9		ExtData		Reserved	O	X'00'

SFI Semantics

SFLength Defines the length of the structured field, including itself.

SFTypeID A three-byte field that uniquely identifies the structured field. It has the form *D3TTCC*, where:

Code **Description**

D3 The structured field *class* code that has been assigned to the MO:DCA architecture.

TT The structured field *type* code. The type code identifies the function of the structured field, such as begin, end, descriptor, or data. See "Type Codes" on page 22 for a description of type codes.

CC The structured field *category* code. It identifies the lowest-level component that can be constructed using the structured field, such as document, active environment group, page, or object. The same category code point assigned to a component's begin structured field also is assigned to that component's end structured field. These code points identify and delimit an entire component within a data stream or an encompassing component. See "Category Codes" on page 23 for a description of category codes.

FlagByte Specifies the value of the optional indicators. Indicator bits are defined as follows:

Bit **Indicator name and meaning**

0 ExtFlag is the SFI extension flag. See "Structured Field Introducer Extension" on page 24 for details.
B'0' No SFI extension exists.
B'1' This structured field has an SFI extension.

2 SegFlag is the segmentation flag. See "Structured Field Segmentation" on page 24 for details.
B'0' No segmentation in effect.
B'1' The data for this structured field has been segmented.

4 PadFlag is the padding flag. See "Structured Field Padding" on page 24 for details.
B'0' No padding data appended.
B'1' Padding data has been appended to the end of this structured field.

All others
Reserved; must be binary zero

| **Bytes 6–7** Reserved; should be zero

Application Note: In AFP environments, some applications use bytes 6–7 of the Structured Field Introducer to specify a sequence number for the structured field. This is an unarchitected use of these bytes and should be avoided.

ExtLength Specifies the length of the SFI extension, including the length of ExtLength itself. For ExtLength to be valid, bit 0 of FlagByte must be B'1'.

ExtData Contains up to 254 bytes of application-defined SFI extension data. For ExtData to be valid, bit 0 of FlagByte must be B'1'.

Type Codes

The following type codes have been defined. All other type codes are reserved.

Table 8. Type Codes

Type Code	**Function**	**Description**
X'A0'	Attribute	An *attribute* structured field defines an attribute with parameters such as name and value.
X'A2'	Copy Count	A *copy count* structured field specifies groups of sheet copies, called *copy subgroups*, that are to be generated, and identifies modification control structured fields that specify modifications to be applied to each group.
X'A6'	Descriptor	A *descriptor* structured field defines the initial characteristics and, optionally, the formatting directives for all objects, object areas, and pages. Depending on the specific descriptor structured field type, it may contain some set of parameters that identify: • The size of the page or object • Measurement units • Initial presentation conditions
X'A7'	Control	A *control* structured field specifies the type of modifications that are to be applied to a group of sheet copies, or a copy subgroup.
X'A8'	Begin	A *begin* structured field introduces and identifies a document component. In general, a begin structured field may contain a parameter that identifies the name of the component.
X'A9'	End	An *end* structured field identifies the end of a document component. In general, an end structured field may contain a parameter that identifies the name of the component.
X'AB'	Map	A *map* structured field provides the following functions in the MO:DCA architecture: • All occurrences of a variable embedded in structured field parameter data can be given a new value by changing only one reference in the mapping, rather than having to physically change each occurrence. Thus all references to font X may cause a Times Roman font to be used in one instance and a Helvetica font in another instance merely by specifying the proper *map coded font* structured field. • The presence of the map structured field in a MO:DCA environment group indicates use of the named resource within the scope of the environment group.
X'AC'	Position	A *position* structured field specifies the coordinate offset value and orientation for presentation spaces.
X'AD'	Process	A *process* structured field specifies processing to be performed on an object.

Table 8. Type Codes (continued)

Type Code	Function	Description
X'AF'	Include	An *include* structured field selects a named resource which is to be embedded in the including data stream as if it appeared *inline*. External resource object names on the *begin* structured field may or may not coincide with the library name of that object, as library name resolution is outside the scope of the MO:DCA architecture.
X'B0'	Table	A *table* structured field contains a list of items of the same or similar type that are related to one another.
X'B1'	Migration	A *migration* structured field is used to distinguish the MO:DCA structured field from a structured field with the same acronym from an earlier data-stream architecture. The earlier version is called *Format 1*. The MO:DCA version is called *Format 2*.
X'B2'	Variable	A *variable* structured field defines or contains variable information.
X'B4'	Link	A *link* structured field defines a logical connection, or linkage, between two document components.
X'EE'	Data	A *data* structured field consists of data whose meaning and interpretation is governed by the object architecture for the particular data object type.

Category Codes

The following category codes have been defined. All other category codes are reserved.

Category Code	Description
X'5F'	Page Segment
X'6B'	Object Area
X'77'	Color Attribute Table
X'7B'	IM Image
X'88'	Medium
X'8A'	Coded Font
X'90'	Process Element
X'92'	Object Container
X'9B'	Presentation Text
X'A7'	Index
X'A8'	Document
X'AD'	Page Group
X'AF'	Page
X'BB'	Graphics
X'C3'	Data Resource
X'C4'	Document Environment Group (DEG)
X'C6'	Resource Group
X'C7'	Object Environment Group (OEG)
X'C9'	Active Environment Group (AEG)
X'CC'	Medium Map
X'CD'	Form Map
X'CE'	Name Resource
X'D8'	Page Overlay
X'D9'	Resource Environment Group (REG)
X'DF'	Overlay
X'EA'	Data Suppression
X'EB'	Bar Code

X'EE'	No Operation
X'FB'	Image

Structured Field Data

The structured field's data is contained in a parameter set that immediately follows the structured field's introducer. The syntax and semantics for each MO:DCA structured field parameter set is given in Chapter 5, "MO:DCA Structured Fields," on page 103. Depending on the structured field and its purpose, the parameter set may contain zero or more parameters. If parameters are present, they contain specific information appropriate for the structured field. The data occupies as many bytes as needed, up to a maximum of 32,759 bytes.

Structured Field Introducer Extension

A structured field introducer may be extended by up to 255 bytes. The presence of an SFI extension is indicated by a value of B'1' in bit 0 of the SFI flag byte. If an extension is present, the introducer is at least 8 bytes, but not more than 263 bytes, in length. The first byte of the extension specifies its length. If an extension to the structured field introducer is present, the structured field's data can occupy up to 32,759 bytes, less the length of the extension.

Structured Field Segmentation

When the total length of the introducer and the data portion of a structured field exceeds 32,767 bytes, the data must be split into two or more fragments and specified on multiple consecutive structured fields. This is known as *segmenting* a structured field. Segmenting normally only occurs for those structured fields that contain OCA data.

When a structured field is segmented, the OCA may require that the data be split on specific data element boundaries. The MO:DCA architecture permits other structured fields to be interspersed between the segmented structured fields. However, for those cases where it is undesirable to split the data at a specific boundary or to permit other structured fields to appear between the segmented structured fields, the MO:DCA architecture provides a segmentation flag. This flag indicates that the segmented structured fields are all part of a single, uninterrupted parameter string. When bit 2 of the SFI flag byte is set to B'1', the parameter data may be split at any byte boundary and *no* other structured fields are permitted to appear between the segmented structured fields. The segmentation flag value for the last structured field in a sequence of structured fields containing a segmented parameter string must be B'0'.

Structured Field Padding

Padding bytes may be used by an application to extend the physical length of a structured field beyond what is required by its introducer and parameter set. This could be done, for example, to make all structured fields the same length or to make each structured field's length a multiple of some number. The use of padding is indicated by a value of B'1' in bit 4 of the SFI flag byte.

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 specified in the last three bytes of the padding data. The last byte must be 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.

When padding is indicated:
- The structured field length value specifies the total length of the structured field, including the padding data.
- The padding length value specifies the total length of the padding data, including the padding length byte(s).

Structured Field Formats

The MO:DCA architecture has evolved from several previous IBM data streams, namely the Composed Page Data Stream (CPDS), the Composite Document Presentation Data Stream (CDPDS), and the Advanced Function Print Data Stream (AFPDS). Because of this, the MO:DCA architecture uses many of the same structured fields originally defined for these architectures. However, in some cases new structured fields have been defined that have the same name, acronym, and usage as these older structured fields. This has only been done for those cases where it became necessary to expand the function of the structured field, but the definition of the original structured field did not lend itself to expansion.

These new structured fields are always assigned a structured field identifier closely resembling the old one. Although the structured field identifiers clearly differentiate between the two versions of the same structured field, when referring to them by name or by acronym, the older version is known as *Format 1* and the newer MO:DCA version is know as *Format 2*. Two such structured fields are the Map Coded Font structured field and the Presentation Text Data Descriptor structured field.

Data Stream Format

The MO:DCA architecture does not dictate the physical format of the data stream or how it is transported. The data stream may be carried within a communication protocol or it may be stored on a tape or disk. It may be one continuous string of bytes or it may be broken up into multiple records. When broken into multiple records, the records may be fixed length or variable length. Each record may contain an individual structured field, a portion of a structured field, or any number of contiguous structured fields. The receiver must be capable of receiving the data stream and parsing or processing it sequentially from start to finish. While receivers may impose reasonable limits on blocking factors for buffer management purposes, they should not be designed to process only one type of data stream format.

MO:DCA Data Stream States

The MO:DCA architecture defines a *state* to be a domain within the data stream, bounded by a begin-end structured field pair, within which certain structured fields are permitted. The processor of a MO:DCA data stream is required to check the validity of the structured field sequence received. A valid structured field sequence is one in which each structured field that is processed belongs to the set of permissible structured fields for the begin-end envelope in which it is found. If a structured field other than one belonging to the set of permissible structured fields is detected, a violation of the state has occurred, and the processor is required to raise an exception condition.

Data Stream States

The MO:DCA architecture recognizes the following states:

State **Description**

Document
Initiated by a Begin Document structured field and terminated by an End Document structured field. The Begin Document structured field defines the beginning of the MO:DCA data stream, within which all other MO:DCA document-level structured fields are contained.

Index Initiated by a Begin Document Index structured field and terminated by an End Document Index structured field. Structured fields that define a document index may be encountered in the index state.

Resource Group
Initiated by a Begin Resource Group structured field and terminated by an End Resource Group structured field. Structured fields that define resources, such as page overlays and color tables, may be encountered in the resource group state.

Named Resource
Initiated by a Begin Resource structured field and terminated by an End Resource structured field. Structured fields that define resources may be encountered in the named resource state.

Resource Environment Group
Initiated by a Begin Resource Environment Group structured field and terminated by an End Resource Environment Group structured field. Structured fields that identify resources for presentation may be encountered in the resource environment group state.

Page Group
Initiated by a Begin Named Page Group structured field and terminated by an End Named Page Group structured field. Structured fields that define pages, or that define other nested page groups, or that specify attributes of the page group may be encountered in page group state.

Page Initiated by a Begin Page structured field and terminated by an End Page structured field. Structured fields that define objects and active environment groups or that specify attributes of the page may be encountered in page state.

Active Environment Group
Initiated by a Begin Active Environment Group structured field and terminated by an End Active Environment Group structured field. Structured fields that provide environment specifications affecting a page and objects within a page may be encountered in the active environment group state.

Data Object
Initiated by a begin object structured field for bar code, graphics, image, or presentation text, and terminated by a corresponding end object structured field. Structured fields that define object environment groups and contain object data may be encountered in the data object state.

Resource Object
Initiated by a begin resource object structured field for resources such as color attribute tables, medium maps, and page overlays, and terminated by a corresponding end resource object structured field. Structured fields that define the contents of resource objects may be encountered in the resource object state.

Object Container
Initiated by a Begin Object Container structured field and terminated by an End Object Container structured field. Structured fields that define object environment groups and contain object data may be encountered in the object container state.

Object Environment Group
Initiated by a Begin Object Environment Group structured field and terminated by an End Object Environment Group structured field. Structured fields that provide environment specifications affecting objects within a page may be encountered in the object environment group state.

State Hierarchies

States are grouped and organized hierarchically. Although individual interchange sets may impose additional restrictions, the general state hierarchy within the MO:DCA architecture is as follows:

- States permitted within Document state:
  - Index
  - Page
  - Page Group
  - Resource Group (MO:DCA-L)
  - Resource Object
  - Resource Environment Group
- States permitted within Index state:
  - None
- States permitted within Resource Group state:
  - Resource Object
  - Named Resource
- States permitted within Named Resource state:
  - Resource Object
- States permitted within Resource Environment Group state:
  - None
- States permitted within Page Group State:
  - Page
  - Page Group
  - Resource Object
  - Resource Environment Group
- States permitted within Page state:
  - Resource Group
  - Active Environment Group
  - Data Object
  - Object Container
- States permitted within Active Environment Group state:
  - None
- States permitted within Data Object state:
  - Resource Group (MO:DCA-L)
  - Object Environment Group
- States permitted within Resource Object state:
  - Active Environment Group if the object is a page overlay
  - Object Environment Group if the object is a data object

  States permitted within Object Container state:
  - Object Environment Group
- States permitted within Object Environment Group state:

- None

See Chapter 4, “MO:DCA-P Objects,” on page 67, “MO:DCA Presentation Interchange Set 1” on page 429, “MO:DCA Presentation Interchange Set 2” on page 444, and “MO:DCA Resource Interchange Set” on page 463 for details of the structured fields that may be encountered in each state in MO:DCA-P, MO:DCA-P IS/1, MO:DCA-P IS/2, and MO:DCA-L data streams respectively.

Environment Hierarchies

The Active Environment Group and Object Environment Group are also hierarchically related. Parameters specified in the OEG *override* like parameters specified in the AEG, while like parameters specified within the same environment—whenever this is allowed—*replace* the previous specification. To illustrate this point, consider the following example. Note that the same LID mapping rules apply when a resource object is mapped with a Map Data Resource (MDR) structured field.

- A page contains an AEG with the following two Map Coded Font structured fields:
  - An MCF that maps LID 1 to font A and LID 2 to font B
  - An MCF that maps LID 3 to font D
- A graphics data object on that same page contains an OEG with the following two Map Coded Font structured fields:
  - An MCF that maps LID 3 to font E and LID 4 to font F
  - An MCF that maps LID 5 to font H

For objects on that page that do not specify their own MCFs within their own OEGs, the LIDs and their associated fonts would be:

- LID 1 = font A, from AEG MCF #1
- LID 2 = font B, from AEG MCF #2
- LID 3 = font D, from AEG MCF #2

The LIDs and their associated fonts available within the graphics object would be:

- LID 1 = font A, from AEG MCF #1
- LID 2 = font B, from AEG MCF #2
- LID 3 = font E, from OEG MCF #1
- LID 4 = font F, from OEG MCF #1
- LID 5 = font H, from OEG MCF #2

In this case, fonts A and B were made available from the MCFs contained in the AEG which was higher in the environment hierarchy. However, font D was overridden when the first MCF in the OEG mapped LID 3 to font E.

Similarly, if a Presentation Space Reset Mixing triplet were specified on both the Page Descriptor structured field and one or more Object Area Descriptor structured fields within a particular overlay within a resource group, the PGD would control the presentation space mixing for the entire overlay presentation space and the OBDs would control the presentation space mixing for their individual object area presentation spaces.

Resource Environment Groups (REGs) are optional and do not affect AEGs and OEGs. Identifying a resource in a REG does not remove the need to map that resource in the environment groups of the pages and objects that use the resource.

Processing Order

Unless otherwise specified in a structured field's definition, all structured fields are processed in the order in which they appear in the data stream. For example, if a presentation data stream contains a page with a text object, an Include Page Overlay, a graphic object, a second Include Page Overlay, and an image object, in that order, the objects are presented (imaged) on the page in that same order. That is, the text object is presented first, the first overlay is presented second, the graphic object is presented third, the second overlay is presented fourth, and the image object is presented last.

Likewise, unless otherwise specified in the structured field or triplet definition, structured field and triplet parameters are also processed in the order in which they appear in the structured field or triplet.

Resource Search Order

Resources used by a MO:DCA document may be located in resource groups that are internal to the document, in resource groups that are external to the document (printfile-level resource groups), or in resource libraries.

The general search order for MO:DCA resources is as follows:
1. Internal resource groups
2. External (printfile-level) resource groups
3. External resource libraries

For the formal definition of resource groups in MO:DCA-P data streams, see "Resource Groups" on page 78.

Structured Field Parameters

A structured field is composed of a set of parameters that provides data and control information to processors of the data stream. The MO:DCA architecture has established a length, a set of permissible values and a functional definition for each structured field parameter.

Mandatory and Optional Parameters

A parameter can be mandatory or optional. Chapter 5, “MO:DCA Structured Fields,” on page 103 provides a description of each structured field’s parameters. The description indicates whether each parameter is mandatory or optional.

Mandatory Parameters

A *mandatory parameter* appears in a structured field because the function of the parameter is required and a value is essential for proper interpretation of the data stream. A value must be specified for a mandatory parameter. The value specified either must be within the range of permissible parameter values, or it must designate that an existing default value is to be used. A mandatory parameter requires that a suitable value for the parameter must appear somewhere in the hierarchy of structured fields in the data stream.

Optional Parameters

An *optional parameter* can be omitted from a structured field if the function of that parameter is not required, or if, although the function is required, a default value is acceptable. An optional parameter cannot be omitted if the function is required and the default value is not acceptable.

Parameter Categories

A parameter’s category refers to its structure. A parameter can consist of a single data element or it can be a data structure composed of several data elements. Parameters that are data structures can have either a fixed length or a variable length. In the MO:DCA architecture two types of parameters are used: *fixed* and *self-identifying*.

Fixed Parameters

A parameter consisting of a single data element is called a *fixed parameter*. A fixed parameter has a constant size in terms of bits and bytes and it always appears at the same location within its structured field. Fixed parameters also are called *positional parameters*.

Self-identifying Parameters

Self-identifying parameters are data structures that consist of three or more data elements, one of which is used to identify the purpose of the parameter. The self-identifying parameter in the MO:DCA architecture is known as a *triplet*.

A triplet can have a variable length of up to 254 bytes. A triplet must consist of at least three data elements: a length data element, an identifier data element, and one or more data elements for its contents. It can occupy any location after any fixed parameters that occur in the structured field.

Repeating Groups

The MO:DCA architecture also supports another category of parameters known as a *repeating group*. A repeating group consists of specific fixed or self-identifying parameters that have been combined into a defined group. This group then becomes a data structure that may be specified multiple times.

When the repeating group contains self-identifying parameters, the first parameter in the repeating group is a length parameter that indicates how many bytes comprise that repeating group. This length parameter is called the RGLength parameter and the value specified always includes the length of the RGLength parameter itself, which is usually two bytes.

When the repeating group contains only fixed parameters, the MO:DCA architecture may or may not specify that the repeating group contains a RGLength parameter. When it does, the value specified for the RGLength parameter always includes the length of the RGLength parameter itself.

Note: Frequently, a structured field may contain both positional and self-identifying parameters. When this occurs, the positional parameters always occur before any self-identifying parameters. At times, some or all of the positional parameters may be defined as optional. Optional parameters may only occur at the end, *after* all mandatory parameters. When optional self-identifying parameters such as triplets are added to a structured field that has optional positional parameters defined, all of the positional parameters are considered *mandatory* and must appear before the first self-identifying parameter. See "Include Page Overlay (IPO)" on page 194 for an example of this type of structured field.

Parameter Values

A parameter's value can be specified directly, or it can be obtained indirectly through the use of defaults.

Specified Values

The values to be given to a parameter must be consistent with its length and data type. Additional constraints on values may eliminate one or more values that otherwise could be assigned to a parameter.

Default Values

The use of defaults enables the sender of data-stream documents to omit the values for defaulted parameters, permitting the receiving application to use predetermined values. A default value can be given to a parameter by omitting any value for it, or by entering a value, defined by the architecture, requesting use of the default. The source of the default value used for a parameter may be an environment group higher in the document component hierarchy, or it may be an architected default established by the MO:DCA architecture.

Hierarchical Defaults: Parameter values established by an environment group at a higher level in the document component hierarchy will be the default for a subordinate level unless a value is specified at the subordinate level. The scope of a parameter is the same as the scope of the structured field that contains it. Thus the parameters established in an active environment group for the current page will be in effect for the duration of the page, and will be the default parameters for all objects contained in the page. If an object in the page has an associated object environment group that specifies new values, the new parameter values will be in effect for the duration of the object. If the parameters for a subsequent object in the page are unspecified, or if they specify that the default value is to be used, the values from the current page's active environment group will be used. The placement of parameter values at a higher level in the document hierarchy, for the purpose of enabling lower levels to *inherit* these values as defaults, is known as *factoring*.

Architected Defaults: Certain parameters may be given default values by the MO:DCA architecture. Parameters that have been given defaults are identified in the structured field descriptions in Chapter 5, "MO:DCA Structured Fields," on page 103. If a default is not listed for a parameter, no architected default exists.

Default Indicator

One of the values that usually can be given to a parameter is the *default indicator*. Use of the default indicator for a parameter's value specifies that the current default value for the parameter is to be used. In the MO:DCA architecture the default indicator has the value X'F...F'. The default indicator specifies that either a hierarchical default value or an architected default value is to be used for the parameter. A default indicator is implied when a fixed parameter has been omitted at the end of a structured field. A fixed parameter cannot be omitted if any subsequent, optional, positional parameter is present, or if any triplet is present.

Any parameter for which the default indicator is valid must have a default value assigned. This value, which must be valid for the parameter, is used when the default indicator is specified or implied. A structured field whose parameter values are all default indicators has no effect and can be omitted from the data stream.

Parameter Occurrence

Parameters may be *single-occurrence* or *multiple-occurrence*. The syntax tables in Chapter 5, "MO:DCA Structured Fields," on page 103 identify which parameters are single-occurrence and which are multiple-occurrence.

Single-Occurrence Parameters

Single-occurrence parameters can occur only once in a structured field. Single-occurrence parameters can be fixed parameters or triplets. If a value is specified for a single-occurrence parameter, it will be in effect for the scope of its structured field. If the value of a single-occurrence parameter is omitted or if the default indicator is specified, then normal default value inheritance will apply.

Multiple-Occurrence Parameters

Multiple-occurrence parameters are parameters that can appear more than once in a structured field. Multiple-occurrence parameters can be triplets or repeating groups. A repeating group may consist of fixed parameters, triplets, or a combination of fixed parameters and triplets. The following rules apply to multiple-occurrence parameters:

- Triplets will not inherit values from higher levels of the document component hierarchy.
  - If some triplets are omitted from a structured field at a lower level, default values will not be used. The result will be that no values will exist for the omitted parameters for the scope of the structured field.
  - If all triplets are omitted from a structured field, architected default values will be used for those parameters that have them. The result will be that only those parameters having architected defaults will have effect for the scope of the structured field.
- Fixed parameters will inherit values from higher levels of the document component hierarchy. If repeating groups of fixed parameters are specified at more than one level within the document component hierarchy and semantic conflicts occur, then the conflicts are resolved in favor of the lowest level for the scope of the structured field.

Parameter Types

The term *parameter type* refers to a parameter's function rather than to the data type of the parameter's data. For a listing of the six basic data types used by the MO:DCA architecture, see "How to Read the Syntax Diagrams" on page iv. A parameter's function may be closely related to a data type, for example, in the case of a bit string parameter and the BITS data type. A MO:DCA parameter may be a bit string, character string, code, global identifier, local identifier, name, number, or an undefined type.

One of the most important functions for certain types of parameters is their use in referencing other document components. A *reference* is the use of an identifier to refer to a component, structured field, or repeating parameter group. References are usually found in structured fields that map component identifiers to local identifiers, and that *invoke* or *include* components at specific data-stream locations. The effect is the same as if the component appeared at the location in the data stream that contains the structured field that invokes or includes it. Components that are referenced by *include* structured fields provide resource definitions or object definitions. Components that are referenced by *invoke* structured fields provide format information, such as that contained in environment groups.

Bit String

A *bit string* is a string of binary elements and corresponds to the BITS data type. Each bit of a bit string has a value of either B'1' or B'0', which represents *on* or *off* respectively. Each bit usually is independent of the others. Some combinations of bits may be invalid depending on what has been defined for the data element by the MO:DCA architecture. The convention used for addressing bits within a bit string is that the leftmost bit is bit 0.

Character String

A *character string* corresponds closely to the CHAR data type. It is used for identifiers composed of a string of one or more graphic characters. Character strings are compared on the basis of the identifiers of the graphic characters that are presented for the corresponding code points. In the MO:DCA data stream, this is governed by the Coded Graphic Character Set Global Identifier (CGCSGID).

Code

A *code* is a value assigned by the MO:DCA architecture that relates to a particular meaning. The code parameter type relates to the CODE data type. In general, parameters having a code type are given hexadecimal values or value ranges to distinguish them from parameters with a number type.

Global Identifier

A *global identifier* (GID) is a string of bytes that is from 1 to 250 bytes in length. It is usually a coded graphic character string with a data type of CHAR, but it can also be a number or a code. A global identifier has either an alphanumeric character value that is a global name, such as the name of a document, or a numeric value that is unique in the interchange environment. If an identifier is to be used where uniqueness is required, for example to reference a component by name, the same name or value cannot be used more than once within the scope of its reference. For example, the same name must not be given to two different resource definitions of the same type in the same resource group.

Local Identifier

A *local identifier* (LID) is used within the data stream to reference a resource, such as a color attribute table or coded font, from within a structured field or an OCA. The application creating the data stream is responsible for establishing the cross references or mapping between the resources and their LIDs. The use of LIDs and mapping enables an application to make one change in the mapping to effect multiple changes for the scope of an LID, rather than having to make a change at each location where the LID appears.

Once established, an LID has meaning only within the context of the data stream that contains it. An LID has a data type of CODE and its meaning is independent of where the data stream is created, filed, transmitted, or presented.

Whenever a local identifier parameter type is used to relate structured fields present in the data stream, the scope of reference for the LID is the begin-end pair enveloping the referenced resource. Thus both the *referenced* resource and the *referencing* structured field must reside in the same begin-end envelope.

Structured fields, known as map structured fields, that specify a global to local mapping follow the normal MO:DCA environment hierarchy rules.

Name

A *name* is an identifier composed of alphanumeric characters, and is closely related to the CHAR data type. A name parameter type can relate either to a global or a local identifier. Names are compared on the basis of the identifiers of the graphic characters that are presented for the corresponding code points. When comparing names of unequal length, the shorter name is padded with space characters until it is the same length as the longer name.

Generally, names of begin structured fields within a MO:DCA data stream are required to be unique only if their names will be referenced and they reside in the same containing envelope with another begin structured field of the same type. For example, the presence of two color tables named `colortbl` in the same resource group would cause an exception condition.

Name parameters for end structured fields, if used, must match the name parameter for corresponding begin structured fields. However if the first two bytes of the name parameter for an end structured field have the value X'FFFF', it will, by default, match any name on the corresponding begin structured field.

A value of X'0...0' for any positional parameter having a name type indicates that a Fully Qualified Name (FQN) triplet exists in the structured field. The Fully Qualified Name triplet contains a name that is used to replace the positional name parameter value.

The scope of any name reference is limited to the scope of the document component where the name is specified. Thus a name appearing in an Active Environment Group has a scope that is limited to the page or page overlay containing the Active Environment Group, and a name appearing in an Object Environment Group has a scope that is limited to the object containing the Object Environment Group.

Number

A *number* or arithmetic value implies count or magnitude. All numbers used within the MO:DCA architecture are either signed or unsigned integers as indicated in the syntax tables by the SBIN and UBIN data types respectively.

In an unsigned number, all bits are used to express the absolute value of the number. For signed numbers, the leftmost, or high order bit represents the sign, which is followed by the integer field.

Positive numbers are represented in true binary notation with the sign bit set to zero. Negative numbers are represented in two's-complement binary notation with the sign bit set to one. Specifically, a negative number is represented by the two's complement of the positive number. The two's-complement of a number is obtained by inverting each bit of the number and adding a one to the low-order bit position.

Since the MO:DCA architecture defines X'F...F' as a default indicator, the arithmetic value −1 generally is not permitted. However, in the case where a parameter cannot be defaulted, the value which normally is the default indicator is interpreted as −1. Chapter 5, "MO:DCA Structured Fields," on page 103 and Chapter 6, "MO:DCA Triplets," on page 313 identify parameters that cannot be defaulted. The maximum absolute values for numbers that can be assigned to data elements that also can be assigned the default indicator are listed in Table 9.

Table 9. Maximum Absolute Values of Numbers in the MO:DCA Architecture

Number of Bytes	Data Type	Absolute Values	
		Hexadecimal	Decimal
1	SBIN	X'7F'	127
1	UBIN	X'FE'	254
2	SBIN	X'7FFF'	32767
2	UBIN	X'FFFE'	65534
3	SBIN	X'7FFFFF'	8,388,607
3	UBIN	X'FFFFFE'	16,777,214
4	SBIN	X'7FFFFFFF'	2,147,483,647
4	UBIN	X'FFFFFFFE'	4,294,967,294

Unique syntax is used for the expression of values that pertain to units of measurement and to rotation. See "Measurement Units" on page 37 and "Rotation Units" on page 42 for details of this syntax.

Coordinate Systems

The MO:DCA architecture defines a multi-level coordinate system hierarchy that allows a large degree of flexibility in presenting data on a physical medium. A MO:DCA coordinate system is an orthogonal coordinate system based on the fourth quadrant of a standard Cartesian coordinate system. Both the X axis and the Y axis specify positive values, which is a difference from the Cartesian system where the Y axis in the fourth quadrant specifies negative values.

Wherever negative offsets are supported, such as in the positioning of a page presentation space on the medium presentation space, the negative X axis is generated by extending the X axis left of the origin, and the negative Y axis is generated by extending the Y axis above the origin. Negative numbers are expressed in two's complement notation.

Each individual coordinate system is associated with a specific presentation space. The MO:DCA architecture defines the following presentation spaces:

Medium Presentation Space
The presentation space for the physical medium. This is the base presentation space onto which all other presentation spaces are merged.

Page Presentation Space
The presentation space for the page, also called a *logical page*.

Overlay Presentation Space
The presentation space for an overlay.

Object Area Presentation Space
The presentation space for an object area.

Data Object Presentation Space
The presentation space for a data object. This presentation space is defined by the corresponding data object architecture. For details on data object presentation spaces, refer to the reference manual for each specific data object architecture.

The coordinate systems that correspond to the MO:DCA presentation spaces are listed in Table 10. Each coordinate system defines its own coordinate axes, measurement units, and extents.

Table 10. MO:DCA Coordinate Systems

Coordinate System	**Notation for Axes**	
	x direction	**y direction**
Medium	X_m	Y_m
Page	X_{pg}	Y_{pg}
Overlay	X_{ol}	Y_{ol}
Object Area	X_{oa}	Y_{oa}

The origin of all MO:DCA coordinate systems is the point (0,0) where X equals zero and Y equals zero. The X and Y axes form the top and left edges, respectively, of the presentation space, as shown in Figure 5 on page 37.

The presentation space associated with the MO:DCA page can be specified to exist on either side of a sheet, and multiple page presentation spaces can exist on the same side of a sheet.

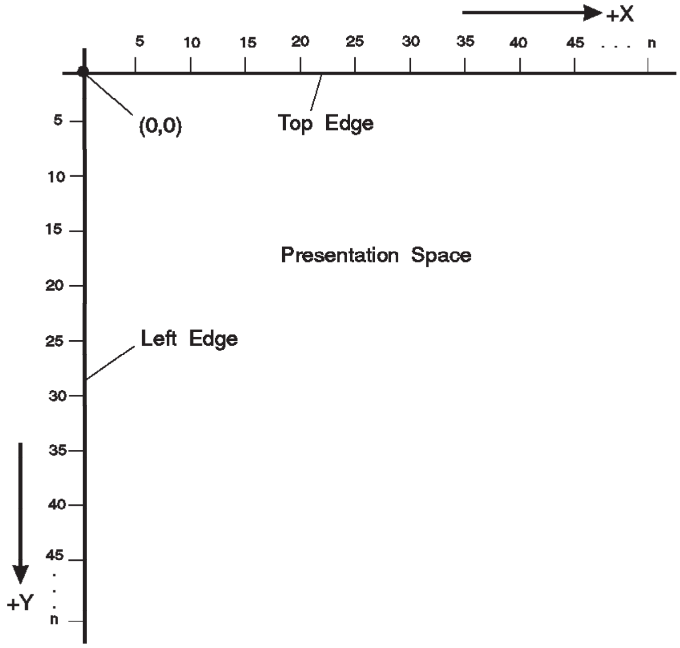


Figure 5. A MO:DCA Presentation Space Coordinate System

Measurement and Rotation

Measurement and rotation conventions are essential to the specification and interpretation of layout information for data-stream documents. MO:DCA's conventions for measurement include data element formats and definitions for units, extent, and position. Its conventions for rotation include data element formats and definitions for units.

Measurement

The distance of a point from an origin is known as its absolute position. The distance of a point from another point is known as its relative position. Distances are measured in *addressable positions*, and can mean X_m,Y_m units, X_{pg},Y_{pg} units, X_{ol},Y_{ol} units, or X_{oa},Y_{oa} units, depending on the extent or offset being measured.

Measurement Units

Measurement units are used throughout the MO:DCA architecture to identify the units of measure to be used for such things as extents and offsets along the X and Y axes of a coordinate system.

Each individual measurement unit is specified as two separate values:

Unit base

This value represents the length of the measurement base. It is specified as a one-byte coded value. The valid codes and their associated meanings are as follows:

X'00'	Ten inches
X'01'	Ten centimeters

Units per unit base

This value represents the number of units in the measurement base. It is specified as a two-byte numeric value between 1 and 32767.

The term *units of measure* is defined as the measurement base value divided by the units per unit base value.

For example, if the measurement base is 10 inches and the units per unit base is 5000, then the units of measure is 10 inches / 5000 or one five-hundredth of an inch.

The base measurement units for each axis is specified as part of the definition of a presentation space. Each MO:DCA coordinate system may specify base measurement units independent from other coordinate systems appearing on the same medium. Although the overall architecture design permits each axis to have a different unit base, current implementations require that both unit bases be identical.

Measurement Unit Formats

The format used to resolve addressable positions into a unit of measure is a set of four parameters that specify the X and Y units of length used for measurements in the X and Y direction, respectively.

Parameter	Description
X unit base	A one-byte code
Y unit base	A one-byte code
X units per unit base	A two-byte binary number from 1 through 32767 in units of the X unit base
Y units per unit base	A two-byte binary number from 1 through 32767 in units of the Y unit base.

Since presentation devices can be built to support different units of measure along different axes, the units of measure to which the presentation spaces have been designed can be specified in the data stream. The target presentation device may determine if it can accept the specified length unit, if it can convert from the specified addressable positions to one of its own, or if it recognizes a problem and possibly rejects that portion of the data stream. The origins of coordinate systems can be established at any addressable position that exists within a presentation space.

Extent

Each presentation space has two *extents*: the X extent, which parallels the X axis as it currently is oriented, and the Y extent, which parallels the Y axis as it currently is oriented. Extents start at the origin of a presentation space and end at a point determined by summing the extent value and the origin value. Negative extent values are not permitted since the area enclosed by a MO:DCA coordinate system always starts at the origin and proceeds in positive X and Y directions within its current orientation. In Figure 6 on page 39 the X extent of the presentation area is represented by line segment *0R* and the Y extent by line segment *0D*.

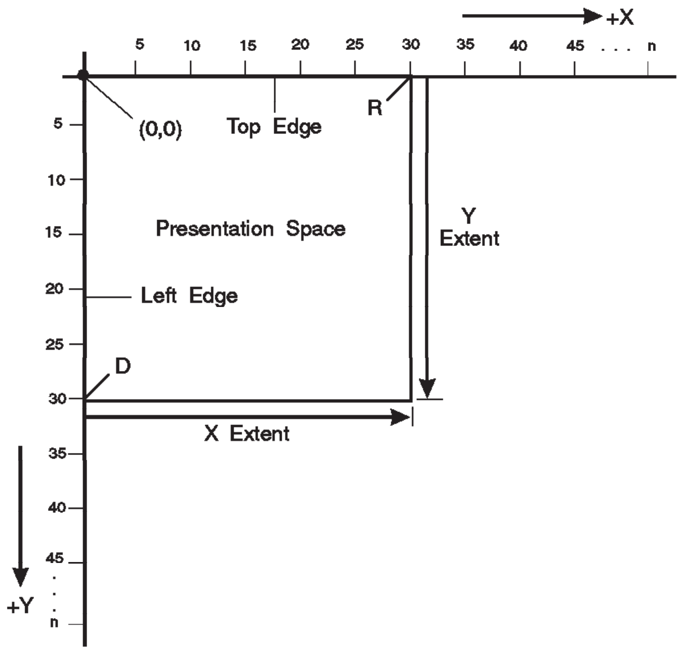


Figure 6. Presentation Space Extents

The bottom edge of a presentation space is a line parallel to the X axis of the presentation space that intercepts the Y axis at the end point of the Y extent. The right edge of a presentation space is a line parallel to the Y axis of the presentation space that intercepts the X axis at the end point of the X extent.

The two extents specify the size of the presentation space. Using the example of a measurement base of 10 inches and a units per unit base of 5000, if the X extent were specified as 4250 and the Y extent as 5500, the presentation space size would be 8.5 by 11 inches.

Offset

The origin of any MO:DCA coordinate system is expressed as an *offset* from the origin of another coordinate system. The offset values for the X and Y axes can be positive or negative. Negative offset values are expressed in two’s complement notation. Any MO:DCA coordinate system that is offset from a reference coordinate system need not be contained within that reference coordinate’s extents.

The medium coordinate system is the base coordinate system from which all the other coordinate systems are directly or indirectly offset. A coordinate system for a document component that is placed within a superior document component *references* the coordinate system of the superior document component. For example, the coordinate system of an object or a page overlay that is placed on a page references the page’s coordinate system. Since each MO:DCA coordinate system can be expressed in different base measurement units, the offset of the origin of a subordinate coordinate system, relative to the origin of the reference coordinate system, is always measured in the reference system’s base measurement units. This permits the reference system to influence the placement of the contained system.

The offset coordinate system inherits the orientation of the reference coordinate system. In Figure 7 on page 40, the origin for coordinate system B is offset ten X units and ten Y units from the reference coordinate system A. Coordinate system

B's origin is specified as the intersection of the lines drawn perpendicular to the X and Y axes at the specified X and Y offset values from coordinate system A.

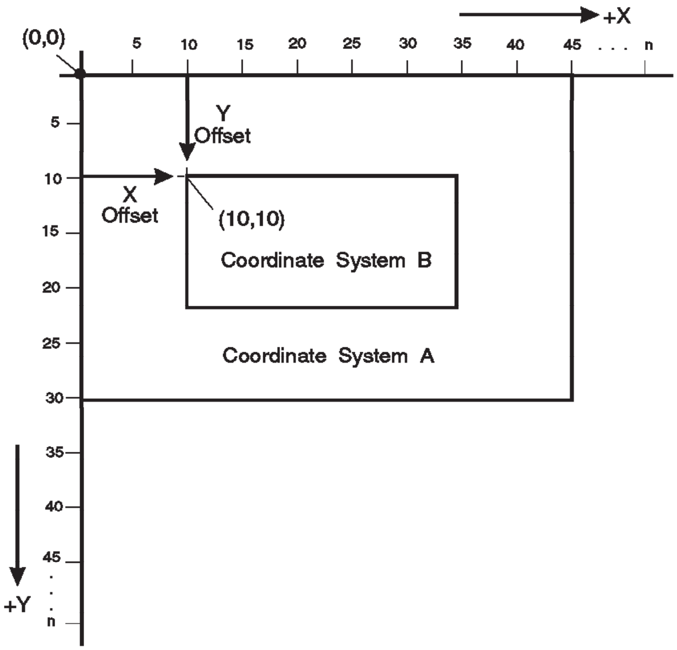


Figure 7. Offset of a Coordinate System

Any portion of a coordinate system may be overlapped by one or more peer coordinate systems. For example, two different object areas could be defined with the same origin so that one completely overlapped the other, or their origins could be specified such that only a portion of the object areas overlapped.

Rotation

Rotation is used to change the presentation orientation of a document component with respect to that of the superior document component that contains it.

Orientation refers to the rotation of a document component and its coordinate system with respect to the coordinate system that contains it. After a MO:DCA coordinate system's origin and X and Y extents have been established, the orientation value of the coordinate definition may cause the defined space to rotate in a clockwise direction around its origin. Orientation is expressed in degrees, with the Y axis orientation value being 90 degrees greater than the X axis orientation value.

Figure 8 on page 41 shows the effect of rotating one coordinate system, shown as a series of rectangles, within a containing coordinate system. Note how the X and Y extents, and thus the rectangle formed by these extents, rotate around the contained coordinate system's origin point of 3 and 4 units from the origin of the containing coordinate system.

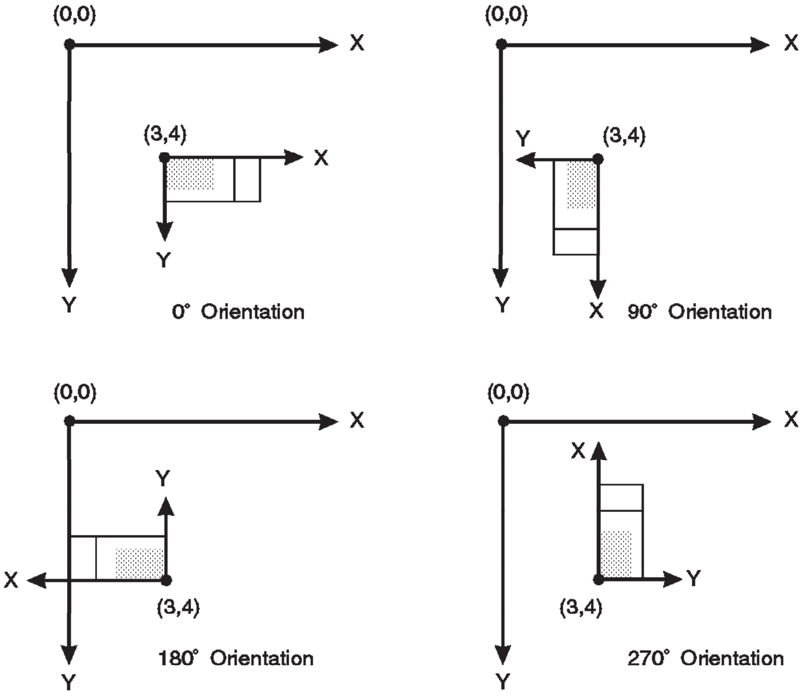


Figure 8. Examples of Coordinate System Orientation

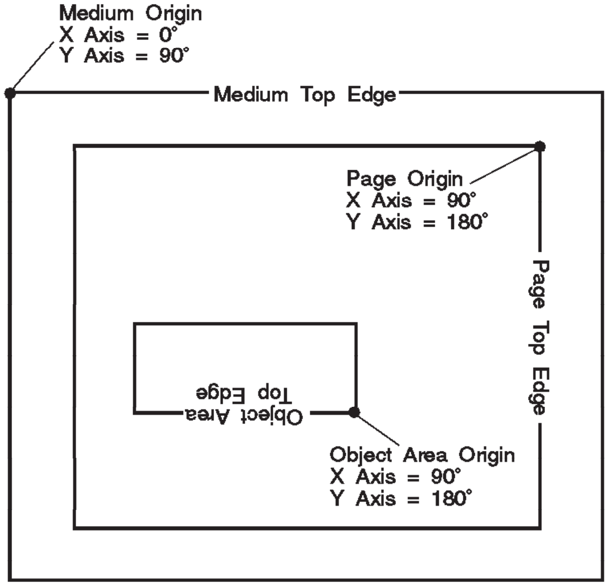


Figure 9. Inheritance of Coordinate System Orientation

The orientation characteristics possessed by a MO:DCA coordinate system do not have to be the same as those of its reference coordinate system. Any MO:DCA coordinate system may possess orientation characteristics that are the same as, or different from, their reference coordinate system or any other MO:DCA coordinate system. Figure 9 shows the effect of offsetting a page from a medium, then rotating it 90 degrees and then offsetting an object area from the page and rotating it 90 degrees. The object area inherited the 90 degree page rotation which, when added to its 90 degrees rotation, produced a cumulative orientation value of 180 degrees.

Rotation Units

The rotation of the X and Y axes of a page overlay or an object area are specified in terms of rotation units. Rotation unit values are expressed in degrees and minutes using two-byte, three-part binary numbers as shown in Table 11.

Table 11. Format for Numbers Expressed in Rotation Units

Bit Position	Name	Meaning
Bit 0–Bit 8	Degrees	Used to represent 0 through 359 degrees. Values from 360 through 511 are invalid.
Bit 9–Bit 14	Minutes	Used to represent 0 through 59 minutes. Values from 60 through 63 are invalid.
Bit 15	Reserved	Value must be zero.

A rotation value of zero, X'0000', specifies no rotation with respect to the X axis of the presentation space in which the origin of the page overlay, object area, or object is located. Increasing values indicate increasing clockwise rotation. The four major orientations, plus-X, plus-Y, minus-X, and minus-Y, have values of 0 degrees, 90 degrees, 180 degrees, 270 degrees respectively. They are encoded as X'0000', X'2D00', X'5A00', and X'8700'. See Figure 10.

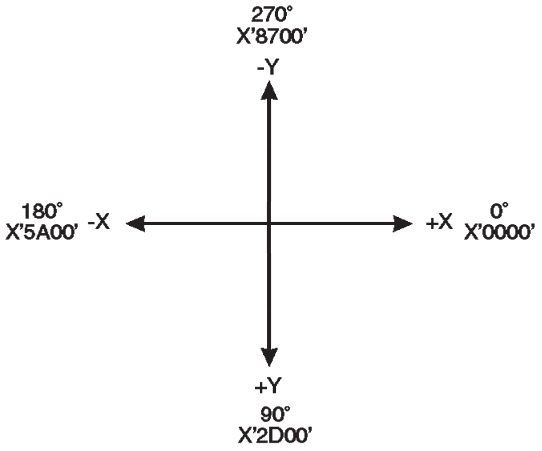


Figure 10. Rotation of the X and Y Axes

Overlays for a page are always positioned relative to the current orientation of the page coordinate system. However, their X and Y extent values remain constant regardless of the orientation. Figure 11 on page 43 shows this graphically.

Shape

The X and Y axes are perpendicular to each other, and the rotation of the Y axis is exactly 90 degrees more than the rotation specified for the X axis. All MO:DCA presentation spaces must be rectangles. The shape of the data object is not defined by the MO:DCA architecture and can take on any visual appearance.

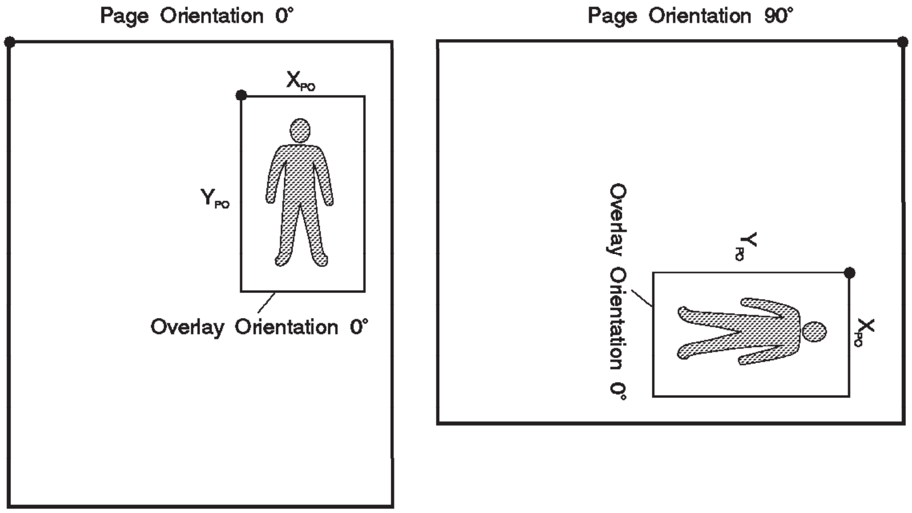


Figure 11. A Page Overlay Applied to a Page in Two Different Orientations

Presentation Space Mixing

Foreground and Background

MO:DCA presentation spaces such as the medium, page, overlay, and data object presentation spaces consist of two parts: foreground and background. Foreground is the part of the presentation space that is occupied with object data. This data can be pure object data such as text, or mixed object data such as image overlaying text. Background is the part of the presentation space that is not occupied with object data. For data object presentation spaces, the data object defines foreground and background, and may specify a color attribute for both. For each data object type, foreground, background, and color attributes are defined by the architecture that defines the object content. For example, in a text presentation space, characters and rules are foreground, everything else is background. Foreground is assigned a color attribute using the "Set Extended Text Color" control sequence. Background cannot be assigned a color and is therefore implicitly assigned the color of the medium. When no color is specified for the background of a presentation space, the background is implicitly assigned the color of the medium. The medium, page, and overlay presentation spaces are initially empty. Empty MO:DCA presentation spaces contain only background, which is assigned the color of the medium.

Table 12 summarizes the definition of foreground and background in IBM OCA-based object presentation spaces:

Table 12. Foreground/Background in Data Object Presentation Spaces

Data Type	Foreground	Background
PTOCA Text	• Stroked and filled portion of text characters • Stroked area of text rules • Stroked area of underscores	Everything else
IM image	B'1' image points	B'0' image points
IOCA bilevel image IOCA bilevel tiled image	Significant image points, except image points for which a transparency mask specifies B'0'	• Insignificant image points • Image points for which a transparency mask specifies B'0' • All portions of the presentation space not covered by image or tiles

Table 12. Foreground/Background in Data Object Presentation Spaces (continued)

Data Type	Foreground	Background
IOCA grayscale or color image	Entire image, except image points for which a transparency mask specifies B'0'	• Image points for which a transparency mask specifies B'0' • All portions of the presentation space not covered by image points
IOCA grayscale or color tiled image	Entire tile, except image points for which a transparency mask specifies B'0'	• Image points for which a transparency mask specifies B'0' • All portions of the presentation space not covered by tiles
GOCA Graphics	• Stroked area of arcs • Stroked area of lines • Stroked and filled portion of pattern symbols • Stroked and filled portion of marker symbols • Stroked and filled portion of graphic characters • B'1' image points • Entire area with solid fill	Everything else
BCOCA Bar Code	• Bars • Stroked and filled portions of HRI characters	Everything else
Colored object area, page, or overlay presentation space	Complete presentation space	None
Empty object area, page, or overlay presentation space	None	Complete presentation space
Non-OCA Presentation Objects	See “Object Type Identifiers” on page 535	See “Object Type Identifiers” on page 535

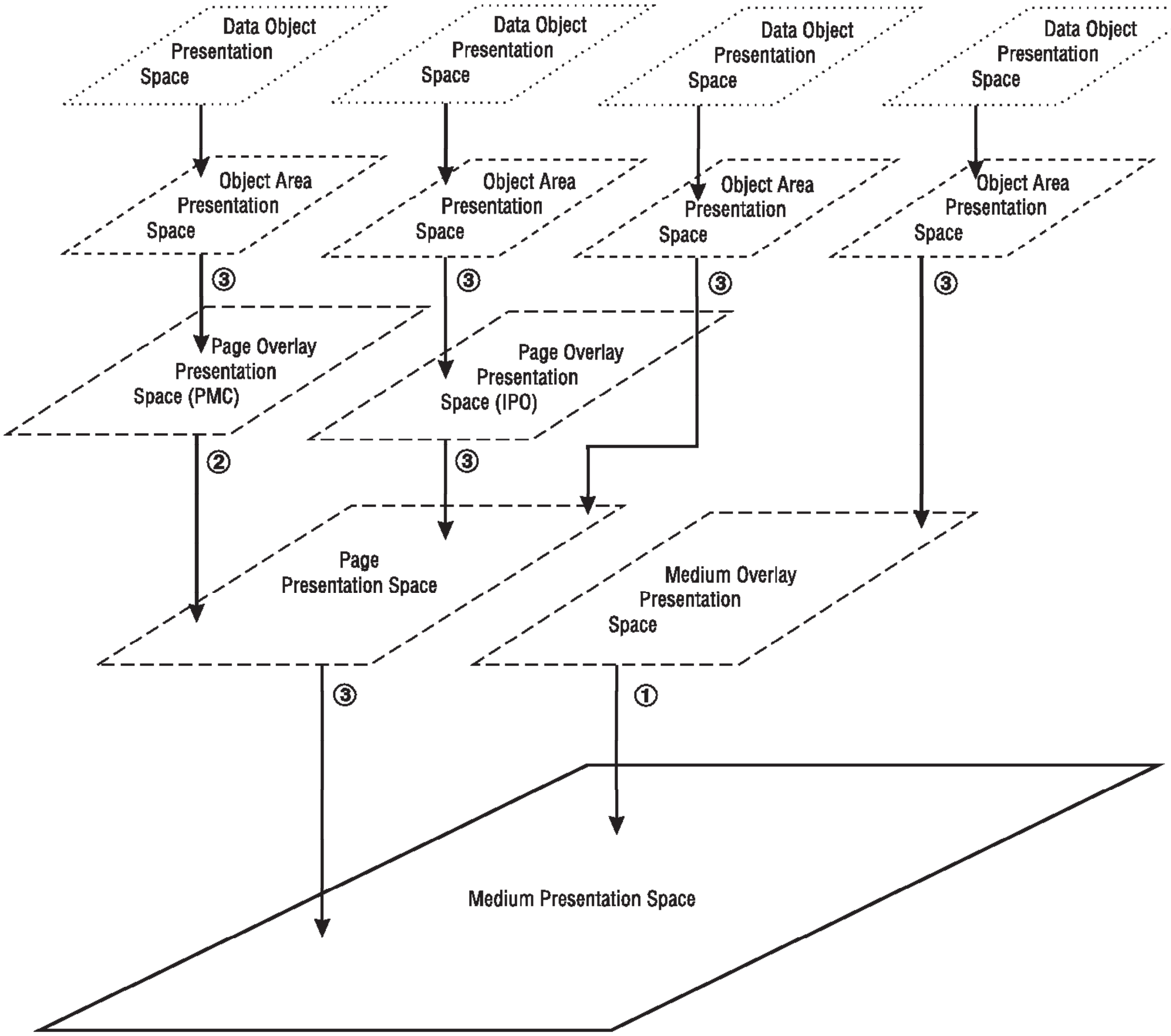
Merging Presentation Spaces

Presentation spaces in a MO:DCA document are merged in the order in which the document components that define these presentation spaces appear in the data stream, as follows:

- **Medium presentation space**. This is the base MO:DCA presentation space upon which all other presentation spaces are merged.
  - **Medium overlay presentation space**. Merged on the medium presentation space with a keyword on the Medium Modification Control (MMC) structured field in a Medium Map. Medium overlays are merged on the medium presentation space before any pages are merged. Multiple medium overlay presentation spaces are merged in the order in which their keywords appear on the MMC structured field.
  - **Page presentation space**. Merged on the medium presentation space in the order in which the corresponding page appears in the document, in accordance with the specifications in the active Medium Map.
    - **Object area presentation space**. Merged on the page presentation space in the order in which the corresponding data object is included on the page.
      - **Data object presentation space**. Merged on the corresponding object area presentation space.

- **Page overlay presentation space**. If the page overlay is included via an IPO, it is merged on the page presentation space in the order in which the overlay is included on the page. If the page overlay is included via a PMC in a Medium Map, it is merged on the page presentation space before any data objects or overlays included via an IPO are merged.
  - **Object area presentation space**. Merged on the overlay presentation space in the order in which the corresponding data object is included on the overlay.
    - **Data object presentation space**. Merged on the corresponding object area presentation space.

The MO:DCA presentation space merge-order is shown in Figure 12 on page 46.


Data Object Presentation Space
Data Object Presentation Space
Data Object Presentation Space
Data Object Presentation Space
Object Area Presentation Space
Object Area Presentation Space
Object Area Presentation Space
Object Area Presentation Space
③
③
③
③
Page Overlay Presentation Space (PMC)
Page Overlay Presentation Space (IPO)
②
③
Page Presentation Space
Medium Overlay Presentation Space
③
①
Medium Presentation Space

① Merged first on the medium presentation space as specified in a Medium Map print control object. Multiple medium overlays are merged in the order in which they occur.

② Merged first on the page presentation space as specified in a Medium Map print control object. Multiple overlays are merged in the order in which they occur in the data stream.

③ May occur multiple times and is merged in the order in which it occurs in the data stream.

Figure 12. Merging Presentation Spaces

Mixing Rules

When multiple MO:DCA presentation spaces are merged, the background and foreground of the presentation spaces *mix*. The resultant foreground is the union of all presentation space foregrounds, that is, once an area is defined to be foreground, it remains foreground even if its color attribute is changed due to an "underpaint" mixing rule. The resultant background is everything else. The color of the resultant foreground and background is determined by the mixing rules specified in the MO:DCA architecture.

When a new presentation space P_n is merged onto an existing presentation space P_e, four types of mixing must be considered. Let F_e and B_e denote the P_e foreground and background, respectively, and let F_n and B_n denote the P_n foreground and background, respectively, then the mixing types can be characterized as follows:

Mixing Type	**Description**
B_n on B_e	Background on background
B_n on F_e	Background on foreground
F_n on B_e	Foreground on background
F_n on F_e	Foreground on foreground

For each type of mixing, the resultant color is determined by the mixing rule that is specified. The following mixing rules are defined for presentation space mixing:

Mixing Rule **Definition**

Overpaint When part of P_n overpaints part of P_e, the intersection is assigned the color attribute of P_n. This is also referred to as *opaque* or *knock-out* mixing.

Underpaint When part of P_n underpaints part of P_e, the intersection keeps the color attribute of P_e. This is also referred to as *transparent* mixing or *leave alone* mixing.

Blend When part of P_n blends with part of P_e, the intersection assumes a new color attribute which represents a color-mixing of the color attribute of P_n with the color attribute of P_e. For example, if P_n has foreground color attribute blue and P_e has foreground color attribute yellow, the area where the two foregrounds intersect would assume a color attribute of green.

Default Mixing Rule

When no presentation space mixing rule is specified, the following default MO:DCA mixing rule applies:

When a new presentation space P_n is merged onto an existing presentation space P_e, the background of P_n underpaints the background and foreground of P_e, and the foreground of P_n overpaints the background and foreground of P_e.

This default mixing rule can be summarized as follows:

Table 13. Default Color Mixing Rules

Mixing Type	**Default Mixing Rule**
B_n on B_e	Underpaint
B_n on F_e	Underpaint
F_n on B_e	Overpaint
F_n on F_e	Overpaint

Font Technologies

The MO:DCA architecture supports references to various font technologies for rendering character data. These font technologies can be separated into two classes:

FOCA fonts

Non-FOCA fonts, also called *data-object fonts*

FOCA fonts have a structure that is defined by the Font Object Content Architecture (FOCA). They are referenced in a MO:DCA data stream using a Map Coded Font (MCF) structured field. Non-FOCA fonts are fonts who's structure is not defined by the FOCA architecture. The structure of such fonts is not modified when they are used in MO:DCA data streams and in AFP environments. However, such fonts may be carried in MO:DCA object containers, if, for example, they are to be placed in an AFP resource group. Non-FOCA fonts are referenced in a MO:DCA data stream using a Map Data Resource (MDR) structured field. Examples of non-FOCA fonts that are supported in MO:DCA data streams are TrueType fonts (TTFs) and OpenType fonts (OTFs).

Relationship Between FOCA Character Metrics and TrueType Character Metrics: Implementation Issues

It is important to have consistent presentation results regardless of the font technology used. The FOCA Architecture defines the basic concepts and provides a rich set of font and character metrics; these FOCA concepts lay out the presentation goals. The PTOCA architecture provides the capability to present strings of text at various orientations as shown in Figure 73 on page 422. The following describes the relationship between various TrueType metrics and the corresponding FOCA-defined metrics and provides recommendations for simulating metrics that are needed for presentation but are not directly provided in some TrueType fonts.

Horizontal Metrics

When a TrueType rasterizer RIPs the outline descriptions into character bitmaps, TrueType metrics are provided for positioning the bitmaps horizontally within a line of text. These metric values provide enough information to calculate the metrics defined by FOCA for the 0 degree character rotation. This information includes the width and depth of the bitmap, the distance from the character origin to a corner of the bitmap, and the distance to the origin of the next character.

Figure 13 on page 49 compares the parameters commonly used with TrueType fonts to the horizontal (0 degree) metrics provided by a FOCA font. In practice, many TrueType fonts are built so that there is no top indent or left indent; in this case, the bitmap is a tight box around the character and the indent values are zero.

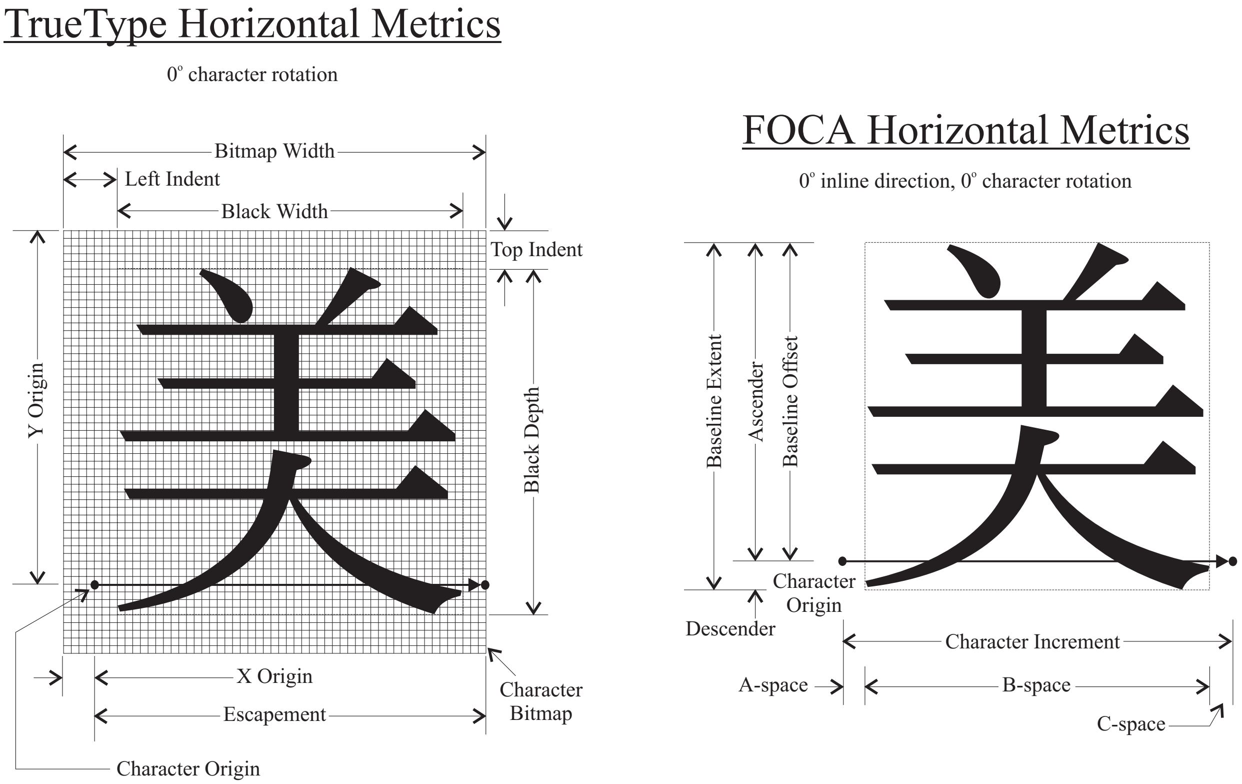


Figure 13. Horizontal Metrics: TrueType/OpenType Fonts and FOCA Fonts

Based on this illustration, the key FOCA horizontal metrics can be calculated as follows:

```
Character Increment (HCI) = Escapement
A-space (HAS) = Left Indent - X Origin
B-space (HBS) = Black Width
C-space (HCS) = Escapement - A-space - B-space
Baseline Extent (HBE) = Black Depth
Baseline Offset (HBO) = Y Origin - Top Indent
Character Descender (HCD) = Top Indent + Black Depth - Y Origin
```

The FOCA metrics for 180- degree rotation (upside-down) have a simple relationship to those for 0-degree rotation. The A-space and the C-space metrics are reversed, as are the baseline offset and character descender metrics. The character increment, B-space, and baseline extent metrics are identical.

Note that, in practice, font rasterizers don't provide all of the parameters shown in the picture, but do provide other parameters. For example, the font rasterizer can return the offset (xorigin, yorigin) from the character origin of the top-left corner of the bitmap. This information can be related to the metrics formulas; for example:

```
A-space (HAS) = Left Indent - X Origin = Left Indent + xorigin
Baseline Offset (HBO) = Y Origin - Top Indent = yorigin - Top Indent
```

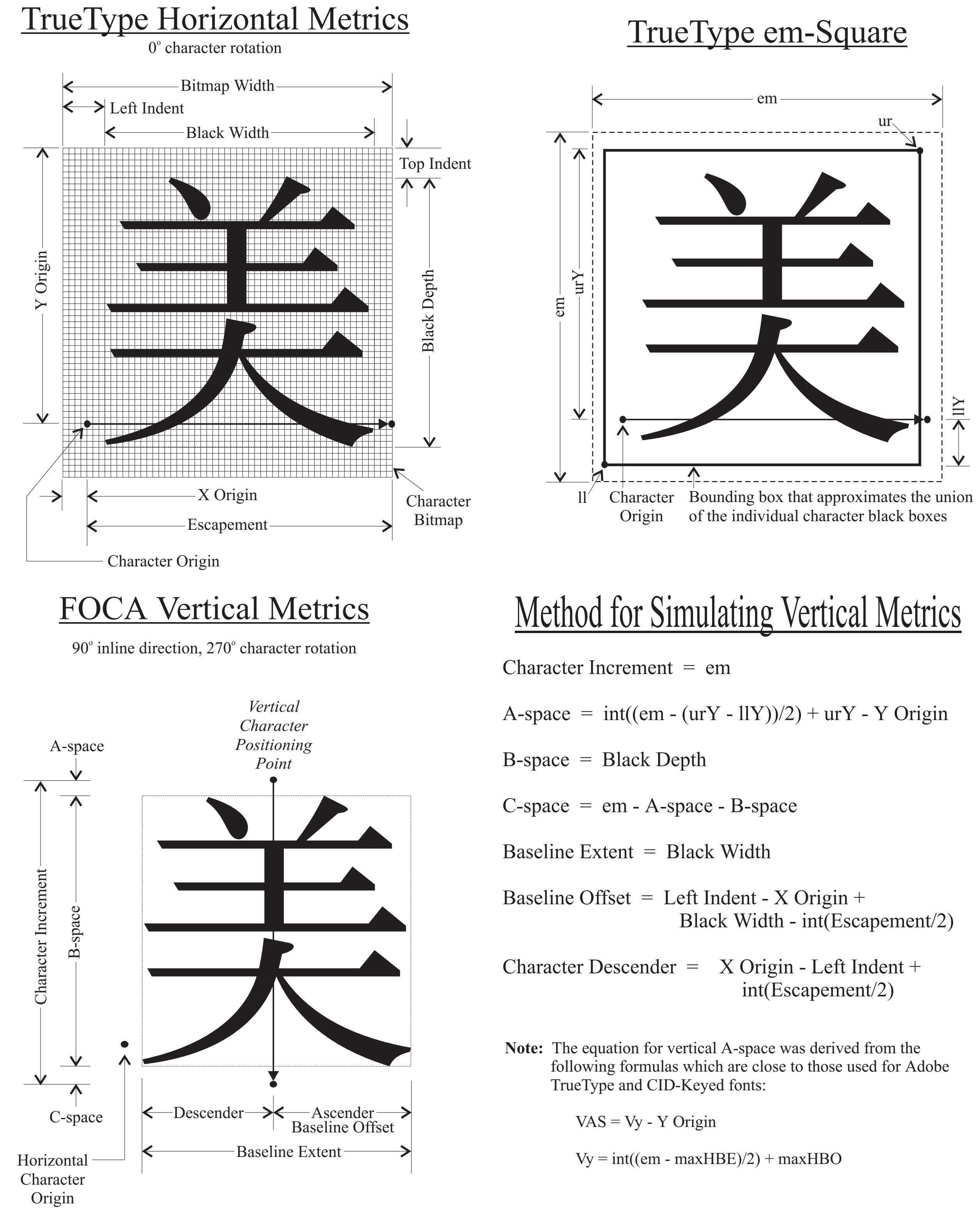
Vertical Metrics

Character rotations of 90 and 270 degrees are used to support vertical forms of writing. In addition to the metrics mentioned earlier, vertical positioning and character increment metrics are needed to place characters in these rotations. Some TrueType fonts provide metrics for vertical writing in a structure called a "vtmx table", but others don't provide these metrics. The TrueType advance height corresponds to the FOCA vertical character increment (VCI) and the TrueType top sidebearing corresponds to the FOCA vertical A-space (VAS), but there is no TrueType metric that corresponds to the FOCA baseline offset.

When the vtmx metrics are available they can be used to calculate the equivalent FOCA vertical metrics. But, when the font designer omitted them or when they can’t be obtained from the TrueType rasterizer, a method is needed to estimate appropriate FOCA equivalent values.

Simulating Vertical Metrics

Figure 14 on page 51 shows again the TrueType horizontal metrics and some additional TrueType metrics that can be obtained to describe the em-square. The figure also shows the target FOCA vertical metrics and a method for simulating 270 degree FOCA vertical metrics from TrueType horizontal metrics.



Method for Simulating Vertical Metrics

Character Increment = em

A-space = int((em - (urY - llY))/2) + urY - Y Origin

B-space = Black Depth

C-space = em - A-space - B-space

Baseline Extent = Black Width

Baseline Offset = Left Indent - X Origin + Black Width - int(Escapement/2)

Character Descender = X Origin - Left Indent + int(Escapement/2)

Note: The equation for vertical A-space was derived from the following formulas which are close to those used for Adobe TrueType and CID-Keyed fonts:

VAS = Vy - Y Origin

Vy = int((em - maxHBE)/2) + maxHBO

Figure 14. Vertical Metrics: TrueType/OpenType Fonts and FOCA Fonts

Any approach taken to approximate these metrics is well served to consider the scripts in which vertical writing is most popular: East Asian scripts which use ideographic characters. These full width characters have properties that can be utilized to make these estimations. First, they typically have an equal, or fixed, increment. Second, they are designed on a square grid, so their width and height are equal. Third, they are usually the largest characters in the font.

For these reasons, using a fixed vertical character increment (VCI) equal to the largest horizontal increment will be quite satisfactory for vertical writing. Generally, the maximum values for many basic metrics, such as character increment, descender, and baseline offset can be obtained from the font file. Alternatively, the properties listed previously make it reasonable to set VCI to the Em-Space Increment. The Em-space is defined such that one em equals the height of the design space. Scalable font metrics are expressed as fractions of this unit-Em.

These alternatives can be summarized mathematically as:

```
Character Increment (VCIestimated) = max(Escapement)
  — or —
Character Increment (VCIestimated) = 1 em
```

Techniques to estimate appropriate values for VAS must keep two goals in mind. First, it should result in the bitmaps of ideographic characters being placed within the vertical increment. Second, the vertical position of the bitmap should reflect the relative horizontal baseline offset of the character. For example, the bitmap widths for the BLACK LENTICULAR BRACKETS, U+3010 and U+3011, are small compared to their increment and are designed to be positioned close to the character they enclose. This property must be preserved for vertical writing.

To accomplish these goals, first compute a constant value (Vy) to place the horizontal character origin relative to the vertical character positioning point, using the TrueType em-square metrics and the following equation (note that max(HBE) = urY + llY and max(HBO) = urY):

```
Vy(est) = int((em - max(HBE))/2) + max(HBO)
```

The first component of this equation, int((em - max(HBE))/2), is designed to position all of the character bitmaps of the font within the vertical increment. The second component, max(HBO), calibrates the V Origin metric to the highest character(s) within the font. With this reference, then calculate VAS for individual characters with the equation:

```
VASestimated = Vy(est) - Y Origin
```

and achieve the design goals.

For fonts that are not based on ideographic characters, a different method of constructing a vertical character increment and A-space could be used. For example, a fixed percentage (20%) of extra space, based on the desired pointsize, could be added to the black depth to yield the VCIestimated. The extra space could be divided evenly between the vertical A-space and vertical C-space. For characters without any black depth (space characters), the pointsize could be used as VCIestimated.

The last task to address is estimating the horizontal position of the character bitmap. For vertical rotations, this is reflected in the baseline offset (VBO) and character descender (VCD) metrics. Similar to the goal for vertical positions, this metric should reflect the character's horizontal position within its horizontal increment. Therefore, the metric calculations should essentially center the character's horizontal increment on the baseline and preserve its horizontal position with respect to the increment. This is achieved with the equations:

```
Baseline Offset (VBO) =  Left Indent - X Origin + Black Width - int(Escapement/2)
Character Descender (VCD) =  X Origin - Left Indent + int(Escapement/2)
```

The remaining metrics for 270-degree character rotation can be calculated from the horizontal bitmap metrics and those derived previously:

```
Baseline Extent (VBE) = Black Width
B-space (VBS) = Black Depth
C-space (VCS) =  VCI - VAS - Black Depth
```

The vertical metrics for 90-degree character rotation can be directly deduced from the 270-degree metrics, in the same manner used to convert 0-degree metrics to 180-degree metrics.

Document Indexing

The document index defined by the MO:DCA architecture provides functions for indexing the document based on document structure and on application-defined document tags. The index is delimited by a Begin Document Index structured field and an End Document Index structured field and may be located within the document or external to the document. MO:DCA elements that may be indexed are pages and page groups. When referenced by an index, they are called *indexed objects*. The MO:DCA elements within a document index that reference indexed objects are Index Element (IEL) structured fields. The MO:DCA elements within a document index that support content-based tagging are Tag Logical Element (TLE) structured fields.

A MO:DCA document index consists of the following structured fields. These structured fields are described in detail in Chapter 5, “MO:DCA Structured Fields,” on page 103. Note that the IEL and TLE structured fields may occur multiple times.

```
Begin Document Index (BDI)
    Index Element (IEL)
    Link Logical Element (LLE)
    Tag Logical Element (TLE)
End Document Index (EDI)
```

When the document index is external to the document, the BDI structured field references the document using a Fully Qualified Name type X'83' triplet. The document name specified in this triplet is inherited by all IEL and TLE strucured fields in the index.

Index Elements

The Index Element (IEL) structured field supports indexing of pages and page groups. When an IEL references an indexed object, the type of indexed object (page or page group) is indicated by the name reference to the indexed object. The name of the IEL structured field is specified by a Fully Qualified Name type X'CA' triplet, and the name of the indexed object is specified by either a Fully Qualified Name (FQN) type X'87' triplet for a page or by a FQN type X'0D' triplet for a page group. An IEL that references a page is called a page-level IEL. An IEL that references a page group is called a page-group-level IEL. A MO:DCA index may contain page-level IELs, page-group-level IELs, or both. The order in which page-level IELs and page-group-level IELs appear in the index must be the same as the order in which the indexed Begin Page and Begin Page Group structured fields appear in the document.

The IEL structured field provides the following information for the indexed object:

- Direct byte offset of the Begin indexed object structured field from the start of the Begin Document structured field.
- Byte extent of the indexed object, from the first byte in the Begin structured field to the last byte in the End structured field.
- Structured field offset of the Begin indexed object structured field, where the Begin Document structured field has offset 0, and all following structured fields increment the offset by 1.
- Structured field extent of the indexed object, which is a count of the number of structured fields in the indexed object, starting with the Begin indexed object structured field and ending with the End indexed object structured field.

- Object offset of the Begin indexed object structured field, using a specified object type. For example, this parameter may specify the number of pages that precede an indexed page group in the document.
- Object extent of the indexed object, using a specified subordinate object type. For example, if the subordinate object is a page, this parameter may specify the number of pages in an indexed page group.
- If the indexed object is a page:
  - The name of the medium map object that is active for formatting the indexed page on a physical medium
  - The number of the indexed page in the set of sequential pages controlled by the active medium map, where the first page in the set is number 1
  - The PGP repeating group used to process the page.
- If the indexed object is a page group:
  - The number of pages that precede the page group in the document
  - The number of pages contained in the page group
  - The name of the medium map object that is active for formatting the first page in the indexed page group on a physical medium
  - The number of the first page-group page in the set of sequential pages controlled by the active medium map, where the first page in the set is number 1, and where “active medium map” refers to to the medium map that is active at the beginning of the page-group.
  - The PGP repeating group used to process the first page-group page.

An example of a page-level IEL that specifies page offset and page extent is shown in Figure 15.

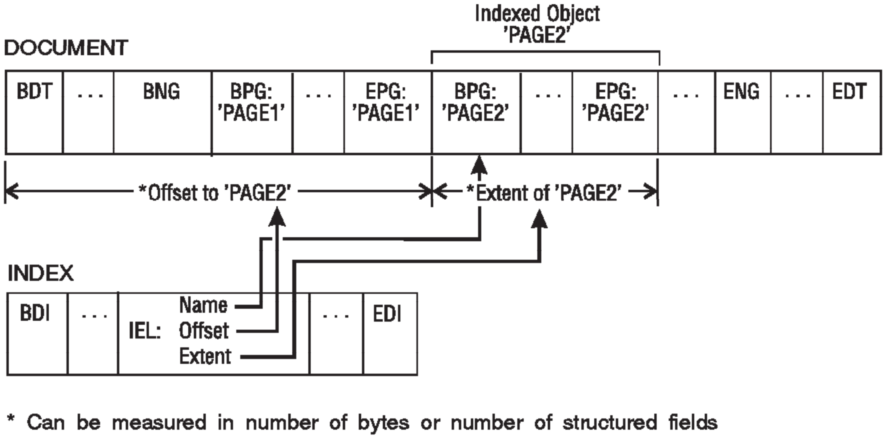


Figure 15. Page-level IEL: Offset and Extent

An example of a page-group-level IEL that specifies page group offset and page group extent is shown in Figure 16 on page 56.

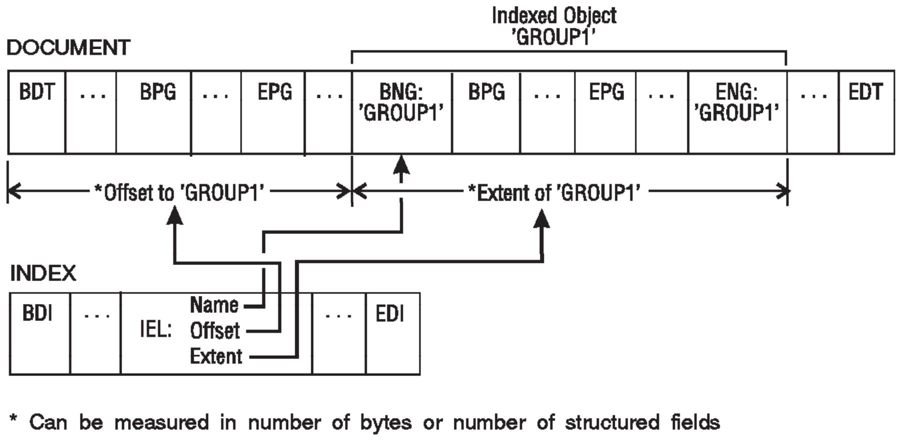


Figure 16. Page-group-level IEL: Offset and Extent

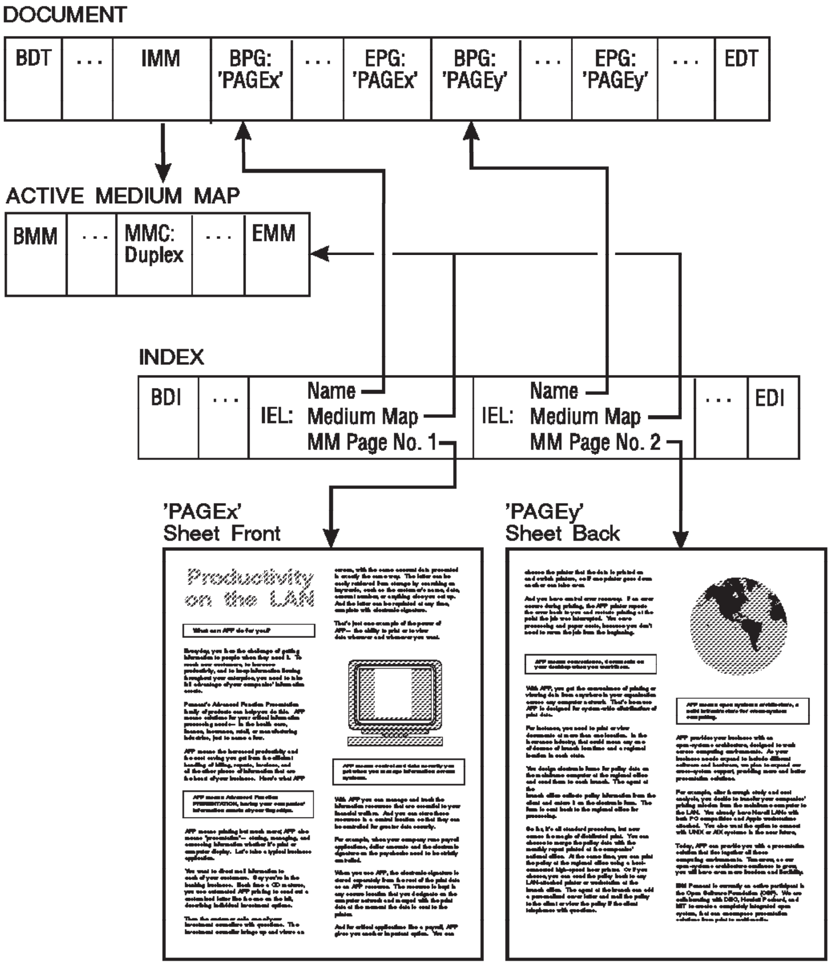
Figure 17 on page 57 shows how the Medium Map information in a page-level IEL is used to determine page placement on a side of a sheet.

Tag Logical Elements

The Tag Logical Element (TLE) structured field supports the tagging of pages and page groups with an attribute that may be used as an index key. The attribute is specified using attribute name and attribute value triplets on the TLE structured field. When the TLE is specified in a document index, the element to be tagged may be identified using a Fully Qualified Name triplet on the TLE structured field:

- FQN type X'87' triplet for a page
- FQN type X'0D' triplet for a page group

If a TLE in a document index does not contain an explicit page or page group reference, it inherits such a reference from the last preceding IEL in the index. A TLE that explicitly references a page, or that inherits a page reference from the last preceding IEL, is called a page-level TLE. A TLE that explicitly references a page group, or that inherits a page group reference from the last preceding IEL, is called a page-group-level TLE.



NOTE: IEL contains sufficient presentation-control information to present the page on media without processing the entire document.

Figure 17. Page-level IEL: Use of Medium Map Information

The TLE structured field tags the referenced element with the following information:

- Name of the attribute
- Value of the attribute
- Sequence number of the attribute, used to distinguish otherwise identical attributes
- Level number of the attribute, used to logically position the attribute in an application-defined hierarchy

Figure 18 on page 58 shows how logical tags are applied to pages in a document using TLEs in an external document index.

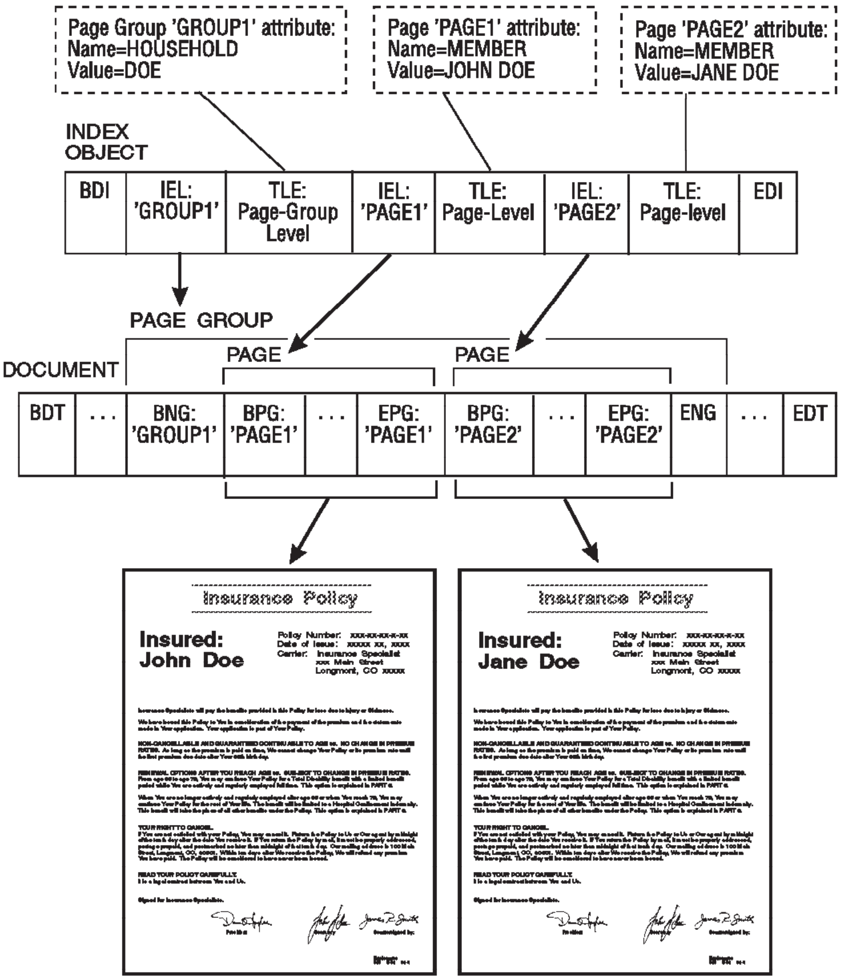


Figure 18. A Document with Logical Tags

Document Links

Online, interactive forms of document processing require that linkages be established among components within the document and from components within the document to components external to the document. One example of such processing is the use of *hypertext* links, which are logical connections from one string of text in a document to another string of text that is contextually related to the first. A viewing application can highlight the source text, such as a technical term, and using hypertext links can provide the user with the option of jumping to the linked text that is the glossary definition of the technical term. Another example is the processing of annotations. A reviewer of a document may add comments to a string of text in a source document, and require a link to connect these comments as annotations to the appropiate area in the source document. A third example is the processing of appends. A document may be composed of pages summarizing monthly phone calls. If a particular phone call is recorded late, it may need to be appended to an existing page in the document, which requires a link from the existing page to the document component that contains the late phone bill.

Document links in the MO:DCA architecture are supported with Link Logical Element (LLE) structured fields.

Link Logical Elements

Link Logical Elements (LLE) structured fields are process elements that provide a general and extendable linking capability between document components such as documents, page groups, pages, overlays, data objects, and logical tags. The LLE structured field identifies a source and a target and specifies the purpose of the link from source to target. The LLE optionally can specify a name that may be used to reference the LLE and parameter data to be associated with the link.

LLEs may be embedded directly in the document that contains the source for the link. In that case, the source link specified in the LLE inherits the document name and the names of all objects that are higher in the document hierarchy. For example, if the LLE is in a page that is part of a page group, and if the source link specifies an area on the page, then the source link inherits the names of the document, page group, and page.

LLEs may be embedded directly in the document that contains the target for the link. In that case, the target link specified in the LLE inherits the document name and the names of all objects that are higher in the document hierarchy. For example, if the LLE is in a page that is part of a page group, and if the target link specifies an area on the page, then the target link inherits the names of the document, page group, and page.

LLEs may also be embedded in the index for the document that contains the source for the link, the target for the link, or both. In that case, the source or target link in the LLE can inherit the document name from the index if the document name is not explicitly specified in the respective repeating group. The source or target link may also inherit the page or page group name specified by a preceding Index Element (IEL) structured field if such names are not specified by the corresponding repeating group in the LLE and if the repeating group specifies an object that is lower in the document hierarchy than the object defined by the IEL.

Document links defined by LLEs do not provide a presentation specification. It is left up to the application using the LLEs to determine how to present the relationship between document components that are linked with an LLE. For example, if an LLE is used to link a source document page to an object containing an annotation, a viewing program may choose to highlight the annotated area on the source page and to display the annotation in a separate window next to the source page. On the other hand, a print subsystem may choose to simply gather all annotations and print them at the end of the source document with appropriate pointers to the source pages.

An example showing how an LLE can be embedded in a document index to link an area on a page in the source document to a text annotation is shown in Figure 19 on page 60.

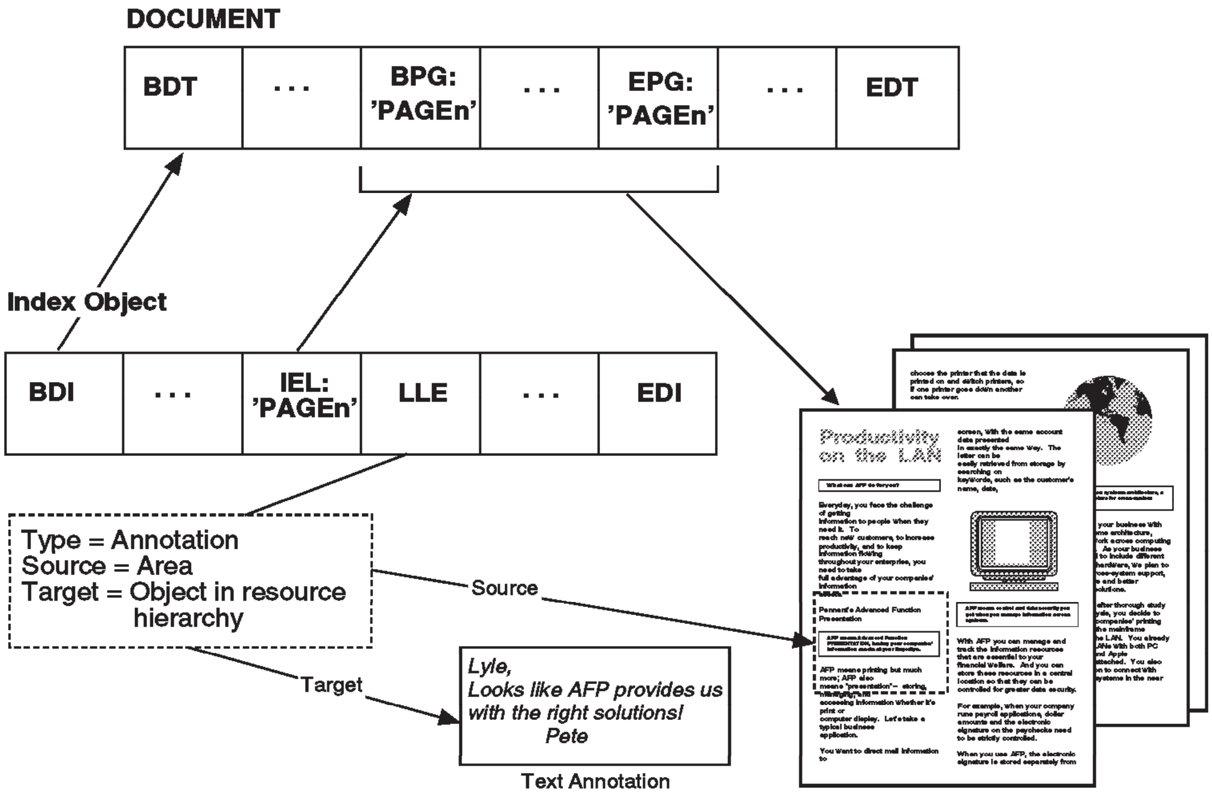


Figure 19. Document Annotation using the LLE

Annotations and Appends

An *annotation* is a comment or explanation that is associated with the contents of a source document. Annotations are normally generated based on a review of the final-form document using an interactive presentation device such as a document viewer. Annotation data can be generated with a variety of data types such as text and image, and can be carried within a number of document components including object containers, overlays, pages, page groups, resource groups, and documents. Annotations are linked to the source document component to which they apply using a Link Logical Element structured field.

An *append* is an addition to a source document component or a continuation of a source document component. Appends can be generated with any MO:DCA-P document component. The simplest form of an append is one document appended to another document. Appends are linked to the source document component to which they apply using a Link Logical Element structured field.

The location of document components that carry annotations and appends follows the normal MO:DCA-P object structure rules. For example, if an annotation is built using a page or a page group, it must be carried in a document. If it is built using a data object, resource object, or object container, it can be carried in a resource group.

N-up Presentation

N-up is a presentation format where multiple pages are presented on a single physical medium. This format provides the user with a high degree of flexibility for composing page objects onto sheets. When used on a continuous-forms printer with a wide carriage, it can result in significant paper savings and improvements in print reliability. In N-up presentation, each side of the physical medium is divided into a number of equal-size partitions, where the number of partitions is indicated by the number “N” in “N-up”. If duplex is specified, the same N-up partitioning is applied to the back side as is applied to the front side. With simplex N-up presentation, N pages are placed on the physical medium, and with duplex N-up presentation, 2N pages are placed on the physical medium. Pages are placed into partitions using either a *default N-up page placement* or an *explicit N-up page placement*, as specified in the Page Position (PGP) structured field. In the default N-up page placement, consecutive pages in the data stream are placed into consecutively-numbered partitions. In explicit N-up page placement, consecutive pages in the data stream are placed into explicitly-specified partitions. For more information on page placement, see “Page Position (PGP) Format 2” on page 282. Pages may be rotated within their partitions, and Page Modification Control (PMC) overlays may be applied to pages before they are placed in their partition.

Figure 20 shows the partitioning for wide continuous-forms media, narrow continuous-forms media, and cut-sheet media; partitioning is not used with envelope media. Partition numbering for various media is shown in Figure 58 on page 290 to Figure 69 on page 296.

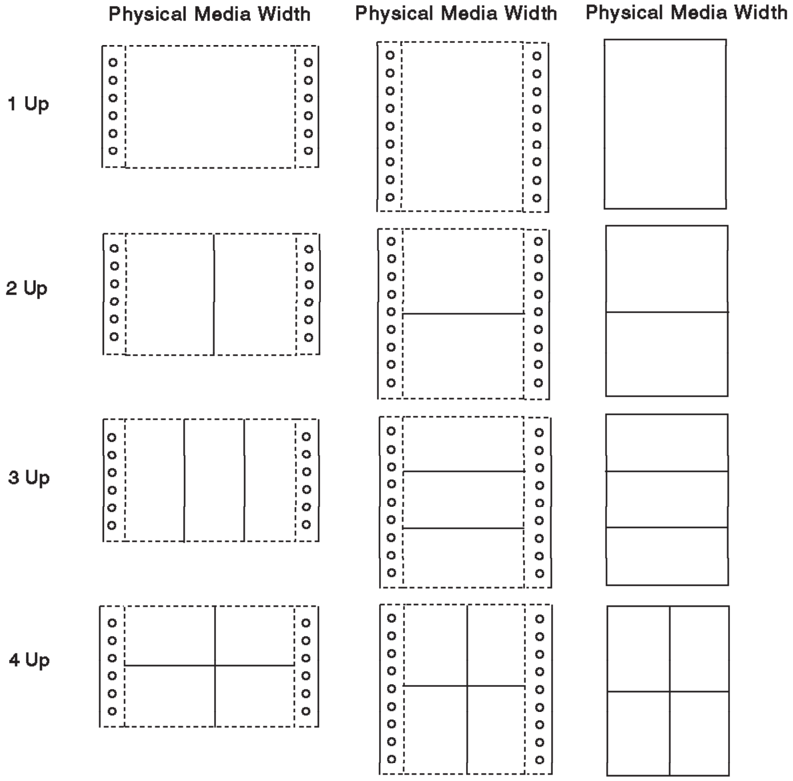


Figure 20. N-up Partitions for Various Physical Media

Cut-sheet Emulation (CSE) Print Mode

Some IPDS printers provide a *cut-sheet emulation mode* that can be used to print on continuous-forms media that, once slit and collated, emulates two sheets of cut-sheet output. In this mode, the printer logically divides the continuous-forms media in half parallel to the carrier strips and controls the placement of pages on either the left side or the right side of the physical media as defined by a printer configuration option. The two portions of the physical media are called *sheetlets* and are treated as if they were two separate pieces of cut-sheet media. This logical division of the continuous-forms media is shown in Figure 21. When a MO:DCA document is sent to a print server for printing in CSE mode, MO:DCA sheets and their content are mapped to cut-sheet CSE sheetlets at the printer. Note that the top of each sheetlet is a narrow edge, and the default sheetlet origin is the top-left corner of the sheetlet.

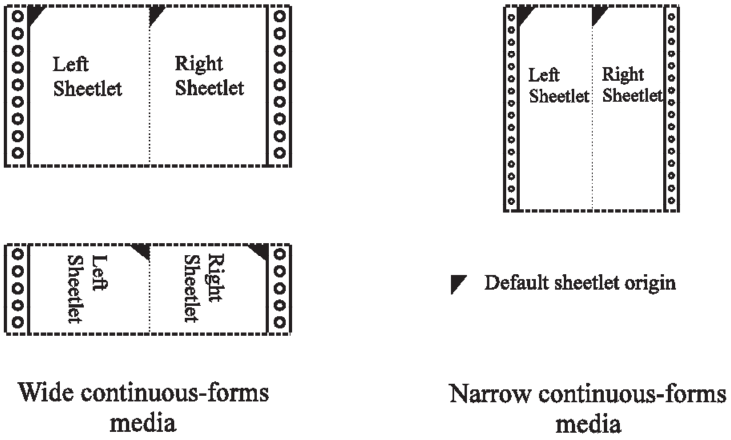


Figure 21. Logical Division of Continuous Forms for Cut-sheet Emulation

The printer is configured for cut-sheet emulation mode by the printer operator while the printer is disconnected from the print server. Cut-sheet emulation mode is activated by the print server after the printer has indicated support for the mode. Note that cut-sheet emulation mode is not supported in viewing environments. Note also that cut-sheet emulation mode is not supported with N-up presentation. When N-up is specified in the active Medium Map, CSE mode is deactivated for the duration of that Medium Map.

When finishing operations are specified for a printer operating in CSE mode, the operations are specified for and applied to each CSE sheetlet. That is, for finishing operations in CSE mode, the media is the sheetlet. This is true whether the finishing operation is specified with a Finishing Operation (X’85’) triplet or a UP3i Finishing Operation (X8E’) triplet.

Document Finishing

Finishing operations, such as stapling and folding, for a print file may be specified using structures in the form definition invoked for the print file. Such finishing operations may be applied at different levels of the print file, and at each level the finishing operations have a defined scope:

- *Print-file-level finishing:* the scope is the complete print file.
- *Document-level finishing, all documents:* the scope is each individual document in the print file.

- *Document-level finishing, selected document:* the scope is a single document in the print file.
- *Medium-map-level finishing, group of sheets:* the scope is a collection of sheets.
- *Medium-map-level finishing, each sheet:* the scope is a single sheet.

Finishing operations for all levels are specified with a Medium Finishing Control (MFC) structured field. For print-file-level and document-level finishing, the MFC is specified in the document environment group (DEG) of the form definition. For medium-map-level finishing, the MFC is specified in a medium map.

The actual finishing operation and its parameters are specified on the MFC with finishing triplets. Two triplets are supported:

- Finishing Operation (X'85') triplet
- UP3i Finishing Operation (X'8E') triplet

These two triplets may be specified in any combination at any level, however the finishing operations must be compatible.

When more than one finishing operation that involves a collection of media is specified for some portion of the print file, a nesting of the operations is defined first by the scope of the operation (print file, document, medium collection), and second by the order of the operation in the data stream. Finishing operations with an inherently broader scope, for example, operations at the print file level, are nested outside of finishing operations with an inherently narrower scope, for example, operations at the medium-map-level. If more than one operation is specified with the same scope, the order of the finishing operation triplets defines the order of the nesting. The first finishing operation specified defines the outermost nesting, and the last finishing operation specified defines the innermost nesting. When a finishing operation is applied, all finishing operations nested inside this operation are also applied. Finishing operations that are nested outside this operation are not affected. For a complete definition of the finishing operation nesting rules, see “Finishing Operation Nesting Rules” on page 242.

Exception Conditions

The application creating the data stream is responsible for producing a valid MO:DCA data stream, and the application using the MO:DCA data stream is responsible for preserving a valid format. Nonetheless, exception conditions may arise. A valid MO:DCA data stream is one that does not violate the architecture. A MO:DCA data stream is in violation of the architecture when its structure or contents do not conform to the requirements of the architecture.

An error is a product failure that produces or results in a data stream that violates the architecture. Since the cause of an architecture violation cannot be determined when an application interprets a data stream, all architecture violations are handled as exception conditions.

If absolute fidelity of a presentation document is not required, MO:DCA documents can be interchanged among a larger set of products. It is possible for the processor of a MO:DCA data stream to continue processing when it encounters exception conditions. This permits a process that cannot faithfully present a document to continue with its best approximation.

Classifications

Exception conditions can be classified as:
- Syntactic
- Semantic

Syntactic exception conditions defined for this architecture include:
- Invalid or unknown structured field introducer (SFI); see "MO:DCA Structured Field Syntax" on page 20 for further discussion
- Invalid or unknown parameter within a recognized structured field
- Invalid parameter value within a recognized structured field
- Component appears in an invalid location or is missing
- Structured field appears in an invalid location or is missing
- Parameter is missing within a recognized structured field

Semantic exception conditions defined for this architecture include:
- Inconsistent or contradictory specifications
- Invalid relationships among the data-stream structured fields

Detection

A MO:DCA-compliant product must detect the exception conditions defined by the architecture that apply to the interchange set supported, within the scope of the supported OCAs. Exception conditions detected in the structured fields and parameters that it interprets as it processes the data stream should be identified to an exception handler within the receiver. The MO:DCA architecture defines eight categories of exception conditions that can occur in an interchange data stream. The eight categories and their descriptions are as follows:

Category **Description**

Invalid structured field identifier
The structured field identifier contains invalid parameter values. Examples are structured field identifiers with length values less than eight or invalid flag settings. Not included in this category are invalid class codes, type codes, or category codes.

Unrecognized identifier code
This exception condition is caused by an unrecognized structured field identifier code. It includes class codes or type codes that are not valid in this architecture, or that are valid in this architecture, but are not acceptable in the particular interchange set being used. It does not include invalid category codes.

Data stream state violation
A valid structured field appears in an invalid context in the data stream. This exception includes:

- Repetition of a structured field within a state where repetition is not permitted. An example is the appearance of two Page Descriptor structured fields in a MO:DCA-P Active Environment Group.
- Appearance of a structured field within a state where it is not permitted. An example is a Page Descriptor structured field appearing in a MO:DCA-P Object Environment Group.
- Appearance of a structured field outside the specified structured field order for that particular state. An example is a Begin Presentation Text Object structured field appearing in a MO:DCA-P Page before the Active Environment Group.

Note: Not included in this category is the omission of a required structured field.

Unrecognized structured field or triplet

This exception includes:

- An SFI containing a category code:
  - That is not valid in this architecture, or
  - That is valid in this architecture, but is not acceptable in the particular interchange set being used
- A triplet containing an identifier:
  - That is not valid in this architecture, or
  - That is valid in this architecture, but is not valid in the particular interchange set being used

Required structured field missing

A structured field, required to begin a containing component or to satisfy an explicit invocation, is missing from the correct location in the data stream. An example is a Begin Active Environment Group structured field missing from the beginning of a page overlay.

Required parameter missing

A parameter or parameter group, required in a specific structured field or in a set of structured fields, is missing from the document component where it is required. An example is a Begin Document structured field missing a Coded Graphic Character Set Global Identifier triplet.

Unacceptable parameter value

A parameter contains a value that is not valid in this architecture, or it contains a value that is valid in this architecture, but that is not acceptable in the particular interchange set being used. An example is a value of 254 for the X page units-base parameter in a Page Descriptor structured field. See “PGD (X'D3A6AF') Syntax” on page 279.

Inconsistent parameter values

A parameter contains a value that is inconsistent with the value of another parameter in the structured field, or a parameter in another structured field. An example is a name in an end structured field that does not match the name in the corresponding begin structured field.

MO:DCA syntax tables identify the categories of exception conditions that can occur for each data element through the use of a code listed in the *Exc* column. Each of the exception conditions is related to a bit position, as shown in Table 14 on page 66. The value assigned to *Exc* is based on the positions of the bits that represent the exception condition categories that can apply to the data element. If no exception condition is possible, the *Exc* column will contain X'00'.

For example, if it is possible for the data element to contain a value outside of the prescribed range, or if it is possible for its value to conflict with that of another parameter, then both the unacceptable parameter value and the inconsistent parameter value exception conditions can apply. The unacceptable parameter value is represented by bit position six or B'00000010', and the inconsistent parameter value is represented by bit position seven or B'00000001'. The code that is entered into the *Exc* column is formed by ORing the bit representations of the exception condition categories that are possible, in this example resulting in B'00000011' or X'03'.

Table 14. Bit Representation of MO:DCA Exception Condition Categories

Bit Position	Exception Condition Category	Code	
		Binary	Hexadecimal
Bit 0	Invalid structured field identifier	B'10000000'	X'80'
Bit 1	Unrecognized identifier code	B'01000000'	X'40'
Bit 2	Data stream state violation	B'00100000'	X'20'
Bit 3	Unrecognized structured field or triplet	B'00010000'	X'10'
Bit 4	Required structured field missing	B'00001000'	X'08'
Bit 5	Required parameter missing	B'00000100'	X'04'
Bit 6	Unacceptable parameter value	B'00000010'	X'02'
Bit 7	Inconsistent parameter values	B'00000001'	X'01'
None	None	B'00000000'	X'00'

Exception Action

The action to be performed by a product that detects an exception condition is product-dependent.

Chapter 4. MO:DCA-P Objects

This chapter:
- Defines the structure of a MO:DCA-P printfile
- Defines the structure of a MO:DCA-P document
- Defines the structure of a MO:DCA-P index
- Defines the structure of a MO:DCA-P page
- Defines the structure of a MO:DCA-P page group
- Describes the resource objects that may be referenced in a MO:DCA-P document and defines their structure
- Describes how resource objects may be carried in resource groups
- Defines the structure of print control resource objects
- Describes the data objects that may be included in a MO:DCA-P document and defines their structure
- Defines the structure of object containers

Object Syntax Structure

This section specifies the syntax used to define MO:DCA-P objects.

If a structured field that is not identified as being part of the object appears anywhere within the object, a X'40' exception condition exists. If a structured field appears out of the stated order or more than the permitted number of times, a X'20' exception condition exists. If a structured field that is identified as required does not appear within the object, a X'08' exception condition exists.

The conventions used in these structured field groupings are:

( ) The structured field acronym and identifier are shown in parentheses. The presence of dots or periods in the identifier indicates that the item is not a structured field, but instead is a structure, for example a medium map. The structure is composed of an assortment of structured fields, and is defined separately.

[ ] Brackets indicate optional structured fields. When a structured field is shown without brackets, it *must* appear between the begin and end structured fields.

+ Plus signs indicate structured fields may appear in any order relative to those that precede or succeed it except when the preceding or succeeding structured field does not have a plus (+) sign. In that case, the order is as listed.

(S) The enclosed (S) indicates that the structured field may be repeated. When present on a required structured field, at least one occurrence of the structured field is required, but multiple instances of it may occur.

F2 An F2 indicates that the structured field is a format two structured field. See "Structured Field Formats" on page 25

Note: The No Operation structured field may appear within any begin-end domain. Therefore, it is not listed in the structured field groupings.

Printfile

The printfile is an object that contains one or more documents to be printed. A printfile may also optionally contain an external resource group, also referred to as a *printfile-level* resource group, as well as document indexes. Resources carried in a printfile-level resource group are sometimes referred to as *inline* resources.

```
     [  (      D3..C6)     Resource Group                             ]
        (      D3..A7/A8)  Index + Document                      (S)

 Index + Document Structure
     [  (      D3..A7)     Document Index                             ]
        (      D3..A8)     Document                              (S)
```

Figure 22. Printfile Structure

Figure 22 shows the interchange form of a MO:DCA-P printfile.

Warning: Any other form may cause inconsistent, presentation-system-dependent results.

For a definition of the Resource Group structure, see “Resource Groups” on page 78.

Notes:

1. The index, as shown in the Index + Document Structure, is optional. When specified, it must precede the document to be indexed and is implicitly tied to that document. Pointers from the index to the document and pointers from the document back to the index are not needed in this case and are ignored. That is, any FQN type X'83'—Begin Document triplet on the BDI is ignored, and any X'98'—Begin Document Index on the BDT is ignored.
2. Only a single resource group is permitted in a printfile. If multiple resource groups appear before the first document, or if one or more resource groups follow the first document, the treatment of these resource groups is presentation-system dependent.
3. A single document index before the inline resource group is accepted by AFP print servers and is implicitly tied to the first document in the printfile. However, this format is not compliant with the MO:DCA-P interchange printfile format and its use is discouraged.

Document

The document is the highest level object in the MO:DCA-P document component hierarchy. A document is delimited by Begin Document and End Document structured fields.

```
Begin Document (BDT, D3A8A8)
      [  (     D3..A7)     Document Index                            ]
  +   [  (IMM, D3ABCC)     Invoke Medium Map                    (S)  ]
  +   [  (IPG, D3AFAF)     Include Page                         (S)  ]
  +   [  (LLE, D3B490)     Link Logical Element                 (S)  ]
  +   [  (     D3..CC)     Medium Map                           (S)  ]

  +   [  (     D3..D9)     Resource Environment Group           (S)  ]
  +   [  (     D3..AF)     Page                                 (S)  ]
  +   [  (     D3..AD)     Page Group                           (S)  ]
End Docment (EDT, D3A9A8)
```

Figure 23. Document Structure

Figure 23 shows the general form of a MO:DCA-P document. MO:DCA-P interchange sets may specify a more restrictive document structure; however, such a structure must be a proper subset of the general form.

Notes:

1. 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 processing system’s form map.
2. A page that is included with an IPG in document state may be indexed using an offset to the location of the IPG in the document.
3. A Resource Environment Group (REG) maps *some* of the resources required to present the pages that follow. Resources mapped in a REG must still be mapped in the AEG for the page that uses the resources. The scope of the resource mapping in the REG is from the point where it occurs up to the next REG, which is a complete replacement for the current REG, or the end of the document, whichever occurs first.

Application Notes:

1. Internal (inline) medium maps are not supported by all AFP print servers. See the *Advanced Function Presentation: Programming Guide and Line Data Reference* for a specification of the MO:DCA structures and functions supported by AFP print servers.
2. The use of internal medium maps may significantly decrease document processing throughput, especially if the internal Medium Map specifies conditional media ejects using the Media Eject Control (X'45') triplet.
3. For optimum performance a REG is normally placed at the beginning of the document before the first page.

Document Index

A document index is an object that provides functions for indexing the document based on document structure and on application-defined document tags. A document index is delimited by Begin Document Index and End Document Index structured fields.

A document index is used for informational purposes only. Parameters in a document index are descriptive in nature and do not provide presentation specifications.

```
Begin Document Index  (BDI, D3A8A7)
  +      (IEL,   D3B2A7)   Index Element                           (S)
  +   [  (LLE,   D3B490)   Link Logical Element                    (S)  ]
  +   [  (TLE,   D3A090)   Tag Logical Element                     (S)  ]
End Document Index  (EDI, D3A9A7)
```

Figure 24. Document Index Structure

Resource Environment Group

A resource environment group (REG) is associated with a document or a group of pages in a document. It is contained in the document's begin-end envelope in the data stream. 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 is the pages that follow, up to the next REG, which is a complete replacement for the current REG, or the end of the document, whichever occurs first. The mapping of resources in a REG is optional. Resources mapped in a REG must still be mapped in the AEG for the page that uses the resources. When more than one REG is specified in a document, each REG is a complete replacement for the preceding REG.

```
Begin Resource Environment Group  (BSG, D3A8D9)
      [  (MDR,   D3ABC3)   Map Data Resource                      (S)  ]
      [  (MPO,   D3ABD8)   Map Page Overlay                       (S)  ]
      [  (PPO,   D3ADC3)   Preprocess Presentation Object         (S)  ]
End Resource Environment Group  (ESG, D3A9D9)
```

Figure 25. Resource Environment Group Structure

Notes:

1. When an MDR is specified in a REG, the FQN type X'BE' triplet, which specifies the internal identifier used to reference the resource being mapped, is ignored. An example of an internal identifier is the local ID used to reference a data-object font in a PTOCA object. The assignment of internal identifier to resource name is made when the MDR is specified in the environment group of the object that uses the resource. For example, in the case of a data-object font used in a PTOCA object, the internal identifier of the font is mapped to the font name in the AEG of the page. If the data-object font is used in an AFP GOCA object or a BCOCA object, the internal identifier of the font is mapped to the resource name in the OEG of the object.
2. There is no correlation between MPO Resource Local IDs (LIDs) in an AEG and MPO LIDs in an REG. For example, an MPO in an AEG can use LID x, and an MPO for the same overlay in a REG can use LID x or a different LID. The only

restriction is that regardless of where the MPO is specified, it is not permissible *within a given MPO* to map the same LID to more than one overlay.

3. An MDR reference to a specific resource may only be specified once in the REG.
4. Any object specified for preprocessing in a PPO must first be mapped in an MDR or an MPO in the same REG. This includes secondary resources that are specified in the PPO and that are required by the object to be preprocessed.

Application Note: For optimum performance a REG is normally placed at the beginning of the document before the first page.

Page

A page is an object that contains the data objects to be presented. A page establishes its own environment and is independent of any other page in the document. A page is delimited by Begin Page and End Page structured fields. A MO:DCA-P page object has the following syntax structure:

```
Begin Page  (BPG, D3A8AF)
      [  (      D3..C6)     Resource Group                              ]
         (      D3..C9)     Inactive Environment Group
  +   [  (IOB,  D3AFC3)     Include Object                        (S)   ]
  +   [  (IPG,  D3AFAF)     Include Page                                ]
  +   [  (IPO,  D3AFD8)     Include Page Overlay                  (S)   ]
  +   [  (IPS,  D3AF5F)     Include Page Segment                  (S)   ]
  +   [  (LLE,  D3B490)     Link Logical Element                  (S)   ]
  +   [  (TLE,  D3A090)     Tag Logical Element                   (S)   ]
  +   [  (      D3..EB)     Bar Code Object                       (S)   ]
  +   [  (      D3..BB)     Graphics Object                       (S)   ]
  +   [  (      D3..FB)     Image Object                          (S)   ]
  +   [  (      D3..92)     Object Container                      (S)   ]
  +   [  (      D3..9B)     Presentation Text Object              (S)   ]
End Page  (EPG, D3A9AF)

Active Environment Group (AEG)
Begin Active Environment Group  (BAG, D3A8C9)
      [  (MCF,  D3AB8A)     Map Coded Font                   F2   (S)   ]
      [  (MDR,  D3ABC3)     Map Data Resource                     (S)   ]
      [  (MPG,  D3ABAF)     Map Page                                    ]
      [  (MPO,  D3ABD8)     Map Page Overlay                      (S)   ]
      [  (MPS,  D3B15F)     Map Page Segment                      (S)   ]
         (PGD,  D3A6AF)     Page Descriptor
      [  (OBD,  D3A66B)     Object Area Descriptor                      ]
      [  (OBP,  D3AC6B)     Object Area Position                        ]
         (PTD,  D3B19B)     Presentation Text Data Descriptor   F2
End Active Environment Group  (EAG, D3A9C9)
```

Figure 26. Page Structure

Figure 26 shows the general form of a MO:DCA-P page object. MO:DCA-P interchange sets may specify a more restrictive page structure; however, such a structure must be a proper subset of the general form.

Notes:

1. The OBD and OBP structured fields in the AEG for the page are only used for presentation text objects and are optional.
2. The PTD structured field in the AEG for the page is only required when the page contains one or more presentation text objects. When the PTD is included in the AEG for a page, some AFP print servers require that the measurement

units in the PTD match the measurement units in the Page Descriptor (PGD). It is therefore strongly recommended that whenever the PTD is included in the AEG, the same measurement units are specified in both the PTD and PGD.

3. If a presentation text object specifies a coded font other than the presentation environment default font, the font local ID must be mapped to a font global name with an MCF or MDR structured field in the AEG for the page. This mapping must be unique, that is, the font local ID can only be mapped to one font in the AEG. However different font local IDs can be mapped to the same font. For rules on mapping local IDs (LIDs) to resource identifiers such as font global names, see “Environment Hierarchies” on page 28.
4. If an object container is included directly in a page, it must specify, at minimum, BOC/EOC, an OEG with OBD, OBP, CDD, and the object data must be carried in OCDs.
5. When an IPG structured field occurs in a page, the bit map for the referenced page is merged with the data defined for the current page. The referenced page must be mapped in the AEG for the current page and must not contain another IPG. Only a single IPG may occur within a page.
6. When an IPG occurs in a page, the included page becomes a part of the containing page, therefore only the containing page may be indexed using an offset to its location in the document.
7. For purposes of Print Services Facility resource management, each MDR that is specified in an object container OEG must have a corresponding MDR mapping the same resource in the AEG for that page. Note that an FQN type X'BE' triplet, if specified on the MDR in the OEG, is not factored up to the AEG, unless the MDR maps a data-object font.
8. An MDR reference to a specific resource may only be specified once in the AEG.
9. The resource group following BPG, which is also called an *internal* resource group or a *page-level* resource group, is not supported in AFP environments.

MO:DCA-P supports IM image objects on a page for migration purposes. One or more IM image objects may be included on a page in the same manner that IO image objects are included on a page. Both forms of image may coexist on the same page. For a definition of the IM image object, see Appendix C, “MO:DCA Migration Functions,” on page 501.

MO:DCA-P supports the Map Coded Font format-1 (MCF-1) structured field in the AEG for migration purposes. An MCF-1 may appear in place of an MCF format-2 (MCF-2) structured field. If both MCF-1 and MCF-2 structured fields are in the same environment group, the MCF-1 structured fields must precede the MCF-2 structured fields. For a definition of the MCF-1 structured field, see Appendix C, “MO:DCA Migration Functions,” on page 501.

Application Notes:

1. For purposes of Print Services Facility resource management, each MCFor MDR that maps a font in a data object OEG must have a corresponding MCF or MDR mapping the same font in the AEG for that page. The local ID used in the page AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object’s OEG.
2. For purposes of Print Services Facility resource management, each overlay included on a page with an IPO must first be mapped to a local ID with an MPO in the AEG for that page.

3. A page segment included on a page with an IPS may optionally be mapped with an MPS in the AEG for that page. If such a mapping exists, the page segment is sent to the presentation device as a separate object and is called a *hard* page segment. If such a mapping does not exist, the page segment is sent to the presentation device as part of the page and is called a *soft* page segment.

Page Group

A page group object is a named set of sequential pages in a document. All pages in a page group inherit the attributes and processing characteristics that are assigned to the page group. A page group is delimited by Begin Named Page Group and End Named Page Group structured fields.

```
Begin Named Page Group  (BNG, D3A8AD)
     [  (TLE,  D3A090)     Tag Logical Element                (S)  ]
  +  [  (IMM,  D3ABCC)     Invoke Medium Map                  (S)  ]
  +  [  (IPG,  D3AFAF)     Include Page                       (S)  ]
  +  [  (LLE,  D3B490)     Link Logical Element               (S)  ]
  +  [  (      D3..CC)     Medium Map                         (S)  ]

  +  [  (      D3..D9)     Resource Environment Group         (S)  ]
  +  [  (      D3..AF)     Page                               (S)  ]
  +  [  (      D3..AD)     Page Group                         (S)  ]
End Named Page Group  (ENG, D3A9AD)
```

Figure 27. Page Group Structure

Figure 27 shows the general form of a MO:DCA-P page group object. MO:DCA-P interchange sets may specify a more restrictive page group structure; however, such a structure must be a proper subset of the general form.

Notes:

1. 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 processing system's form map.
2. A page that is included with an IPG in page-group state may be indexed using an offset to the location of the IPG in the document.
3. A resource environment group (REG) maps *some* of the resources required to present the pages that follow. Resources mapped in a REG must still be mapped in the AEG for the page that uses the resources. The scope of the resource mapping in the REG is from the point where it occurs up to the next REG, which is a complete replacement for the current REG, or the end of the document, whichever occurs first.

Application Notes:

1. Internal (inline) medium maps are not supported by all AFP print servers. See the *Advanced Function Presentation: Programming Guide and Line Data Reference* for a specification of the MO:DCA structures and functions supported by AFP print servers.
2. The use of internal medium maps may significantly decrease document processing thruput, especially if the internal Medium Map specifies conditional media ejects using the Media Eject Control (X'45') triplet.
3. Page groups are often processed in stand-alone fashion; that is, they are indexed, retrieved, and presented outside the context of the containing

document. While the pages in the group are independent of each other and of any other pages in the document, their formatting on media depends on when the last medium map was invoked and on how many pages precede the BNG since this invocation. To make the media formatting of page groups self-contained, a Medium Map should be invoked at the beginning of the page group between the Begin Named Group (BNG) structured field and the first Begin Page (BPG) structured field. If this is not done, the presentation system may need to "play back" all pages between the invocation of the active medium map and the BNG to determine media formatting such as sheet-side and partition number for the first page in the group.

It is therefore *strongly* recommended that in environments where stand-alone page group processing is required or anticipated, page groups are built with an Invoke Medium Map (IMM) structured field specified after the BNG and before the first BPG. IBM AFP applications that generate page groups will support a user option which ensures that an IMM is specified after BNG and before the first BPG, and IBM AFP archive servers will expect an IMM there and may not present the page group correctly if none is found. However, note that this may cause the complete document to print differently.

A newer method to specify how a page or page group should be formatted involves use of the Page Position Information (X'81') triplet. This triplet may be specified on a BPG and indicates the repeating group in the PGP structured field in the active medium map that should be used to format the page.

4. For optimum performance a REG is normally placed at the beginning of the document before the first page.

Resource Objects

Objects are considered to be resource objects when they are explicitly referenced from the document instead of being directly included in the document. Resource objects may reside in external resource libraries, or in resource groups external to the document, or in resource groups internal to the document. Note that data objects such as IOCA image objects and object containers become resource objects when included with an Include Object (IOB) structured field.

Architecture Note: Any presentation object, other than an overlay, when processed as a resource, must not contain font mappings defined with Map Coded Font (MCF) structured fields in the object environment group. A presentation object is processed as a resource when it is mapped using a Map structured field and included using an Include stuctured field.

Font Objects

A font is a collection of graphic characters with the same type family, typeface, and size. Fonts are referenced by MO:DCA-P documents for presenting text.

Font Object Content Architecture (FOCA) Fonts

The IBM Font Object Content Architecture (FOCA) defines a font format that is supported in MO:DCA-P documents. Suche fonts are referenced in the data stream using an MCF structured field. This font format defines three types of font objects:

- Coded font objects
- Code page objects
- Font character set objects

Each object is bounded by begin and end structured fields that are registered as private structured fields in the MO:DCA architecture. The content of each object is carried in structured fields that are also registered as private structured fields in the MO:DCA architecture. For a description of these objects and their structured fields, see the *Advanced Function Presentation: Host Font Data Stream Reference*.

TrueType/OpenType Fonts

TrueType and OpenType fonts are non-FOCA fonts, also called data-object fonts, that are supported in MO:DCA-P documents. They are referenced in the data stream using an MDR structured field. They can be installed in a resource library in their native, unaltered format, or they can be carried in a printfile-level resource group in an object container. Collections of TrueType or OpenType fonts, called TrueType Collections, are also supported.

The TrueType font format is based on scalable outline technology with flexible hinting. Mathematically, TrueType shapes are based on quadratic curves; this is in contrast to Adobe Type 1 outlines which are based on cubic curves. TrueType is an open font standard and is widely published. The technology is described in the following documents available from the Microsoft and Apple web sites:

- *TrueType Font Files Technical Specification* (Microsoft Corporation)
- *TrueType Reference Manual* (Apple Computer, Inc.)

The OpenType font format is an extension of the TrueType font format that allows better support for international character sets and broader multi-platform support. OpenType defines tables that can be used to carry the formatting information needed to fully support Unicode. Additionally, this format allows either TrueType or Adobe Type 1 outlines to be packaged as an OpenType font. The OpenType font format was developed jointly by the Adobe and Microsoft Corporations, and it is described in the *OpenType Specification*, which is available on the Microsoft web site.

Overlay Objects

An overlay is a MO:DCA-P resource object. It may be stored in an external resource library or it may be carried in a resource group. An overlay is similar to a page in that it defines its own environment and carries the same data objects.

```
Begin Overlay  (BMO, D3A8DF)
         (      D3..C9)     Active Environment Group
  +   [  (LLE,  D3B490)     Link Logical Element                 (S)  ]
  +   [  (TLE,  D3A090)     Tag Logical Element                  (S)  ]
  +   [  (      D3..EB)     Bar Code Object                      (S)  ]
  +   [  (      D3..BB)     Graphics Object                      (S)  ]
  +   [  (      D3..FB)     Image Object                         (S)  ]
  +   [  (      D3..9B)     Presentation Text Object             (S)  ]
  +   [  (      D3..92)     Object Container                     (S)  ]
  +   [  (IOB,  D3AFC3)     Include Object                       (S)  ]
  +   [  (IPS,  D3AF5F)     Include Page Segment                 (S)  ]
End Overlay  (EMO, D3A9DF)

Active Environment Group (AEG)
Begin Active Environment Group  (BAG, D3A8C9)
      [  (MCF,  D3AB8A)     Map Coded Font                  F2   (S)  ]
      [  (MDR,  D3ABC3)     Map Data Resource                    (S)  ]
      [  (MPS,  D3B15F)     Map Page Segment                     (S)  ]
         (PGD,  D3A6AF)     Page Descriptor
      [  (OBD,  D3A66B)     Object Area Descriptor                    ]
      [  (OBP,  D3AC68)     Object Area Position                      ]
         (PTD,  D3B19B)     Presentation Text Data Descriptor   F2
End Active Environment Group  (EAG, D3A9C9)
```

Figure 28. Overlay Structure

Figure 28 shows the general form of a MO:DCA-P overlay object. MO:DCA-P interchange sets may specify a more restrictive overlay structure; however, such a structure must be a proper subset of the general form.

Notes:

1. The OBD and OBP structured fields in the AEG for the overlay are only used for presentation text objects and are optional.
2. The PTD structured field in the AEG for the overlay is only required when the overlay contains one or more presentation text objects. When the PTD is included in the AEG for an overlay, some AFP print servers require that the measurement units in the PTD match the measurement units in the Page Descriptor (PGD). It is therefore strongly recommended that whenever the PTD is included in the AEG, the same measurement units are specified in both the PTD and PGD.
3. If a presentation text object specifies a coded font other than the presentation environment default font, the font local ID must be mapped to a font global name with an MCF or MDR structured field in the AEG for the overlay. This mapping must be unique, that is, the font local ID can only be mapped to one font in the AEG. However different font local IDs can be mapped to the same font. For rules on mapping local IDs (LIDs) to resource identifiers such as font global names, see “Environment Hierarchies” on page 28.
4. If an object container is included directly in an overlay, it must specify, at minimum, BOC/EOC, an OEG with OBD, OBP, CDD, and the object data must be carried in OCDs. See “Object Containers” on page 100 for a complete definition of the object container structure.
5. For purposes of Print Services Facility resource management, each MDR that is specified in an object container OEG must have a corresponding MDR mapping the same resource in the AEG for that overlay. Note that an FQN type X'BE' triplet, if specified on the MDR in the OEG, is not factored up to the AEG, unless the MDR maps a data-object font.
6. An MDR reference to a specific resource may only be specified once in the AEG.

MO:DCA-P supports IM image objects on an overlay for migration purposes. One or more IM image objects may be included on an overlay in the same manner that IO image objects are included on an overlay. Both forms of image may coexist on the same overlay. For a definition of the IM image object, see Appendix C, “MO:DCA Migration Functions,” on page 501.

MO:DCA-P supports the Map Coded Font format-1 (MCF-1) structured field in the AEG for migration purposes. An MCF-1 may appear in place of an MCF format-2 (MCF-2) structured field. If both MCF-1 and MCF-2 structured fields are in the same environment group, the MCF-1 structured fields must precede the MCF-2 structured fields. For a definition of the MCF-1 structured field, see Appendix C, “MO:DCA Migration Functions,” on page 501.

Application Notes:

1. For purposes of Print Services Facility resource management, each MCF or MDR that maps a font in a data object OEG must have a corresponding MCF or MDR mapping the same font in the AEG for that overlay. The local ID used in the overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object’s OEG.
2. A page segment included on an overlay with an IPS may optionally be mapped with an MPS in the AEG for that overlay. If such a mapping exists, the page segment is sent to the presentation device as a separate object and is called a *hard* page segment. If such a mapping does not exist, the page segment is sent to the presentation device as part of the overlay and is called a *soft* page segment.

Page Segment Objects

A page segment is a MO:DCA-P resource object. It may be stored in an external resource library or it may be carried in a resource group. Page segments contain any combination of the following data objects:

- Image objects in IOCA format
- Graphics objects in GOCA format
- Bar code objects in BCOCA format

A page segment does not define a presentation space and has no coordinate system, therefore objects cannot be positioned relative to each other within a page segment. Instead, all objects in a page segment must specify an object area offset of zero. Objects within the page segment may be positioned on the including page or overlay at a reference point specified by the IPS structured field, or they may be positioned at the including page or overlay origin. This positioning is specified by the Reference Coordinate System parameter in the object’s Object Area Position (OBP) structured field.

A page segment resource object does not contain an active environment group and therefore does not define its own environment. Instead, the page segment assumes the environment definition of the including page or overlay.

```
Begin Page Segment  (BPS, D3A85F)
  +   [  (      D3..EB)     Bar Code Object                          (S)  ]
  +   [  (      D3..BB)     Graphics Object                          (S)  ]
  +   [  (      D3..FB)     Image Object                             (S)  ]
End Page Segment  (EPS, D3A95F)
```

Figure 29. Page Segment Structure

MO:DCA-P supports the AFP Page Segment object for migration purposes. For a definition of this object, see “AFP Page Segment” on page 519.

Application Note: For purposes of PSF resource management, the OEGs for all objects in a page segment must not contain MCF or MDR structured fields when the page segment is referenced with an IOB or IPS structured field.

Resource Groups

A resource group is an object that contains a collection of resource objects, including:

- Overlays
- Page segments
- Form maps
- Referenced data objects
- Object containers

Resource groups in MO:DCA–P documents may be located inside the document, in which case they are called *internal or page-level* resource groups, or outside the document in a printfile, in which case they are called *external or printfile-level* resource groups. Resources that are carried in resource groups are said to be *inline*. A resource group is delimited by Begin Resource Group and End Resource Group structured fields.

The scope of a resource group is the object or component that contains the resource group. That is, the resources within the resource group are available for use by the presentation system only for the duration of the containing object or component. For example, when a resource group is specified outside the document as part of a printfile, that is, when it is specified as an external resource group, the resources within the group are available only for the duration of the printfile. Once the last document in the printfile has been processed, these resources are no longer available to the presentation system for use with another printfile.

The general search order for MO:DCA resources is as follows:

1. Internal (page-level) resource groups
2. External (printfile-level) resource groups
3. External resource libraries

Within a resource group, resource objects of the same type must have unique identifiers; if they do not, the treatment of such resources is presentation-system dependent.

```
Begin Resource Group  (BRG, D3A8C6)
  +   [  (      D3..DF)     Overlay                           (S)  ]
End Resource Group  (ERG, D3A9C6)
```

Figure 30. Internal (Page-level) Resource Group Structure

```
Begin Resource Group  (BRG, D3A8C6)
  +   [  (      D3..DF)     Overlay                                  (S)  ]
  +   [  (      D3..5F)     Page Segment                             (S)  ]
  +   [  (      D3..CD)     Form Map                                 (S)  ]
  +   [  (      D3..EB)     Bar Code Object                          (S)  ]
  +   [  (      D3..BB)     Graphics Object                          (S)  ]
  +   [  (      D3..FB)     Image Object                             (S)  ]

  +   [  (      D3..92)     Object Container                         (S)  ]
      [  (      D3..A8)     Document                                 (S)  ]
End Resource Group  (ERG, D3A9C6)
```

Figure 31. External (Printfile-level) Resource Group Structure

Notes:

1. In AFP environments, resources carried in external resource groups are called *inline* resources.
2. If an object container is included in a resource group, it must at a minimum be bounded by a BOC/EOC pair, an Object Classification (X'10') triplet must be specified on the BOC with a registered object-type identifier (object-type OID) for the object data, and the data must be carried in OCDs.
3. Within a resource group, resource objects of the same type must have unique identifiers.
4. Documents are carried as resource objects in a resource group so that pages in these documents can be processed and saved in the presentation device for fast subsequent retrieval using Include Page (IPG) structured fields.
5. The resource group following BPG, which is also called an *internal* resource group or a *page-level* resource group, is not supported in AFP environments.

Each resource object in an external resource group may be wrapped with an optional Begin Resource (BRS) and End Resource (ERS) envelope as shown in Figure 32.

```
      [  (BRS,  D3A8CE)     Begin Resource                                ]
         (      D3..xx)     Resource Object
      [  (ERS,  D3A9CE)     End Resource                                  ]
```

Figure 32. BRS/ERS Envelope for Resources in External (Printfile-level) Resource Group

The BRS and ERS structured fields must be specified as a pair, that is, one may not be specified without the other.

Notes:

1. The BRS/ERS envelope is mandatory for resources carried in an external resource group in AFP presentation environments.
2. In AFP environments, the following objects are also included in external resource groups:
   - Page maps (also called page definitions or pagedefs)
   - Font objects
     - Coded fonts
     - Code pages
     - Font character sets

   For a description of page maps, see the *Advanced Function Presentation: Programming Guide and Line Data Reference*. For a description of font objects, see the *Advanced Function Presentation: Host Font Data Stream Reference*.

External Resource 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. The length of the name is determined by the length of the triplet, and all bytes in the triplet are considered to be part of the name.

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. Examples of such cases are the Code Page, Font Character Set, and Coded Font names carried in the FQN type X'85', X'86', and X'8E' triplets, respectively, which define a fixed EBCDIC 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 and its scope in the document hierarchy, see “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317. The X'01' triplet is mandatory on the Begin Document (BDT) structured field and 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.

Application Note: In AFP environments, print Servers treat an external object name - other than a TrueType or OpenType full font name - as a 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 or 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 use only the recommended characters and are 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'F0'–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.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same. For TrueType/OpenType fonts, optimal performance can be achieved by using UTF-16BE as the encoding scheme.

Print Control Objects

Print control objects are resource objects that are used to control the presentation of pages on physical media, also known as forms or sheets, in a printer. There are two types of print control objects, *form maps*, also known as *form definitions* or *formdefs*, and *medium maps*.

Form Map

A form map is a print control resource object that consists of:

- An optional document environment group (DEG) that defines the print environment for the form map
- One or more medium map resource objects that are invokable on document and page boundaries and that specify a complete set of print controls. The name assigned to each medium map object is unique within the form map.

A form map is selected for controlling document presentation when the document print request is generated.

The scope of a form map is a document, and control for presentation starts with the first medium map in the form map. If the form map is associated with a printfile that contains multiple documents, the scope of the form map is the printfile, and control for presentation is returned to the first medium map in the form map whenever a new document is encountered.

```
Begin Form Map  (BFM, D3A8CD)
      [  (      D3..C4)     Document Environment Group                   ]
         (      D3..CC)     Medium Map                              (S)
End Form Map  (EFM, D3A9CD)
```

Figure 33. Form Map Structure

Document Environment Group

The document environment group (DEG), when present, establishes the presentation environment for a form map resource object. This presentation environment consist of the following:

- A definition of the medium presentation space, including units of measure, size, and orientation
- The default position of the logical page on the medium presentation space
- A mapping of overlay local IDs, as specified in a medium map in the form map, to overlay names
- A mapping of text suppression local IDs, as specified in a medium map in the form map, to text suppression names
- A specification of the fidelity that is required for presentation
- A specification of finishing operations that are to be applied to media.

If a parameter is specified in the DEG as well as in a medium map, the specification in the medium map takes precedence.

Note: When an internal (inline) medium map is used, structured fields which can be specified in the DEG and/or in a medium map, specifically the MDD, MMO, and PGP, must be specified in the internal medium map. If they are specified in the Document Environment Group (DEG), they do not apply to internal medium maps. Structured fields which can only be specified in the DEG and not in a medium map, such as the MSU, and PFC, apply to the complete document or printfile and are independent of internal medium maps and medium maps in the form map. The MFC structured field can be specified in the DEG and/or a Medium Map and defines its scope explicitly.

```
Begin Document Environment Group  (BDG, D3A8C4)
      [  (PFC, D38288)     Presentation Fidelity Control           (S)  ]
      [  (MMO, D3B1DF)     Map Medium Overlay                           ]
      [  (MSU, D3ABEA)     Map Suppression                              ]
         (PGP, D381AF)     Page Position                      F2
         (MDD, D3A688)     Medium Descriptor
      [  (MFC, D3A088)     Medium Finishing Control                (S)  ]
End Document Environment Group  (EDG, D3A9C4)
```

Figure 34. Document Environment Group Structure

Notes:

1. An MMO is required in either the document environment group or a medium map if an MMC structured field references a medium overlay. If specified in both, the structured field in the medium map takes precedence.
2. A PGP and an MDD is required in either the document environment group or a medium map. If specified in both, the structured field in the medium map takes precedence.

3. The DEG may contain one printfile-level MFC that applies to the complete printfile, one document-level MFC that applies to all documents in the printfile, and one or more document-level MFCs that apply to single documents in the printfile. In the last case, only one MFC in the DEG may select a given document in the printfile. If the DEG contains more than one printfile-level MFC, more than one document-level MFC that applies to all documents, or more than one document-level MFC that selects the same document, only the last-specified MFC having that particular scope is used.

Medium Map

A medium map is a print control resource object that contains the print control parameters for presenting pages on a physical medium and for generating copies of the physical medium. Print control parameters may be grouped into two categories:

- Medium-level controls
- Page-level controls

Medium-level controls are controls that affect the medium, such as the specification of medium overlays, medium size, medium orientation, medium copies, simplex or duplex, medium finishing, and media source and destination selection. These controls are defined by the Map Medium Overlay (MMO), Medium Descriptor (MDD), Medium Copy Count (MCC), Medium Finishing Control (MFC), Map Media Type (MMT), and Medium Modification Control (MMC) structured fields. Page-level controls are controls that affect the pages that are placed on the medium, such as the specification of page modifications, page position, and page orientation. These controls are defined by the Map Page Overlay (MPO), Page Position (PGP), and Page Modification Control (PMC) structured fields. When N-up partitioning is specified, the Media Eject Control (X'45') triplet may be included on the Begin Medium Map structured field to specify the type of media eject that is performed and the type of controls that are activated when the medium map is invoked.

A medium map contains one Medium Copy Count (MCC) structured field that generates a *copy group* for each sheet, therefore a medium map is also sometimes referred to as a copy group. Each MCC contains one or more *copy subgroups* that specify the number of copies of a sheet to be generated for the copy subgroup and the modifications to be applied to all copies in the copy subgroup. The modifications are specified by a Medium Modification Control (MMC) structured field. If the modifications for a copy subgroup specify duplexing, that copy subgroup and all successive copy subgroups are paired such that the first copy subgroup in the pair specifies the copy count as well as the modifications to be applied to the front side of each copy, and the second copy subgroup in the pair specifies the same copy count as well as an independent set of modifications to be applied to the back side of each copy. The pairing of copy subgroups continues as long as duplexing is specified. Note that with simplex printing, each copy subgroup builds the front sheet-side on all sheet copies generated by the copy subgroup. With duplex printing, the first and second copy subgroup in each pair of copy subgroups build front and back sheet-sides, respectively, on all sheet copies generated by the pair of copy subgroups. Figure 35 on page 84 illustrates the copy subgroup concept.

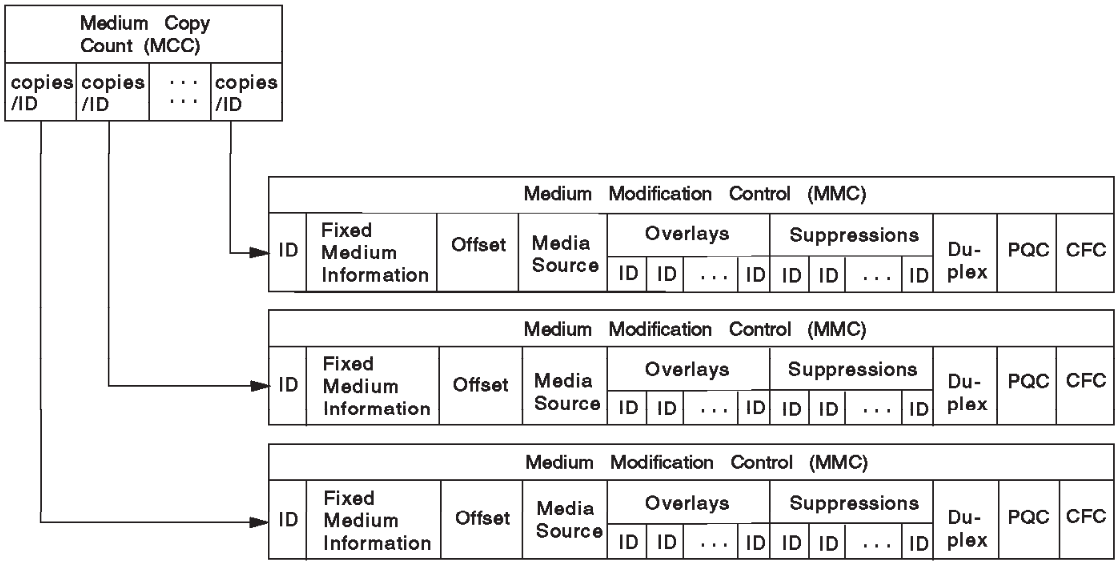


Figure 35. Copy Subgroups

Invocation of Medium Maps

- A medium map can be invoked by name on any page boundary in a document. This is done by including an IMM (Invoke Medium Map) structured field in the document data stream. Multiple IMMs may be used within a single document.
- A medium map can be directly included on any page boundary in the document data stream. Such a medium map is called an *internal* medium map. Multiple internal medium maps may be included in a document. An internal medium map is activated by following it immediately with an IMM that invokes the internal medium map. If an internal medium map is not explicitly invoked with an immediately-following IMM, it is ignored. IMMs cannot be used to invoke internal medium maps elsewhere in the document. When an IMM does not follow and reference an internal medium map, it references an external medium map in the processing system's form map.

  The name assigned to each internal medium map object is unique within the document.

  Note: When an internal (inline) medium map is used, structured fields which can be specified in the DEG and/or in a medium map, specifically the MDD, MMO, and PGP, must be specified in the internal medium map. If they are specified in the document environment group (DEG), they do not apply to internal medium maps. Structured fields which can only be specified in the DEG and not in a medium map, such as the MSU, and PFC, apply to the complete document or printfile and are independent of internal medium maps and medium maps in the form map. The MFC can be specified in a DEG and/or a medium map and defines its scope explicitly.

  Application Notes:

  1. Internal (inline) medium maps are not supported by all AFP print servers. See the *Advanced Function Presentation: Programming Guide and Line Data Reference* for a specification of the MO:DCA structures and functions supported by AFP print servers.

2. The use of internal medium maps may significantly decrease document processing throughput, especially if the internal medium map specifies conditional media ejects using the Media Eject Control (X'45') triplet.
3. Internal medium maps are also sometimes referred to as *inline* medium maps. The term “internal” is preferred.

- A medium map remains in effect until another medium map is selected or the end of the document is reached.
- If a document does not invoke a medium map by name, and if it does not include an internal medium map, the first medium map in the selected form map controls the printing.
- When an invoked medium map is used to process medium overlays or variable page data, it causes a media eject to occur before any data is presented. If not explicitly specified otherwise, the eject is to a new physical medium (form). When N-up partitioning is specified, the Media Eject Control (X'45') triplet may be included on the Begin Medium Map structured field to specify one of the following partition ejects:
  - Conditional eject to next partition
  - Conditional eject to next front-side partition
  - Conditional eject to next back-side partition

  However, this triplet is ignored when it occurs on the medium map that is activated at the beginning of a document regardless of whether this medium map is explicitly invoked or implicitly invoked as the default.
- If a contiguous sequence of IMMs is specified in the data stream, they are processed according to the following rules:
  - If the sequence of IMMs is followed by a page, the last IMM must invoke a medium map that allows the presentation of pages. If it does not, an exception is generated.
  - If the sequence of IMMs is followed by a page, only the last invoked medium map is used for processing; preceding medium maps are ignored. For example, if the first invoked medium map specifies a conditional eject to the next front partition and the last invoked medium map specifies a conditional eject to the next partition, the page is placed into the next partition. Similarly, if the first invoked medium map specifies “constant front” but allows page placement on the back, and if the last invoked medium map specifies “constant back” but allows page placement on the front, the first invoked medium map is ignored and the page is placed on the front, with constant data placed on the back.
  - If the sequence of IMMs invoke medium maps that prohibit the presentation of pages but that present medium overlays or PMC overlays, each medium map generates a sheet or multiple copies of a sheet with constant overlay data, as specified. These sheets are generated whether the last IMM is followed by a page or not.

Application Note: Page groups are often processed in stand-alone fashion, that is, they are indexed, retrieved, and presented outside the context of the containing document. While the pages in the group are independent of each other and of any other pages in the document, their formatting on media depends on when the last medium map was invoked and on how many pages precede the BNG since this invocation. To make the media formatting of page groups self-contained, a Medium Map should be invoked at the beginning of the page group between the Begin Named Group (BNG) structured field and the first Begin Page (BPG) structured field. If this is not done, the presentation system may need to “play back” all pages between the invocation of the active medium map and the BNG to determine media formatting such as sheet-side and partition number for the first page in the group.

It is therefore *strongly* recommended that in environments where stand-alone page group processing is required or anticipated, page groups are built with an Invoke Medium Map (IMM) structured field specified after the BNG and before the first BPG. IBM AFP applications that generate page groups will support a user option which ensures that an IMM is specified after BNG and before the first BPG, and IBM AFP archive servers will expect an IMM there and may not present the page group correctly if none is found. However, note that this may cause the complete document to print differently.

A newer method to specify how a page or page group should be formatted involves use of the Page Position Information (X'81') triplet. This triplet may be specified on a BPG and indicates the repeating group in the PGP structured field in the active Medium Map that should be used to format the page.

```
 Begin Medium Map  (BMM, D3A8CC)
       [  (MMO, D3B1DF)     Map Medium Overlay                          ]
       [  (MPO, D3ABD8)     Map Page Overlay                       (S)  ]
       [  (MMT, D3AB88)     Map Media Type                         (S)  ]
          (PGP, D3B1AF)     Page Position                     F2
          (MDD, D3A688)     Medium Descriptor
          (MCC, D3A288)     Medium Copy Count
       [  (MMC, D3A788)     Medium Modification Control            (S)  ]
       [  (PMC, D3A7AF)     Page Modification Control              (S)  ]
|      [  (MFC, D3A088)     Medium Finishing Control               (S)  ]
 End Medium Map  (EMM, D3A9CC)
```

Figure 36. Medium Map Structure

Notes:

1. An MMO is required in either the document environment group or a medium map if an MMC structured field references a medium overlay. If specified in both, the structured field in the medium map takes precedence.
2. Within a medium map, a given media type local ID may only be mapped once to a media type OID and/or a media type name using an MMT.
3. A PGP and an MDD is required in either the document environment group or a medium map. If specified in both, the structured field in the medium map takes precedence.
4. MMC identifiers must be unique for all MMC structured fields in the medium map. PMC identifiers must be unique for all PMC structured fields in the medium map.
5. Each overlay included on a page with a PMC must first be mapped to a local ID with an MPO in the medium map containing the PMC.
6. Modifications specified by PMC structured fields are applied to pages on the medium dependent on the MMC N-up Format Control (X'FC') keyword as follows:

- If N-up is not specified, the page on each sheet-side is processed with the PGP repeating group for that sheet side. All modifications specified by all PMCs in the active medium map are applied to the page on the sheet-side.
- If N-up with default page placement is specified, all pages on a sheet-side are processed with the PGP repeating group for that sheet side. If this repeating group does not specify a PMC identifier, or if the PMC identifier specifies X'FF', all modifications specified by all PMCs in the active medium map are applied to each page on the sheet side. If this repeating group specifies a PMC identifier, only the modifications included by the selected PMC are applied to all pages on the sheet-side.
- If N-up with explicit page placement is specified, each page is processed with a PGP repeating group. If this repeating group does not specify a PMC identifier, or if the PMC identifier specifies X'FF', all modifications specified by all PMCs in the active medium map are applied to the page. If this repeating group specifies a PMC identifier, only the modifications included by the selected PMC are applied to the page.

7. The actual presentation of the selected PMC modifications is controlled by the MMC Constant Forms Control (X'F9') keyword and the PGP PgFlgs parameter. See “Page Position (PGP) Format 2” on page 282.
8. All overlays included with a PMC structured field are presented on the page presentation space *before* any variable page data is presented.
9. MFCs can be specified in the document environment group, in a medium map, or in both places. When specified in both places, all specified finishing operations are applied according to their scope, as long as the operations are compatible. Note that the location of the MFC may restrict which operations are supported. For rules on how finishing operations are nested, see “Finishing Operation Nesting Rules” on page 242.

Data Objects

Data objects contain presentation data and the controls to present this data. Data objects are generated in an object presentation space in accordance with controls defined by the data object architecture. The object presentation space is mapped to an object area on the page in accordance with controls defined in MO:DCA environment groups. Data object mappings are shown in the specific object descriptions that follow. Object area positioning is shown in Figure 37 on page 88.

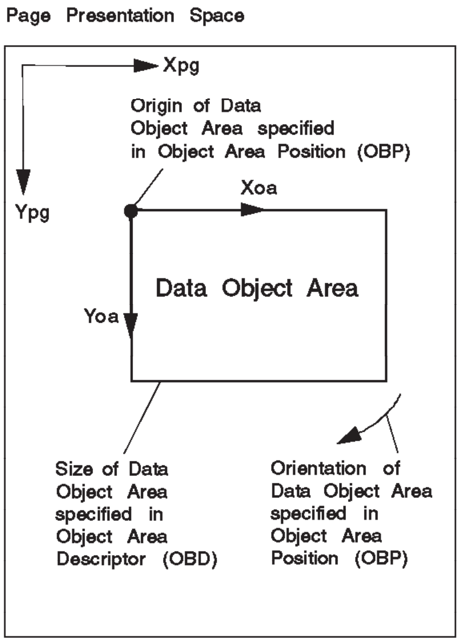

Page Presentation Space
Xpg
Origin of Data Object Area specified in Object Area Position (OBP)
Xoa
Ypg
Data Object Area
Yoa
Size of Data Object Area specified in Object Area Descriptor (OBD)
Orientation of Data Object Area specified in Object Area Position (OBP)

Figure 37. Object Area Positioning on a Page

Data objects are defined for the following types of presentation data: text, image, graphics, bar codes. The corresponding data object architectures may define various functional levels for the data objects. When such levels are formally defined, they are called *function sets* or *subsets*. Wherever support for a data object in MO:DCA-P is limited to particular function sets, the function-set level is indicated in the object structure definition. Wherever a MO:DCA-P interchange set further restricts the level of function set that is supported in the interchange set, such restriction is indicated in the interchange set definition.

Bar Code Objects

Bar code data consists of patterns of bars and spaces that represent alphanumeric information. Characteristics of the patterns are defined by specific bar code symbologies. A bar code object carries the alphanumeric information that is to be presented as a bar code and the controls to present this information using a specific bar code symbology. The bar code data object is defined by the Bar Code Object Content Architecture.

```
Begin Bar Code Object  (BBC, D3A8EB)
         (     D3..C7)     Object Environment Group
      [  (BDA, D3EEEB)     Bar Code Data                          (S)  ]
End Bar Code Object  (EBC, D3A9EB)

Object Environment Group (OEG) for Bar Code Object
Begin Object Environment Group  (BOG, D3A8C7)
         (OBD, D3A66B)     Object Area Descriptor
         (OBP, D3AC6B)     Object Area Position
      [  (MBC, D3ABEB)     Map Bar Code Object                         ]
      [  (MCF, D3AB8A)     Map Coded Font                     F2  (S)  ]
      [  (MDR, D3ABC3)     Map Data Resource                      (S)  ]
         (BDD, D3A6EB)     Bar Code Data Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 38. Bar Code Object (BCOCA BCD1 Level) Structure

Application Note:

1. For purposes of Print Services Facility resource management, each MCF that maps a font in the bar code OEG must have a corresponding MCF mapping the same font in the AEG for the page or overlay that includes the bar code object. The local ID used in the page or overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG.
2. An MDR is used to map a non-FOCA data-object font resource in a bar code object. For purposes of Print Services Facility resource management, each MDR that is maps a font in the bar code OEG must have a corresponding MDR mapping the same font resource and attributes in the AEG for the page or overlay that includes the bar code object. The local ID used in the page or overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG.

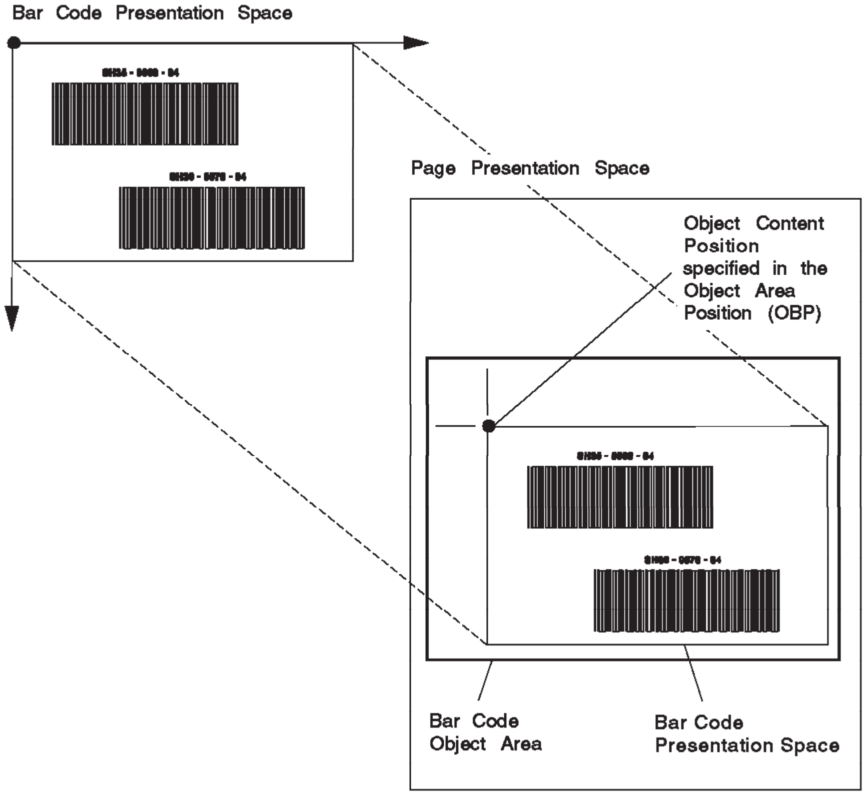

Bar Code Presentation Space
Page Presentation Space
Object Content Position specified in the Object Area Position (OBP)
Bar Code Object Area
Bar Code Presentation Space

Figure 39. Bar Code Presentation Space Mapping: Position

Note: Refer to the *Bar Code Object Content Architecture Reference* for a full description of the BCOCA object content, syntax, and semantics for MO:DCA-P.

Mapping the Bar Code Presentation Space

The mapping option is specified by the Mapping Option (X'04') triplet on the Map Bar Code Object (MBC) structured field. The only valid option is *position*. This mapping is shown in Figure 39.

Graphics Objects

Graphics data consists of controls and parameters to generate pictures based on lines, characters, and shaded areas. The graphics data object is defined by the Graphics Object Content Architecture for Advanced Function Presentation.

```
Begin Graphics Object  (BGR, D3A8BB)
         (     D3..C7)     Object Environment Group
      [  (GAD, D3EEBB)     Graphics Data                          (S)  ]
End Graphics Object  (EGR, D3A9BB)

Object Environment Group (OEG) for Graphics Object
Begin Object Environment Group  (BOG, D3A8C7)
         (OBD, D3A66B)     Object Area Descriptor
         (OBP, D3AC6B)     Object Area Position
      [  (MGO, D3ABBB)     Map Graphics Object                         ]
      [  (MCF, D3AB8A)     Map Coded Font                     F2  (S)  ]
      [  (MDR, D3ABC3)     Map Data Resource                      (S)  ]
         (GDD, D3A6BB)     Graphics Data Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 40. Graphics Object (GOCA DR/2V0 Level) Structure

Application Note:

1. For purposes of Print Services Facility resource management, each MCF that maps a font in the graphics OEG must have a corresponding MCF mapping the same font in the AEG for the page or overlay that includes the graphics object. The local ID used in the page or overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG.
2. An MDR is used to map a non-FOCA data-object font resource in a graphics object. For purposes of Print Services Facility resource management, each MDR that maps a font in the graphics OEG must have a corresponding MDR mapping the same font resource and attributes in the AEG for the page or overlay that includes the graphics object. The local ID used in the page or overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG.

Note: Refer to the *Graphics Object Content Architecture for Advanced Function Presentation Reference* for a full description of the GOCA object content, syntax, and semantics for MO:DCA-P.

Mapping the Graphics Presentation Space

The mapping option is specified by the Mapping Option (X'04') triplet on the Map Graphics Object (MGO) structured field. The valid mapping options are:
- Scale to fit
- Scale to fill
- Center and trim
- Position and trim

The replicate-and-trim mapping option has been retired for graphics objects; see "Retired Parameters" on page 515.

These mapping options are shown in Figure 41 on page 91, Figure 42 on page 92, Figure 43 on page 93, and Figure 44 on page 94.

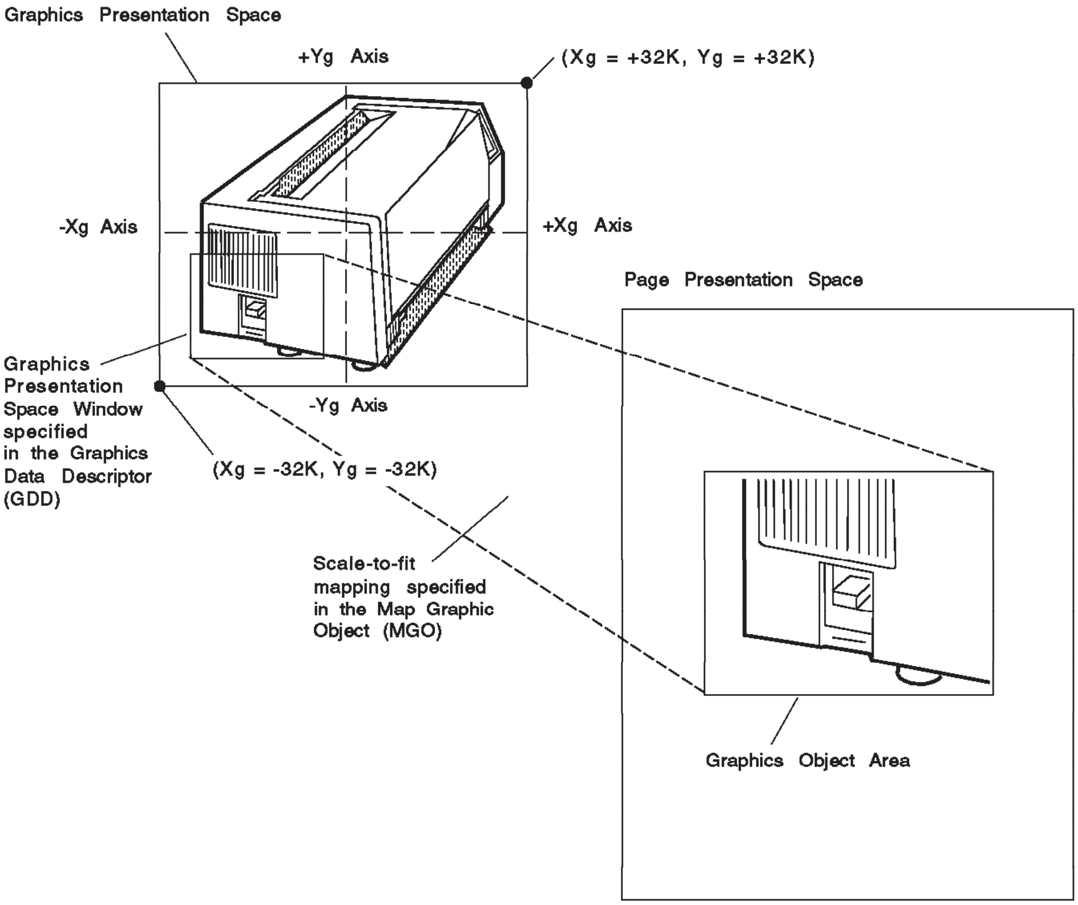

Graphics Presentation Space
+Yg Axis
(Xg = +32K, Yg = +32K)
-Xg Axis
+Xg Axis
Page Presentation Space
Graphics Presentation Space Window specified in the Graphics Data Descriptor (GDD)
-Yg Axis
(Xg = -32K, Yg = -32K)
Scale-to-fit mapping specified in the Map Graphic Object (MGO)
Graphics Object Area

Figure 41. Graphics Presentation Space Mapping: Scale to Fit

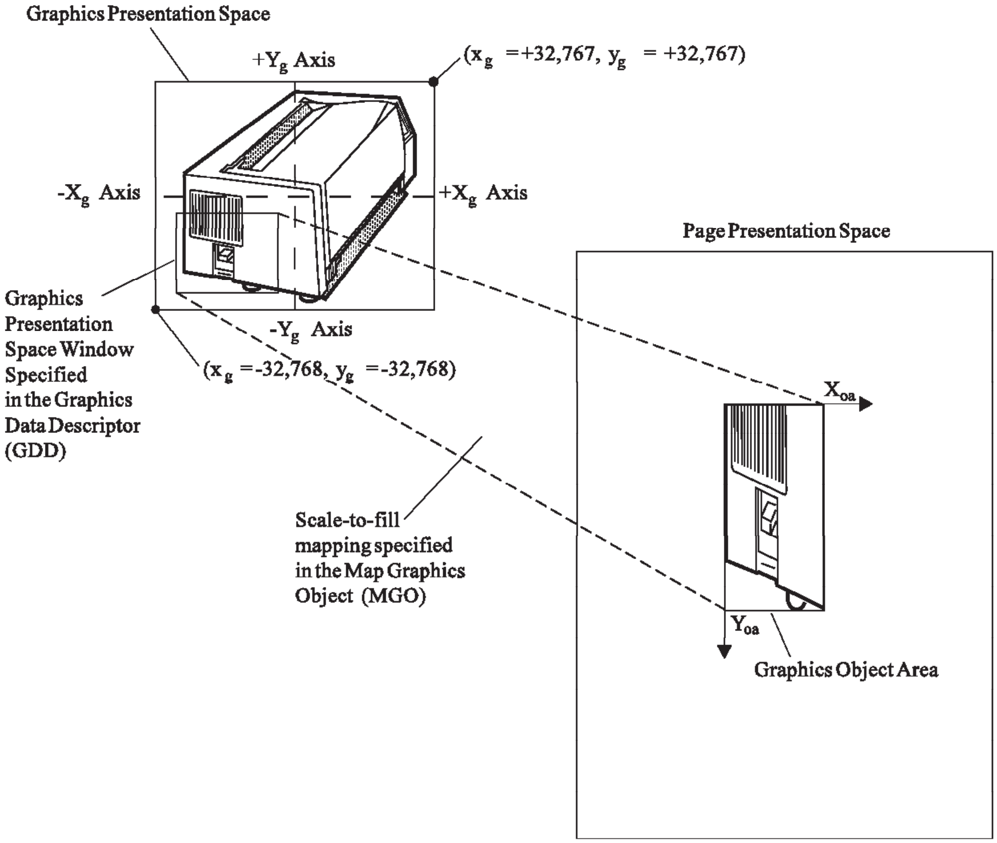

Graphics Presentation Space
+Y$_g$ Axis
(x$_g$ =+32,767, y$_g$ = +32,767)
-X$_g$ Axis
+X$_g$ Axis
Page Presentation Space
Graphics Presentation Space Window Specified in the Graphics Data Descriptor (GDD)
-Y$_g$ Axis
(x$_g$ =-32,768, y$_g$ =-32,768)
X$_{oa}$
Scale-to-fill mapping specified in the Map Graphics Object (MGO)
Y$_{oa}$
Graphics Object Area

Figure 42. Graphics Presentation Space Mapping: Scale to Fill

Note that the scale to fill mapping option is similar to scale to fit except that the Graphics presentation space window may be scaled asymmetrically to fill the object area completely. This means that the aspect ratio of the graphics picture may not be preserved.

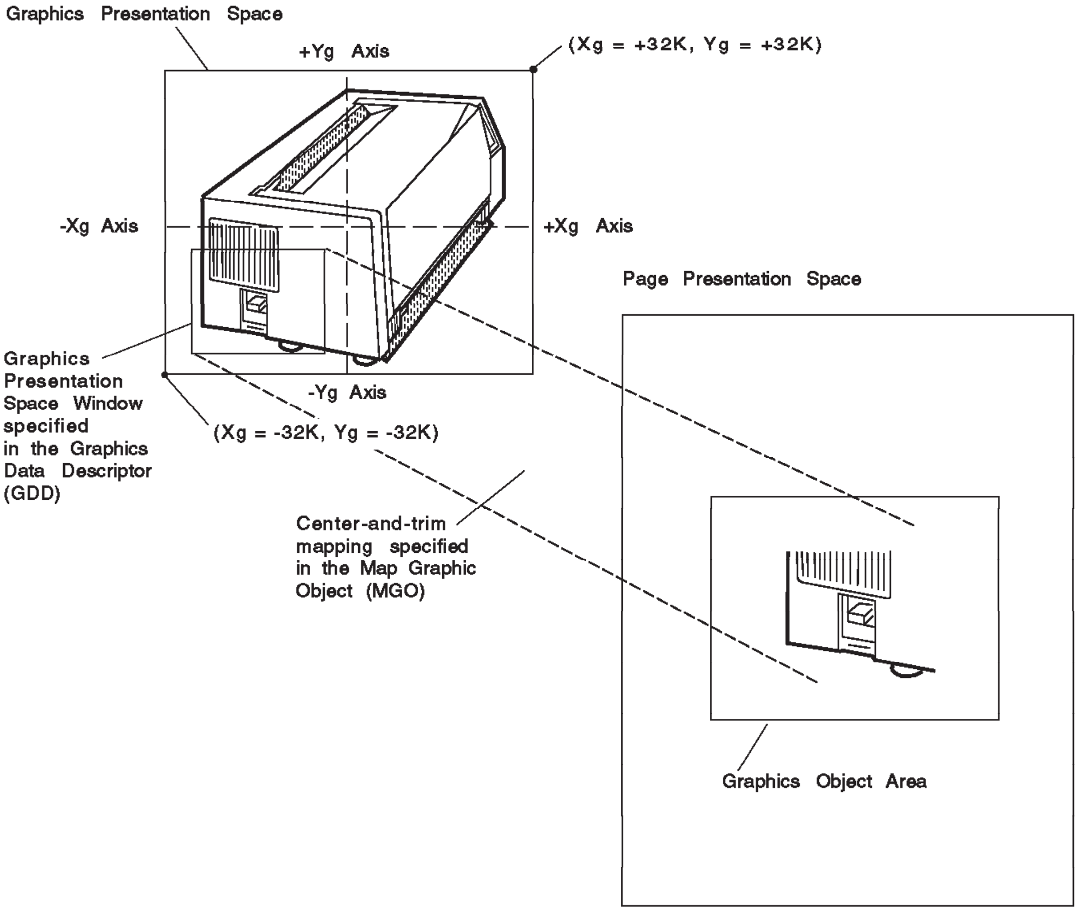

Graphics Presentation Space
+Yg Axis
(Xg = +32K, Yg = +32K)
-Xg Axis
+Xg Axis
Page Presentation Space
Graphics Presentation Space Window specified in the Graphics Data Descriptor (GDD)
-Yg Axis
(Xg = -32K, Yg = -32K)
Center-and-trim mapping specified in the Map Graphic Object (MGO)
Graphics Object Area

Figure 43. Graphics Presentation Space Mapping: Center and Trim

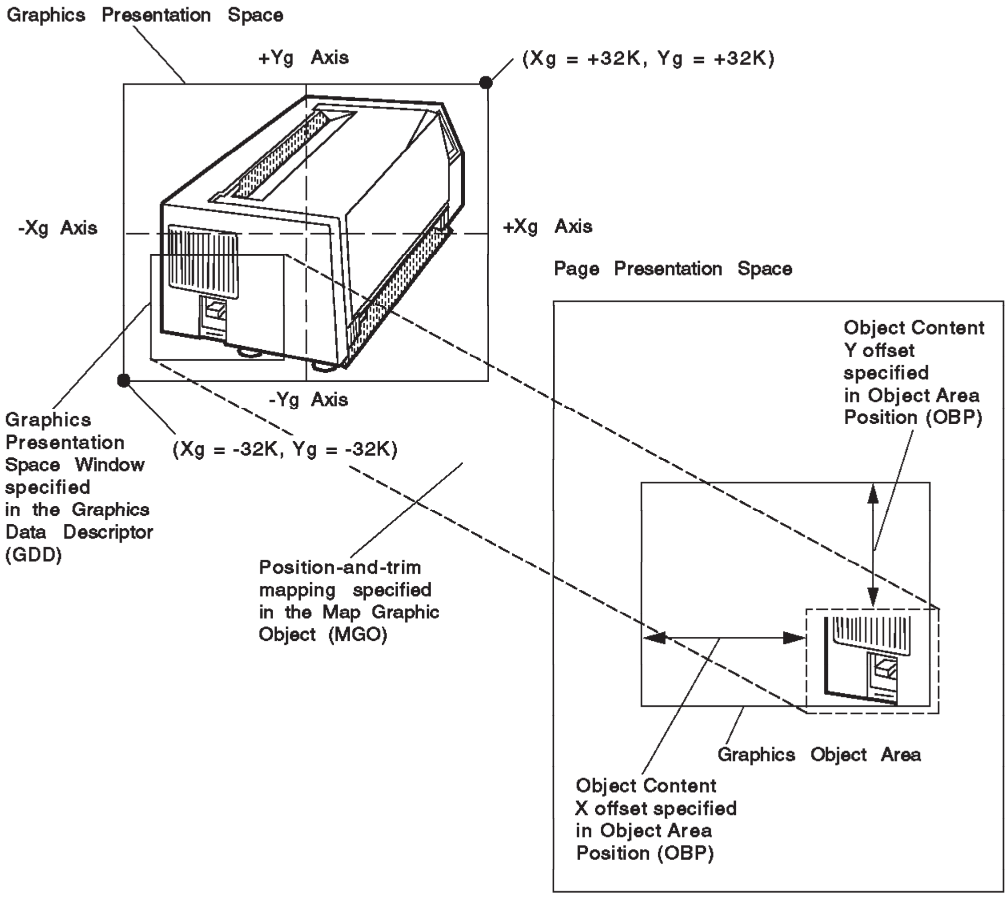

Graphics Presentation Space
+Yg Axis
(Xg = +32K, Yg = +32K)
-Xg Axis
+Xg Axis
Page Presentation Space
Object Content Y offset specified in Object Area Position (OBP)
-Yg Axis
Graphics Presentation Space Window specified in the Graphics Data Descriptor (GDD)
(Xg = -32K, Yg = -32K)
Position-and-trim mapping specified in the Map Graphic Object (MGO)
Graphics Object Area
Object Content X offset specified in Object Area Position (OBP)

Figure 44. Graphics Presentation Space Mapping: Position and Trim

Image Objects

Image data consists of an electronic representation of a picture in the form of an array of raster data, along with the controls to present this data. The image data object is defined by the Image Object Content Architecture and is sometimes referred to as an *IO image* object.

MO:DCA-P also supports the IM image object for migration purposes. For a definition of this object, see “IM Image Object” on page 520.

```
Begin Image Object  (BIM, D3A8FB)
         (     D3..C7)     Object Environment Group
      [  (IPD, D3EEFB)     Image Picture Data                     (S)  ]
End Image Object  (EIM, D3A9FB)

Object Environment Group (OEG) for IOCA FS10, FS11, FS40, or FS42 Image Object
Begin Object Environment Group  (BOG, D3A8C7)
         (OBD, D3A66B)     Object Area Descriptor
         (OBP, D3AC6B)     Object Area Position
      [  (MIO, D3ABFB)     Map Image Object                            ]
         (IDD, D3A6FB)     Image Data Descriptor
End Object Environment Group  (EOG, D3A9C7)

Object Environment Group (OEG) for IOCA FS45 Image Object
Begin Object Environment Group  (BOG, D3A8C7)
         (OBD, D3A66B)     Object Area Descriptor
         (OBP, D3AC6B)     Object Area Position
      [  (MIO, D3ABFB)     Map Image Object                            ]
      [  (MDR, D3ABC3)     Map Data Resource                      (S)  ]
         (IDD, D3A6FB)     Image Data Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 45. Image Object (IOCA FS10, FS11, FS40, FS42, or FS45 Level) Structure

Note: Refer to the *Image Object Content Architecture Reference* for a full description of the IOCA object content, syntax, and semantics for MO:DCA-P.

Mapping the Image Presentation Space

The mapping option is specified by the Mapping Option (X'04') triplet on the Map Image Object (MIO) structured field. The valid mapping options are:

- Scale to fit
- Scale to fill
- Center and trim
- Position and trim

These mapping options are shown in Figure 46 on page 96, Figure 47 on page 97, Figure 48 on page 98, and Figure 49 on page 99.

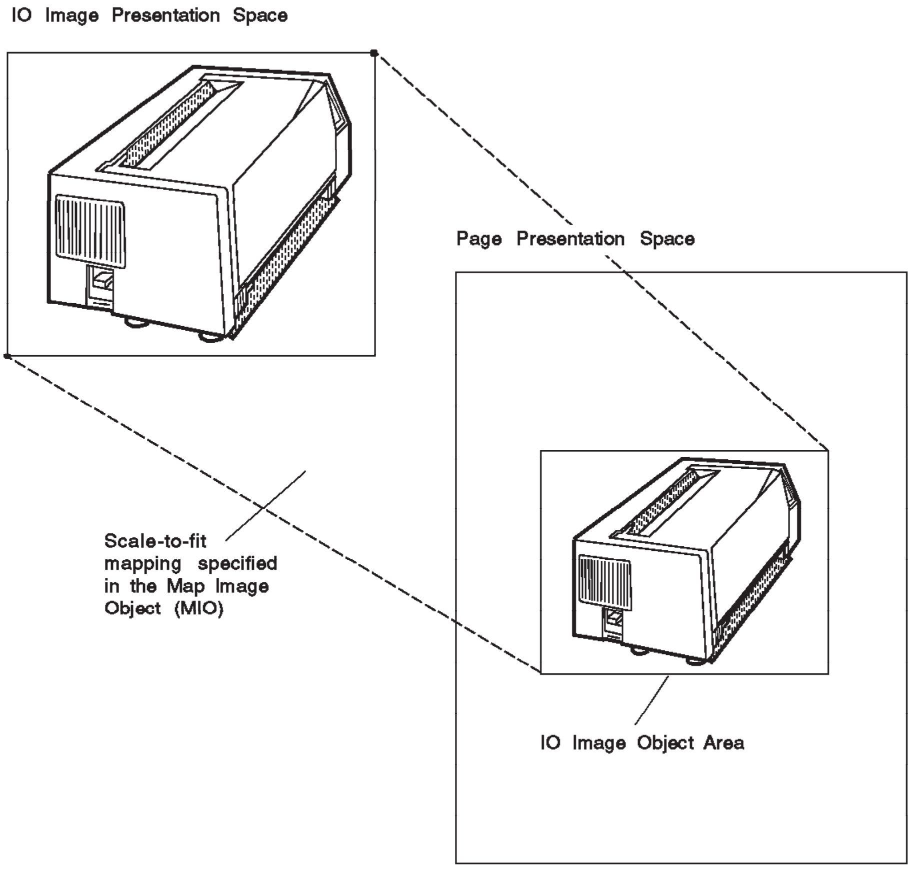

IO Image Presentation Space
Page Presentation Space
Scale-to-fit mapping specified in the Map Image Object (MIO)
IO Image Object Area

Figure 46. Image Presentation Space Mapping: Scale to Fit

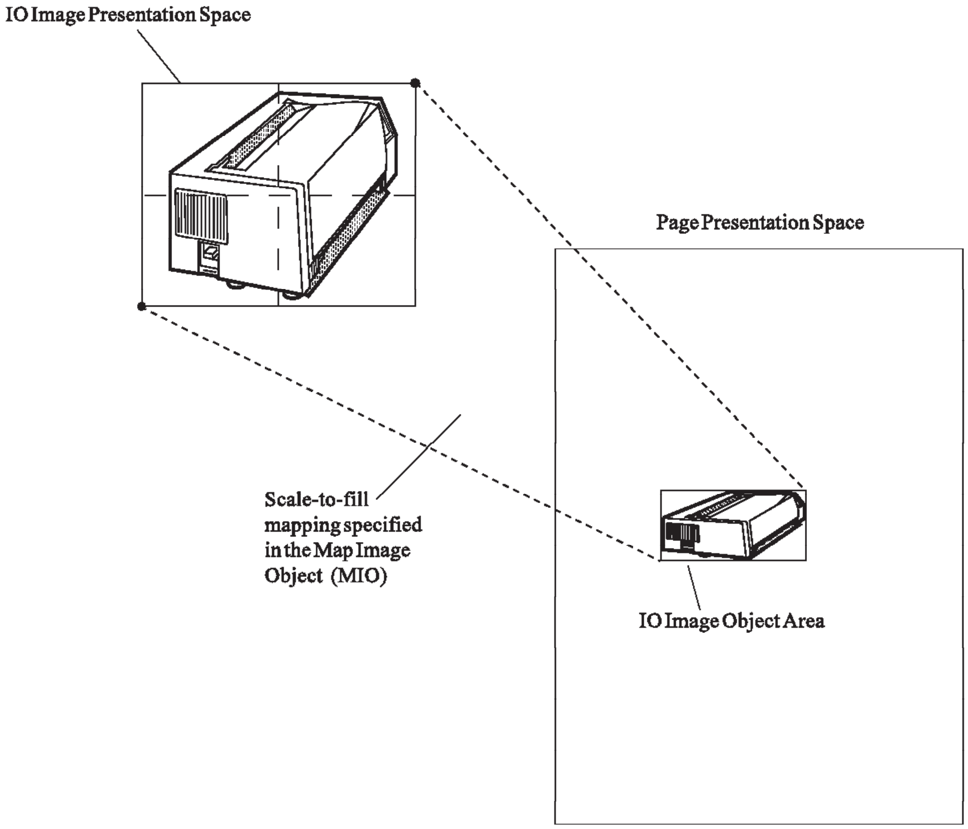

IO Image Presentation Space
Page Presentation Space
Scale-to-fill mapping specified in the Map Image Object (MIO)
IO Image Object Area

Figure 47. Image Presentation Space Mapping: Scale to Fill

Note that the scale to fill mapping option is similar to scale to fit except that the Image presentation space may be scaled asymmetrically to fill the object area completely. This means that the aspect ratio of the image may not be preserved.

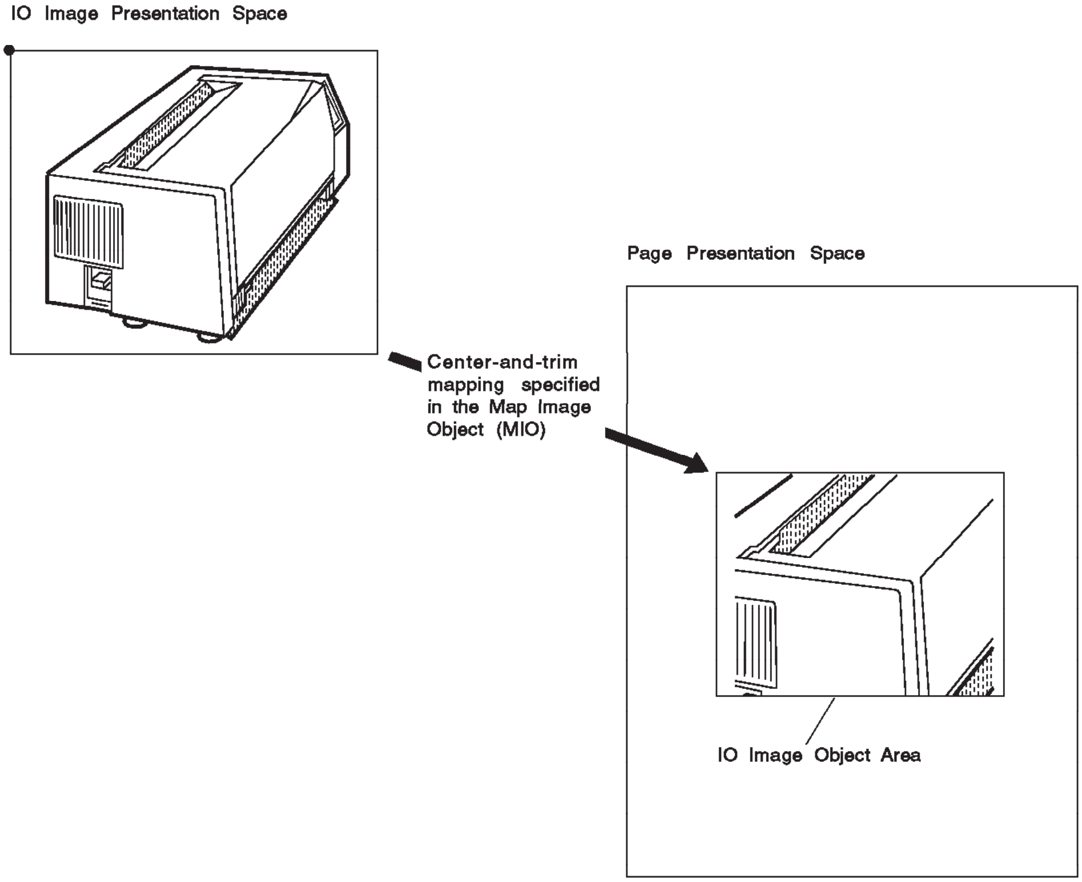

IO Image Presentation Space
Page Presentation Space
Center-and-trim mapping specified in the Map Image Object (MIO)
IO Image Object Area

Figure 48. Image Presentation Space Mapping: Center and Trim

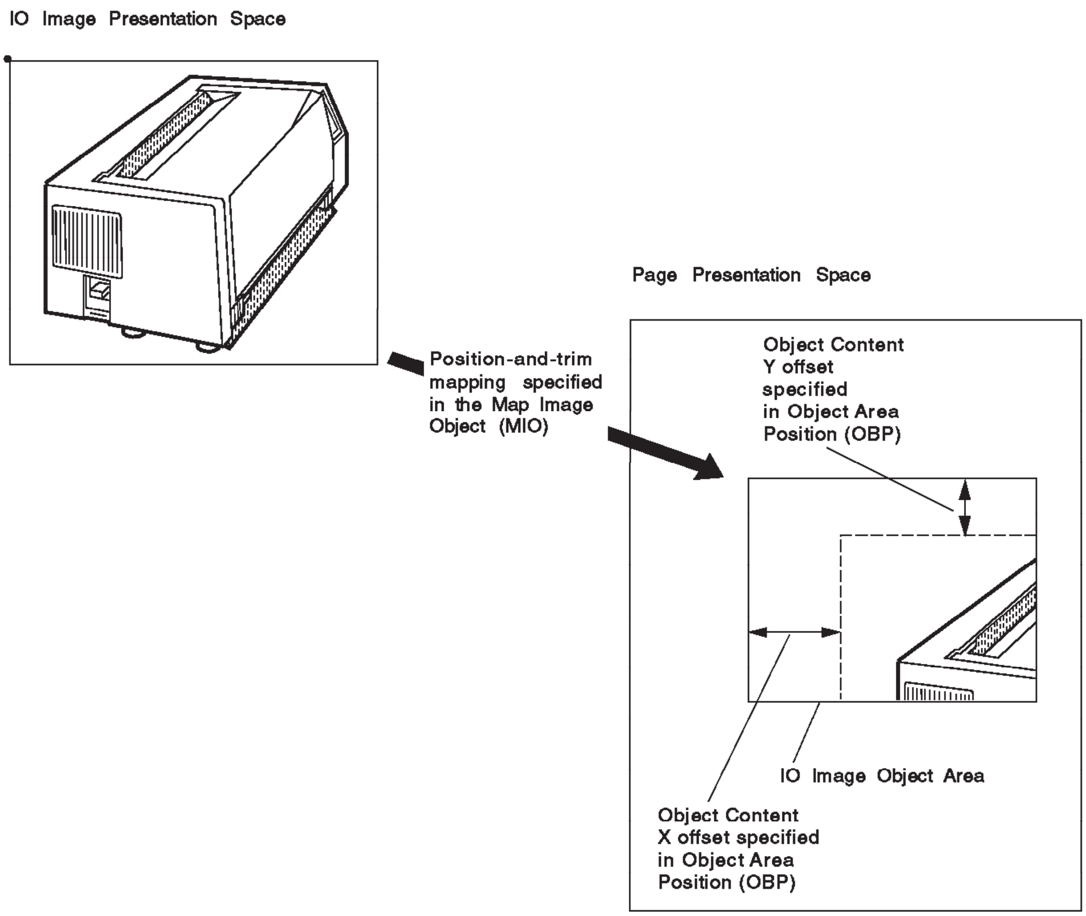

IO Image Presentation Space
Page Presentation Space
Position-and-trim mapping specified in the Map Image Object (MIO)
Object Content Y offset specified in Object Area Position (OBP)
IO Image Object Area
Object Content X offset specified in Object Area Position (OBP)

Figure 49. Image Presentation Space Mapping: Position and Trim

The MO:DCA-P architecture supports three additional mappings for the IOCA FS10 object for IM image migration purposes. For a definition of these mappings, see “Coexistence Parameters” on page 533.

Text Objects

Presentation text data consists of graphic character code points and the controls required to position and present the corresponding graphic characters. The presentation text data object is defined by the Presentation Text Object Content Architecture.

```
Begin Presentation Text Object  (BPT, D3A89B)
     [  (PTX, D3EE9B)     Presentation Text Data                  (S)  ]
End Presentation Text Object  (EPT, D3A99B)
```

Figure 50. Presentation Text Object (PTOCA PT1, PT2, or PT3 Level) Structure

Note: Refer to the *Presentation Text Object Content Architecture Reference* for a full description of the PTOCA object content, syntax, and semantics for MO:DCA-P.

When the BPT structured field is processed, all initial text conditions specified in the Presentation Text Descriptor (PTD) structured field are set prior to processing the text object.

Application Note: Whenever a BPT is encountered, AFP presentation servers set default page-level initial text conditions before the PTD initial conditions are set, see Table 15 on page 140.

Object Containers

Object containers are MO:DCA objects that envelop and carry object data. The object data may or may not be specified by an IBM presentation architecture. The object data is not constrained to be traditional text, image, or graphics. However if it is a presentation object, it must have a well-defined processing semantic resulting in a fixed, deterministic presentation when processed by a receiver capable of presenting the object. If the object is a traditional time-invariant presentation object, it must be paginated, that is its presentation space must be constrained to a single page. For presentation objects, the object data in the container is presented when the object container is included on a page or overlay using the Include Object (IOB) structured field. The object container may also be included directly on a page or overlay. Figure 51 shows how object container data is included on a page using the Include Object (IOB) structured field.

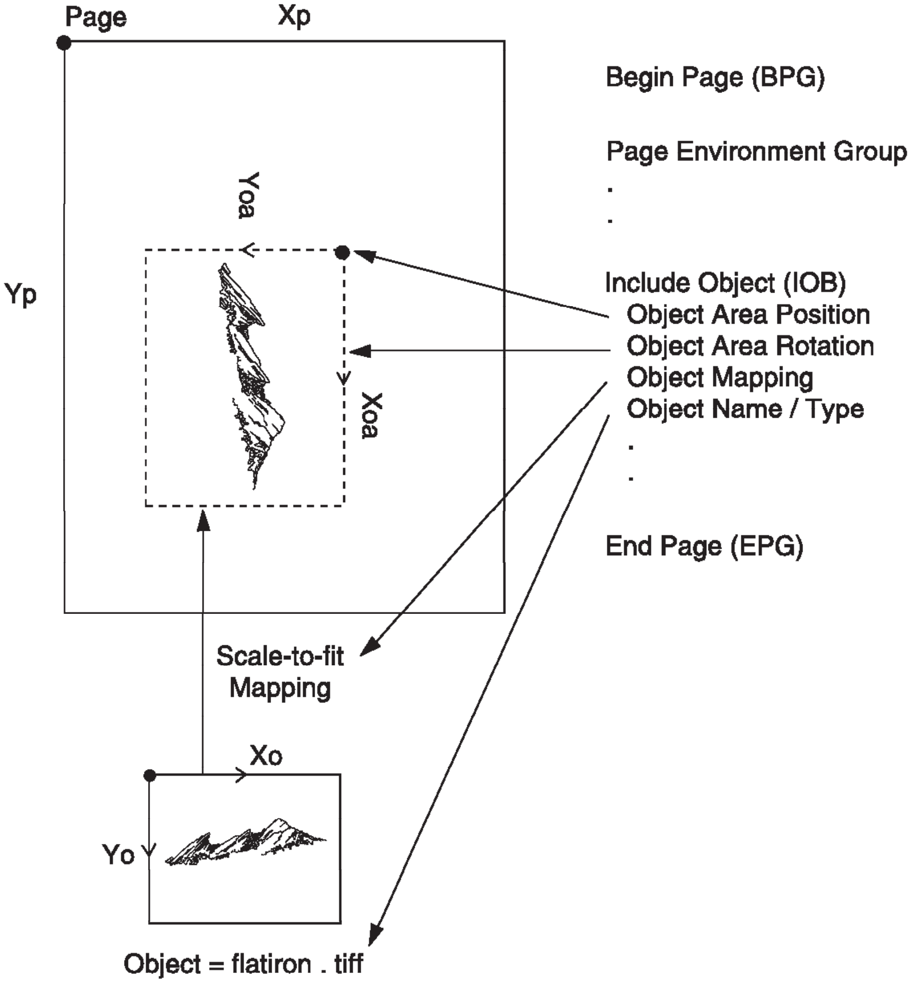


Figure 51. Use of the IOB to Include Object Container Data

The object container provides a range of functions that may be used to identify and structure the enveloped object data. At minimum, the container provides Begin and End structured fields, categorizes the object into a class, identifies the object type using a registered numeric identifier, and carries the object data in OCD structured fields. Above this minimum level of function, the object container may include additional optional functions such as an OEG to specify data object presentation space size, position, mapping and orientation.

For presentation objects, the required container structure depends on where the object is stored and how it is included in a page or overlay:

- If the object is included directly in a page or overlay, the container must, at a minimum, have the following structure:
  - BOC/EOC with the Object Classification (X'10') triplet on the BOC specifying the registered object-type identifier (object-type OID) for the object data format
  - OEG with OBD, OBP, and CDD
  - All object data partitioned into OCDs
- If the object is included using an Include Object (IOB) structured field and is carried in an external (printfile-level) resource group, the container must, at a minimum, have the following structure:
  - BOC/EOC with the Object Classification (X'10') triplet on the BOC specifying the registered object-type identifier (object-type OID) for the object data format
  - All object data partitioned into OCDs
- If the object is included using an Include Object (IOB) structured field and is stored in a resource library, there is no minimum container structure requirement, that is, the object may be stored and included in its unaltered, original form. However, if the included object is carried in a BOC/EOC container, the object data must be partitioned into OCDs.

```
Begin Object Container  (BOC, D3A892)
      [  (     D3..C7)     Object Environment Group                         ]
      [  (OCD, D3EE92)     Object Container Data                       (S)  ]
End Object Container  (EOC, D3A992)

Object Environment Group (OEG) for Object Container
Begin Object Environment Group  (BOG, D3A8C7)
      [  (OBD, D3A66B)     Object Area Descriptor                           ]
      [  (OBP, D3AC6B)     Object Area Position                             ]
      [  (MCD, D3AB92)     Map Container Data                               ]
      [  (MDR, D3ABC3)     Map Data Resource                          (S)   ]
      [  (CDD, D3A692)     Container Data Descriptor                        ]
End Object Environment Group  (EOG, D3A9C7)
```

Figure 52. Object Container Structure for Presentation Objects

Application Notes:

1. For purposes of Print Services Facility resource management, each MDR that is specified in the object container OEG must have a corresponding MDR mapping the same resource in the AEG for the page or overlay that includes the object container. Note that an FQN type X'BE' triplet, if specified on the MDR in the OEG, is not factored up to the AEG, unless the MDR maps a data-object font.
2. An MDR reference to a specific resource may only be specified once in the object container OEG.

For non-presentation objects, the required container structure depends on where the object is stored:

- If the object is carried in an external (printfile-level) resource group, the container must have the following structure:
  - BOC/EOC with the Object Classification (X'10') triplet on the BOC specifying the registered object-type identifier (object-type OID) for the object data format
  - All object data partitioned into OCDs
- If the object is stored in a resource library, there is no minimum container structure requirement, that is, the object may be stored in its unaltered, original form. However, if the object is stored in a BOC/EOC container, the object data must be partitioned into OCDs.

```
Begin Object Container  (BOC, D3A892)
     [  (OCD, D3EE92)    Object Container Data                     (S)  ]
End Object Container  (EOC, D3A992)
```

Figure 53. Object Container Structure for Non-Presentation Objects

Application Notes:

1. When a TrueType/OpenType font or a TrueType Collection is installed in an AFP resource library, it is not stored in a BOC/EOC container so that non-AFP applications that do not understand MO:DCA object containers are able to use the same font or collection.
2. When an object container is carried in an external (printfile-level) resource group in AFP environments, a BRS/ERS envelope is mandatory.

Mapping the Container Data Presentation Space

The mapping option is specified by the Mapping Option (X'04') triplet on the Map Container Data (MCD) structured field. The valid mapping options are:

- Scale to fit
- Scale to fill
- Center and trim
- Position and trim
- Position

For a description of the supported mapping options see “Mapping Option Triplet X'04'” on page 332. For the scale-to-fit and scale-to-fill mapping of presentation data in an object container, a data object presentation space size is required. See “Object Type Identifiers” on page 535 for information on how the presentation space size is specified by various data objects. If the presentation space size is not specified by the object, the achitected default is the presentation space size of the including page or overlay.

Chapter 5. MO:DCA Structured Fields

This chapter:
- Briefly describes the purpose of each MO:DCA structured field
- Provides the syntax and semantics for each MO:DCA structured field
- Identifies each structured field's parameter set
- Identifies exception conditions

General Information

Chapter 3, "MO:DCA Overview," on page 19 provides a general discussion of the syntax and semantics of MO:DCA structured fields. Detailed formats, syntaxes and semantics are provided here to enable product developers to design and produce applications that can use MO:DCA data streams.

The syntax tables in this chapter describe the less restrictive requirements of the overall architecture. Thus, these syntax tables may not agree exactly with a specific interchange set with regard to:
- Whether a data element is mandatory or optional
- The number of times a particular data element may validly occur
- The order in which the data elements must occur

In those cases where there is disagreement with an interchange set, the interchange set requirement governs.

The exception condition column of the syntax tables for these structured fields identifies only those exception conditions that could occur for the individual parameters.

Structured fields that have triplets reflect an exception condition code of either X'10' or X'14' in this column for the triplet entry. This reflects only the possibility that the structured field could include an invalid triplet, or that a required triplet could be missing. Any exception conditions relating to a triplet's data elements are addressed in Chapter 6, "MO:DCA Triplets," on page 313.

Those exception conditions that may occur because of special conditions such as a mismatch between the individual parameters of one or more structured fields are listed under the *Semantics* headings when only one such exception condition is identified. When multiple exception conditions are identified, all are listed under the "Exception Condition Summary" heading. A more detailed explanation may be provided under the "Semantics" heading.

Architected defaults are identified in the semantic description of the individual parameters. When an architected default exists for an entire structured field, the default is documented at the end of the semantic description for that structured field.

The following structured field definitions are sorted in alphabetical order based on structured field acronym.

Begin Active Environment Group (BAG)

The Begin Active Environment Group structured field begins an Active Environment Group, which establishes the environment parameters for the page or overlay. The scope of the active environment group is the containing page or overlay.

BAG (X'D3A8C9') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8C9'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	AEGName		Name of the active environment group	O	X'02'
8–*n*		Triplets		See “BAG Semantics” for triplet applicability.	O	X'10'

BAG Semantics

AEGName Is the name of the active environment group.

The page or overlay containing the Begin Active Environment Group structured field must also contain a subsequent matching End Active Environment Group structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BAG Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Active Environment Group structured field is not present in the page or overlay.

Begin Bar Code Object (BBC)

The Begin Bar Code Object structured field begins a bar code data object, which becomes the current data object.

BBC (X'D3A8EB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8EB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	BCdoName		Name of the bar code data object	O	X'02'
8–*n*		Triplets		See “BBC Semantics” for triplet applicability.	O	X'10'

BBC Semantics

BCdoName Is the name of the bar code data object.

The page, overlay, or resource group containing the Begin Bar Code Object structured field must also contain a subsequent matching End Bar Code Object structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Bar Code Object structured field name and is used as the name of the bar code data object.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X’62’) triplet may occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BBC Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Bar Code Object structured field is not present in the page, overlay, or resource group.

Begin Color Attribute Table (BCA)

The Begin Color Attribute Table structured field begins a Color Attribute Table resource object, which becomes the current resource object. A color attribute table contains color attribute data.

Note: The BCA structured field is used only in MO:DCA-L data streams.

BCA (X'D3A877') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A877'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	CATName		Name of the color attribute table	M	X'06'
8–*n*		Triplets		See “BCA Semantics” for triplet applicability.	O	X'10'

BCA Semantics

CATName Is the name of the color attribute table. This name may not appear on more than one Begin Color Attribute Table in the same resource group or a X’01’ exception condition exists.

The resource group containing the Begin Color Attribute Table structured field must also contain a subsequent matching End Color Attribute Table structured field, or a X’08’ exception condition exists.

Color attribute tables may reside in external libraries, in one or more resource groups within a MO:DCA document, or in a combination of the two. See “Resource Groups” on page 17 for details on locating resource objects within libraries and resource groups.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Color Attribute Table structured field name and is used as the name of the color attribute table.

Triplet	Type	Usage
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See "Comment Triplet X'65'" on page 385.

BCA Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Color Attribute Table structured field is not present in the same resource group.
- A X'01' exception condition exists when multiple Begin Color Attribute Table structured fields with the same name exist within the same resource group.

Bar Code Data (BDA)

The Bar Code Data structured field contains the data for a bar code object.

BDA (X'D3EEEB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3EEEB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	BCOCAdat		Up to 32759 bytes of BCOCA-defined data	O	X'00'

BDA Semantics

BCOCAdat Contains the BCOCA-defined data. See the MO:DCA environment appendix in the *Bar Code Object Content Architecture Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Bar Code Data Descriptor (BDD)

The Bar Code Data Descriptor structured field contains the descriptor data for a bar code data object.

BDD (X'D3A6EB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A6EB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–*n*	UNDF	BCOCAdes		Up to 32759 bytes of BCOCA-defined descriptor data	O	X'00'

BDD Semantics

BCOCAdes Contains the BCOCA-defined descriptor data. See the MO:DCA environment appendix in the *Bar Code Object Content Architecture Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Begin Document Environment Group (BDG)

The Begin Document Environment Group structured field begins a document environment group, which establishes the environment parameters for the form map object. The scope of the document environment group is the containing form map.

BDG (X'D3A8C4') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8C4'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	DEGName		Name of the document environment group	O	X'02'
8–*n*		Triplets		See “BDG Semantics” for triplet applicability.	O	X'10'

BDG Semantics

DEGName Is the name of the document environment group.

The form map containing the Begin Document Environment Group structured field must also contain a subsequent matching End Document Environment Group structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	**Type**	**Usage**
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BDG Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Document Environment Group structured field is not present in the form map.

Begin Document Index (BDI)

The Begin Document Index structured field begins the document index.

BDI (X'D3A8A7') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8A7'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	IndxName		Name of the document index	O	X'02'
8–*n*		Triplets		See “BDI Semantics” for triplet applicability.	O	X'10'

BDI Semantics

IndxName Is the name of the document index.

If specified within a document, this structured field must be the first structured field after the Begin Document structured field, or a X’20’ exception condition exists.

The print file or document containing the Begin Document Index structured field must also contain a subsequent matching End Document Index structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Document Index structured field name and is used as the name of the document index.
X'02'	Fully Qualified Name	Optional. May occur once. The Fully Qualified Name type that may appear is **X'83'**—*Begin Document Name*. Specifies the name of the document that is indexed by this document index. See “Fully Qualified Name Triplet X'02'” on page 320.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383

Triplet	Type	Usage
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See "Comment Triplet X'65'" on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X'62') triplet may occur once. Assigns a universal date and time stamp to the object. See "Universal Date and Time Stamp Triplet X'72'" on page 394.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BDI Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Document Index structured field is not present in the print file or document.
- A X'20' exception condition exists when this structured field is specified in a document but does not follow the BDT structured field.

Begin Document (BDT)

The Begin Document structured field names and begins the document.

BDT (X'D3A8A8') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8A8'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	DocName		Name of the document	M	X'06'
8–9				Reserved; must be binary zero	M	X'06'
10–*n*		Triplets		See “BDT Semantics” for triplet applicability.	M	X'14'

BDT Semantics

DocName Is the name of the document described by the data stream. If a Fully Qualified Name type X’01’ (Replace First GID) appears in this structured field, the name specified in this parameter is ignored and the GID provided by the triplet is used instead. If the value of the first two bytes of DocName are X'FFFF', the processing system provides the document name.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Mandatory. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'18'	MO:DCA Interchange Set	For *interchange* data streams, this triplet is mandatory and must occur once. For *private* or *exchange* data streams, this triplet is not permitted. See “MO:DCA Interchange Set Triplet X'18'” on page 339.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID name*. This GID overrides the Begin Document structured field name and is used as the name of the document.
X'02'	Fully Qualified Name	Optional. May occur once. The Fully Qualified Name type that may appear is **X'0A'**—*Begin Resource Group Name*. Specifies the name of a resource group that contains resources referenced in this document. See “Fully Qualified Name Triplet X'02'” on page 320.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. The Fully Qualified Name type that may appear is **X'98'**—*Begin Document Index Name*. Specifies the name of a document index resource object that provides index information for this document. See “Fully Qualified Name Triplet X'02'” on page 320.
X'21'	Object Function Set Specification	Optional. May occur once for each object type that exists in the data stream. See “Object Function Set Specification Triplet X'21'” on page 346.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. May occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

The data stream containing the Begin Document structured field must also contain a subsequent matching End Document structured field, or a X’08’ exception condition exists.

BDT Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Document structured field is not present in the data stream.
- A X’01’ exception condition exists when:
  - Multiple type X’01’ (Replace First GID) Fully Qualified Name triplets appear.
  - Multiple MO:DCA Interchange Set (X’18’) triplets appear.

Begin Form Map (BFM)

The Begin Form Map structured field begins a form map object, also called a *form definition* or *formdef*. A form map is a print control resource object that contains one or more medium map resource objects that are invokable on document and page boundaries and that specify a complete set of presentation controls. It also contains an optional document environment group (DEG) that defines the presentation environment for the form map.

BFM (X'D3A8CD') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8CD'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	FMName		Name of the form map	O	X'02'
8–*n*		Triplets		See “BFM Semantics” for triplet applicability.	O	X'10'

BFM Semantics

FMName Is the name of the form map.

A form map resource object must be terminated with a subsequent matching End Form Map structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X’62’) triplet may occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BFM Exception Condition Summary

- A X′08′ exception condition exists when the form map is not terminated with a subsequent matching End Form Map structured field.

Begin Graphics Object (BGR)

The Begin Graphics Object structured field begins a graphics data object which becomes the current data object.

BGR (X'D3A8BB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8BB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	GdoName		Name of the graphics data object	O	X'02'
8–*n*		Triplets		See “BGR Semantics” for triplet applicability.	O	X'10'

BGR Semantics

GdoName Is the name of the graphics data object.

The page, overlay, or resource group containing the Begin Graphics Object structured field must also contain a subsequent matching End Graphics Object structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Graphics Object structured field name and is used as the name of the graphics data object.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X’62’) triplet may occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BGR Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Graphics Object structured field is not present in the page, overlay, or resource group.

Begin Image Object (BIM)

The Begin Image Object structured field begins an IOCA image data object, which becomes the current data object.

Architecture Note: A migration form of the image object is supported in AFP environments and is defined as the *IM Image Object* in “IM Image Object” on page 520.

BIM (X'D3A8FB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8FB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	IdoName		Name of the image data object	O	X'02'
8–*n*		Triplets		See “BIM Semantics” for triplet applicability.	O	X'10'

BIM Semantics

IdoName Is the name of the IOCA image data object.

The page, overlay, or resource group containing the Begin Image Object structured field must also contain a subsequent matching End Image Object structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Image Object structured field name and is used as the identifier of the image data object. The identifier may be specified in one—and only one—of the following formats: • If FQNFmt = X'00', the identifier is a character-encoded name. See “External Resource Naming Conventions” on page 80 for a description of the naming conventions used in AFP environments.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.

Triplet	Type	Usage
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See "Comment Triplet X'65'" on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X'62') triplet may occur once. Assigns a universal date and time stamp to the object. See "Universal Date and Time Stamp Triplet X'72'" on page 394.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BIM Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Image Object structured field is not present in the page, overlay, or resource group.

Begin Medium Map (BMM)

The Begin Medium Map structured field begins a medium map resource object. A medium map is a print control resource object that contains a complete set of controls for presenting pages on physical media such as sheets and for generating multiple copies of sheets with selectable modifications. These controls may be grouped into two categories:

- Medium-level controls
- Page-level controls

Medium-level controls are controls that affect the medium, such as the specification of medium overlays, medium size, medium orientation, medium copies, simplex or duplex, medium finishing, media type, and media source and destination selection. These controls are defined by the Map Medium Overlay (MMO), Medium Descriptor (MDD), Medium Copy Count (MCC), Medium Finishing Control (MFC), Map Media Type (MMT), and Medium Modification Control (MMC) structured fields. Page-level controls are controls that affect the pages that are placed on the medium, such as the specification of page modifications, page position, and page orientation. These controls are defined by the Map Page Overlay (MPO), Page Position (PGP), and Page Modification Control (PMC) structured fields.

BMM (X'D3A8CC') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8CC'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	MMName		Name of the medium map	M	X'06'
8–*n*		Triplets		See “BMM Semantics” for triplet applicability.	O	X'10'

BMM Semantics

MMName Is the name of the medium map.

A medium map resource object must be terminated with a subsequent matching End Medium Map structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.

Triplet	Type	Usage
X'45'	Media Eject Control	Optional. May occur once. See “Media Eject Control Triplet X'45'” on page 356. Specifies the type of media eject that should be performed when this medium map is invoked and N-up partitioning is specified. This triplet is ignored when it occurs on the medium map that is activated at the beginning of a document regardless of whether this medium map is explicitly invoked or implicitly invoked as the default. **Note:** If this triplet is not present, the architected default for the EjCtrl parameter in the triplet is X'01', that is perform a sheet eject and activate all controls specified by the medium map.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.

BMM Exception Condition Summary

- A X’08’ exception condition exists when the medium map is not terminated with a subsequent matching End Medium Map structured field.
- A X’01’ exception condition exists when:
  - The Begin Medium Map structured field specifies a conditional eject to a front-side partition and the PGP in the medium map does not specify a front-side partition
  - The Begin Medium Map structured field specifies a conditional eject to a back-side partition and the PGP in the medium map does not specify a back-side partition.

Begin Overlay (BMO)

The Begin Overlay structured field begins an overlay. An overlay contains an active environment group to establish parameters such as the size of the overlay’s presentation space and the fonts to be used by the data objects. It may also contain any mixture of:

- Bar code objects
- Graphics objects
- Image objects
- Object containers
- Presentation text objects

BMO (X'D3A8DF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8DF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	OvlyName		Name of the overlay	M	X'06'
8–*n*		Triplets		See “BMO Semantics” for triplet applicability.	O	X'10'

BMO Semantics

OvlyName Is the name of the overlay. This name may not appear on more than one Begin Overlay within the same resource group or a X’01’ exception condition exists.

The resource group containing the Begin Overlay structured field must also contain a subsequent matching End Overlay structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is: **X'01'**—*Replace First GID Name*. This GID overrides the Begin Overlay structured field name and is used as the name of the overlay.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.

Triplet	Type	Usage
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X’62’) triplet may occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

Overlays reside in external resource libraries or in resource groups. See “Resource Groups” on page 17 for details on locating resource objects within libraries and resource groups.

Application Note: In AFP environments, the following retired triplets are used on this structured field:

- Object Checksum (X’63’) triplet; see “Object Checksum Triplet X'63'” on page 511
- Object Origin Identifier (X’64’) triplet; see “Object Origin Identifier Triplet X'64'” on page 513

BMO Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Overlay structured field is not present in the same resource group.
- A X’01’ exception condition exists when multiple Begin Overlay structured fields with the same name exist within the same resource group.

Begin Named Page Group (BNG)

The Begin Named Page Group structured field begins a page group, which is a named, logical grouping of sequential pages. A page group may contain other nested page groups. All pages in the page group and all other page groups that are nested in the page group inherit the attributes that are assigned to the page group using TLE structured fields.

BNG (X'D3A8AD') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8AD'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	PGrpName		Name of the page group	M	X'06'
8–*n*		Triplets		See "BNG Semantics" for triplet applicability.	O	X'10'

BNG Semantics

PGrpName
Is the name of the page group.

The document containing the Begin Named Page Group structured field must also contain a subsequent matching End Named Page Group structured field, or a X'08' exception condition exists.

Triplets
Appear in the Begin Named Page Group structured field as follows:

Triplet	**Type**	**Usage**
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID name*. This GID overrides the Begin Named Page Group structured field name and is used as the name of the page group.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'8D'**—*Begin Medium Map Reference*. Specifies the name of the medium map that is active at the beginning of the page group. **Application Note:** This triplet is typically specified on the BNG structured fields when the page group is to be archived with a specific form map. It allows the page group to be retrieved and viewed at a later time without "playing back" the whole document. This triplet is ignored by print servers.
X'56'	Medium Map Page Number	Optional. May occur once. Specifies the sequence number of the first page-group page in the set of sequential pages controlled by the medium map that is active at the beginning of the page group. The first page in the set has sequence number 1. See "Medium Map Page Number Triplet X'56'" on page 374. **Application Note:** This triplet is typically specified on the BNG structured fields when the page group is to be archived with a specific form map. It allows the page group to be retrieved and viewed at a later time without "playing back" the whole document. This triplet is ignored by print servers. Note that similar functionality can be achieved by specifying the Page Position Information (X'81') triplet on the BPG for the pages in the page group.
X'5E'	Object Count	Optional. May occur once for each subordinate object type counted. Specifies how many subordinate objects of a particular type, such as a page, are contained within the page group. See "Object Count Triplet X'5E'" on page 381.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See "Comment Triplet X'65'" on page 385.
X'83'	Presentation Control	Optional. May occur once. Specifies whether the page group is intended to be indexed. If this triplet is not specified, the architected default is that the page group is intended to be indexed. This triplet is ignored for printing. See "Presentation Control Triplet X'83'" on page 405.

Architecture Note: If page-group-level indexing is used for a document that contains page groups, it is recommended that the page group name, whether it is specified by an 8-byte token name or by a fully qualified name, be unique with respect to other page group names within the document.

Application Notes:

1. The FQN Begin Medium Map Reference (type X'8D') triplet and the Medium Map Page Number (X'56') triplet may be used by viewing applications to present the page group in stand-alone fashion as it would be presented within the context of the complete document. These triplets are ignored by print servers.
2. Page groups are often processed in stand-alone fashion, that is, they are indexed, retrieved, and presented outside the context of the containing document. While the pages in the group are independent of each other and of any other pages in the document, their formatting on media depends on when the last medium map was invoked and on how many pages precede the BNG since this invocation. To make the media formatting of page groups self-contained, a medium map should be invoked at the beginning of the page group between the Begin Named Group (BNG) structured field and the first Begin Page (BPG) structured field. If this is not done, the presentation system may need to "play back" all pages between the invocation of the active medium map and the BNG to determine media formatting such as sheet-side and partition number for the first page in the group.

   It is therefore *strongly* recommended that in environments where stand-alone page group processing is required or anticipated, page groups are built with an Invoke Medium Map (IMM) structured field specified after the BNG and before the first BPG. IBM AFP applications that generate page groups will support a user option that ensures that an IMM is specified after BNG and before the first BPG, and IBM AFP archive servers will expect an IMM there and may not present the page group correctly if none is found. However, note that this may cause the complete document to print differently.

   A newer method to specify how a page or page group should be formatted involves use of the Page Position Information (X'81') triplet. This triplet may be specified on a BPG and indicates the repeating group in the PGP structured field in the active medium map that should be used to format the page.

BNG Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Named Page Group structured field is not present in the document.
- A X'01' exception condition exists when the same subordinate object type, such as a page, is counted in more than one X'5E' triplet.

Begin Object Container (BOC)

The Begin Object Container structured field begins an object container, which may be used to envelop and carry object data. The object data may or may not be defined by an IBM presentation architecture.

| **Application Note:** To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same. For TrueType/OpenType fonts, optimal performance can be achieved by using UTF-16BE as the encoding scheme.

BOC (X'D3A892') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A892'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	ObjCName		Name of the object container	M	X'06'
8–*n*		Triplets		See “BOC Semantics” for triplet applicability.	M	X'14'

BOC Semantics

ObjCName Is the name of the object container.

The page, overlay, or resource group containing the Begin Object Container structured field must also contain a subsequent matching End Object Container structured field, or a X’08’ exception condition exists.

Triplets Appear in the Begin Object Container structured field as follows:

Triplet	Type	Usage
X'10'	Object Classification	Mandatory. Must occur once. Specifies information used to classify and identify the enveloped object data. See “Object Classification Triplet X'10'” on page 335.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID name*. This GID overrides the Begin Object Container structured field name and is used as the identifier of the object container. The identifier may be specified in one—and only one—of the following formats: • If FQNFmt = X'00', the identifier is a character-encoded name. See “External Resource Naming Conventions” on page 80 for a description of the naming conventions used in AFP environments. The character-encoded name on the BOC is optional if the container is in a print-file-level resource group and the name is already specified on the BRS that immediately precedes the BOC. If the object in the container is a TrueType/OpenType font (TTF), this version of the triplet may occur more than once, and each instance of the triplet is used to specify the full font name in a language used in the font naming table. The character encoding is UTF-16BE. • If FQNFmt = X'10', the identifier is a ASN.1 OID encoded using the definite short form. This format provides a unique and system-independent method to identify a resource. It may be used to identify resources that are resident in, or have been captured by, the presentation device. Such an identifier is referred to as an *object OID*. Note that the object OID is associated with the resource content; it does not reflect the MO:DCA wrappers used to carry the content. If the BOC specifies an object OID and envelopes either a TrueType/OpenType font file or a TrueType collection file, the OID may be used to locate a printer-resident version of the object. It also makes the object a candidate for capture by the printer. In this case this version of the triplet may only occur once. **Architecture Note:** If the BOC is used to carry a TrueType/OpenType font in an AFP resource group, the FQN type X'01' triplet on the mandatory BRS must specify the full font name using FQNFmt = X'00'. The FQN type X'01' triplet on the BOC may then be used to specify the object OID for the font using FQNFmt = X'10'; this enables the server to use a printer-resident version of the font.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur more than once. See “Fully Qualified Name Triplet X'02'” on page 320. This triplet is optional on the BOC if the container is in a print-file-level resource group and the same triplet is already specified on the BRS that immediately precedes the BOC. The Fully Qualified Name type that may appear is **X'6E'**—*Data-object Font Base Font Identifier*. This triplet may be specified on a BOC to indicate the following: • If the BOC envelopes a TrueType Collection (TTC) file, the FQN type X'6E' triplet specifies a base TrueType/OpenType font that is contained in the collection. The identifier may be specified in the following format. • If FQNFmt = X'00', the identifier is a character-encoded name. The character string that identifies the font must be the *full font name* specified in a name record in the mandatory Naming Table of the font file. This parameter is specified in a name record with Name ID 4. An example of a full font name is *Times New Roman Bold*. Each instance of the FQN X'6E' triplet with FQNFmt = X'00' is used to specify the full font name of the base font in a language used in the font’s Naming Table. The character encoding is UTF-16, which matches the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001') in the Naming Table. The byte order is big endian. For example, if the font Naming Table contains two name records for the full font name (Name ID 4) - one in English - United States (LCID = X'0409') and one in German - Standard (LCID = X'0407'), both in the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001'), each of these names, encoded in UTF-16BE, is carried in a FQN X'6E' triplet on the BOC.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur more than once. See “Fully Qualified Name Triplet X'02'” on page 320. This triplet is optional on the BOC if the container is in a print-file-level resource group and the same triplet is already specified on the BRS that immediately precedes the BOC. The Fully Qualified Name type that may appear is **X'7E'**—*Data-object Font Linked Font Identifier*. This triplet may be specified on a BOC to indicate the following: • If the BOC envelopes a TrueType/OpenType font (TTF/OTF) file, the FQN type X'7E' triplet specifies a linked font for the base font. The order in which the FQN type X'7E' triplets are specified determines the order in which the linked fonts are processed. • If the BOC envelopes a TrueType Collection (TTC) file, the FQN type X'7E' triplet specifies a linked font for the base font that is identified with the immediately preceding FQN type X'6E' triplet. Note that if the base font is specified in multiple languages using multiple FQN type X'6E' triplets, each instance of the FQN type X'6E' triplet must be followed by the sequence of FQN type X'7E' triplets that identify the linked fonts for the base font. The identifier may be specified in the following format. • If FQNFmt = X'00', the identifier is a character-encoded name. The character string that identifies the font must be the *full font name* specified in a name record in the mandatory Naming Table of the font file. This parameter is specified in a name record with Name ID 4. An example of a full font name is *Times New Roman Bold*. The character encoding is UTF-16, which matches the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001') in the Naming Table. The byte order is big endian.
X'57'	Object Byte Extent	Optional. May occur once. Specifies the number of bytes contained in the object container. The byte extent is measured starting with the first byte of the Begin Object Container (BOC) structured field up to and including the last byte of the End Object Container (EOC) structured field. See “Object Byte Extent Triplet X'57'” on page 375.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X’62’) triplet may occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

BOC Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Object Container structured field is not present in the page, overlay, or resource group.

Begin Object Environment Group (BOG)

The Begin Object Environment Group structured field begins an Object Environment Group, which establishes the environment parameters for the object. The scope of an object environment group is its containing object.

BOG (X'D3A8C7') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8C7'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	OEGName		Name of the object environment group	O	X'02'
8–*n*		Triplets		See "BOC Semantics" on page 129 for triplet applicability.	O	X'10'

BOG Semantics

OEGName Is the name of the object environment group.

The object containing the Begin Object Environment Group structured field must also contain a subsequent matching End Object Environment Group structured field, or a X'08' exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See "Comment Triplet X'65'" on page 385.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BOG Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Object Environment Group structured field is not present in the object.

Begin Page (BPG)

The Begin Page structured field begins a presentation page. A presentation page contains an active environment group to establish parameters such as the size of the page's presentation space and the fonts to be used by the data objects. It may also contain any mixture of:

- Bar code objects
- Graphics objects
- Image objects
- Object containers
- Presentation text objects

BPG (X'D3A8AF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8AF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	PageName		Name of the page	O	X'02'
8–*n*		Triplets		See "BPG Semantics" for triplet applicability.	O	X'10'

BPG Semantics

PageName Is the name of the page.

The document containing the Begin Page structured field must also contain a subsequent matching End Page structured field, or a X'08' exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Page structured field name and is used as the name of the page.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'8D'**—*Begin Medium Map Reference*. Specifies the name of the medium map object that is active for presenting the page on a physical medium. **Application Note:** This triplet is typically specified on the BPG structured fields when the page or page group is to be archived with a specific form map. It allows the page or page group to be retrieved and viewed at a later time without “playing back” the whole document. This triplet is ignored by print servers.
X'56'	Medium Map Page Number	Optional. May occur once. Specifies the sequence number of the page in the set of sequential pages controlled by the active medium map. The first page in the set has sequence number 1. See “Medium Map Page Number Triplet X'56'” on page 374. **Application Note:** This triplet is typically specified on the BPG structured fields when the page is to be archived with a specific form map. It allows the page to be retrieved and viewed at a later time without “playing back” the whole document. This triplet is ignored by print servers. Note that the Medium Map Page Number (X'56') triplet is not needed if a Page Position Information (X'81') triplet is specified, and is overridden by the latter.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'81'	Page Position Information	Optional. May occur once. Specifies the PGP repeating group that is used to view the page and its PMC overlay data. The PGP is specified in the medium map referenced by a FQN type X'8D'—Begin Medium Map Reference triplet. If the X'81' triplet is specified, it overrides a Medium Map Page Number (X'56') triplet. This triplet is not used for printing and is ignored by print servers. See “Page Position Information Triplet X'81'” on page 403.
X'83'	Presentation Control	Optional. May occur once. Specified on a BPG to indicate whether the page is intended to be viewed. If this triplet is not specified, the architected default is that the page is intended to be viewed. If this triplet is also specified on an Index Element (IEL) that indexes the page, the IEL triplet overrides if there is a conflict. This triplet is ignored for printing. See “Presentation Control Triplet X'83'” on page 405.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

Architecture Notes:

1. If a page is to be indexed or if it is to be included in a resource document, a page name is required so that the page can be identified and referenced. It is therefore highly recommended that the BPG structured field always specify a page name.
2. If page-level indexing is used for the document that contains this page, or if this page is part of a resource document, it is recommended that the page name, whether it is specified by an 8-byte token name or by a fully qualified name, be unique with respect to other page names within the document.

Application Note: The FQN Begin Medium Map Reference (type X'8D') triplet, the Medium Map Page Number (X'56') triplet, the Page Position Information (X'81') triplet, and the Presentation Control (X'83') triplet may be used by viewing applications to present the page in stand-alone fashion as it would be presented within the context of the complete document. These triplets are ignored by print servers.

BPG Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Page structured field is not present in the document.

Begin Page Segment (BPS)

The Begin Page Segment structured field begins a page segment. A page segment is a resource object that can be referenced from a page or overlay and that contains any mixture of:

- Bar code objects (BCOCA)
- Graphics objects (GOCA)
- Image objects (IOCA)

Objects in a page segment must specify an object area offset of zero so that they are positioned either at the origin of the including page or overlay coordinate system or at a reference point that is defined on the including page or overlay coordinate system by the Include Page Segment (IPS) structured field.

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.

Architecture Note: A migration form of the page segment resource object is supported in AFP environments and is defined in “AFP Page Segment” on page 519.

BPS (X'D3A85F') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A85F'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	PsegName		Name of the page segment	M	X'06'
8–*n*		Triplets		See “BPS Semantics” for triplet applicability.	O	X'10'

BPS Semantics

PsegName Is the name of the page segment. This name may not appear on more than one Begin Page Segment within the same resource group or a X’01’ exception condition exists.

A page segment resource definition must contain a subsequent matching End Page Segment structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.

Triplet	Type	Usage
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X′72′) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X′62′) triplet may occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

Page segments reside in external resource libraries or in resource groups. See “Resource Groups” on page 17 for details on locating resource objects within libraries or resource groups.

Application Notes:

1. For purposes of PSF resource management, the OEGs for all objects in a page segment must not contain MCF or MDR structured fields when the page segment is referenced with an IOB or IPS structured field.
2. In AFP environments, the following retired triplets are used on this structured field:
   - Object Checksum (X′63′) triplet; see “Object Checksum Triplet X'63'” on page 511
   - Object Origin Identifier (X′64′) triplet; see “Object Origin Identifier Triplet X'64'” on page 513

BPS Exception Condition Summary

- A X′08′ exception condition exists when the page segment resource definition is not terminated by a subsequent matching End Page Segment structured field.
- A X′01′ exception condition exists when multiple Begin Page Segment structured fields with the same name exist within the same resource group.

Begin Presentation Text Object (BPT)

The Begin Presentation Text Object structured field begins a presentation text object which becomes the current data object.

BPT (X'D3A89B') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A89B'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	PTdoName		Name of the presentation text data object	O	X'02'
8–*n*		Triplets		See “BPT Semantics” for triplet applicability.	O	X'10'

BPT Semantics

PTdoName Is the name of the presentation text data object.

The page, or overlay containing a Begin Presentation Text Object structured field must also contain a subsequent matching End Presentation Text Object structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Presentation Text Object structured field name and is used as the name of the presentation text data object.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X’72’) triplet may occur once. Assigns a date and time stamp to the object. See “Local Date and Time Stamp Triplet X'62'” on page 383.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X’62’) triplet may occur once. Assigns a universal date and time stamp to the object. See “Universal Date and Time Stamp Triplet X'72'” on page 394.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

When the BPT structured field is processed, all initial text conditions specified in the Presentation Text Descriptor (PTD) structured field are set prior to processing the text object.

Application Note: Whenever a BPT is encountered, AFP presentation servers set the following default page-level initial text conditions before the PTD initial conditions are set:

Table 15. Default BPT Page-Level Initial Text Conditions

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

BPT Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Presentation Text Object structured field is not present in the page, or overlay.

Begin Resource Group (BRG)

The Begin Resource Group structured field begins a resource group, which becomes the current resource group at the same level in the document hierarchy.

BRG (X'D3A8C6') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8C6'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	RGrpName		Name of the resource group	O	X'02'
8–*n*		Triplets		See "BRG Semantics" for triplet applicability.	O	X'10'

BRG Semantics

RGrpName Is the name of the resource group.

The print file, document, page, or data object containing the Begin Resource Group structured field must also contain a subsequent matching End Resource Group structured field, or a X'08' exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Begin Resource Group structured field name and is used as the name of the resource group.
X'02'	Fully Qualified Name	Optional. May occur more than once. The Fully Qualified Name type that may appear is **X'83'**—*Begin Document Name*. Specifies the name of a document that references resources contained in this resource group. See "Fully Qualified Name Triplet X'02'" on page 320.
X'62'	Local Date and Time Stamp	Optional. This triplet or the Universal Date and Time Stamp (X'72') triplet may occur once. Assigns a date and time stamp to the object. See "Local Date and Time Stamp Triplet X'62'" on page 383.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See "Comment Triplet X'65'" on page 385.

Triplet	Type	Usage
X'72'	Universal Date and Time Stamp	Optional. This triplet or the Local Date and Time Stamp (X'62') triplet may occur once. Assigns a universal date and time stamp to the object. See "Universal Date and Time Stamp Triplet X'72'" on page 394.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BRG Exception Condition Summary

- A X'08' exception condition exists when a subsequent matching End Resource Group structured field is not present in the print file, document, page, or data object.

Begin Resource (BRS)

The Begin Resource structured field begins an envelope that is used to carry resource objects in printfile-level (external) resource groups. Resource references in the data stream are matched against the resource identifier specified by the Begin Resource structured field.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a printfile that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same. For TrueType/OpenType fonts, optimal performance can be achieved by using UTF-16BE as the encoding scheme.

BRS (X'D3A8CE') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8CE'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	RSName		Identifier of the resource	M	X'02'
8–9					M	X'06'
10–*n*		Triplets		See “BRS Semantics” for triplet applicability.	M	X'14'

BRS Semantics

RSName Is the identifier used to select the resource. This identifier is matched against the resource reference in the data stream.

The resource group containing the Begin Resource structured field must also contain a subsequent matching End Resource structured field, or a X’08’ exception condition exists.

Triplets Appear in the Begin Resource structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	At least one occurrence of this triplet is mandatory if the BRS envelopes a TrueType Collection (TTC) file; may occur more than once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'6E'**—*Data-object Font Base Font Identifier*. This triplet may be specified on a BRS to indicate the following: • If the BRS envelopes a TrueType Collection (TTC) file, the FQN type X'6E' triplet specifies a base TrueType/OpenType font that is contained in the collection. The identifier may be specified in the following format. • If FQNFmt = X'00', the identifier is a character-encoded name. The character string that identifies the font must be the *full font name* specified in a name record in the mandatory Naming Table of the font file. This parameter is specified in a name record with Name ID 4. An example of a full font name is *Times New Roman Bold*. Each instance of the FQN X'6E' triplet with FQNFmt = X'00' is used to specify the full font name of the base font in a language used in the font's Naming Table. The character encoding is UTF-16, which matches the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001') in the Naming Table. The byte order is big endian. For example, if the font Naming Table contains two name records for the full font name (Name ID 4) - one in English - United States (LCID = X'0409') and one in German - Standard (LCID = X'0407'), both in the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001'), each of these names, encoded in UTF-16BE, is carried in a FQN X'6E' triplet on the BRS.
X'10'	Object Classification	Mandatory if the Resource Object Type triplet specifies ObjType = X'92', Object Container, in which case it must occur once. Characterizes and identifies the object data carried in the object container. See "Object Classification Triplet X'10'" on page 335.
X'21'	Resource Object Type (X'21') triplet; retired triplet, see "Resource Object Type Triplet X'21'" on page 506.	In AFP environments, one occurrence of this retired triplet is mandatory to identify the type of resource object delimited by the BRS.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID name*. This identifier overrides the Begin Resource structured field name and is used as the identifier of the resource. The identifier may be specified in one—and only one—of the following formats: • If FQNFmt = X'00', the identifier is a character-encoded name. See "External Resource Naming Conventions" on page 80 for a description of the naming conventions used in AFP environments. If the retired Resource Object Type (X'21') triplet specifies ObjType=X'92' - Object Container, and if the Object Classification Triplet indicates that the object in the container is a TrueType/OpenType font (TTF), the FQN type X'01' triplet, specified using FQNFmt = X'00', may occur more than once. In that case, each instance of the FQN type X'01' triplet is used to specify the full font name in a language used in the font naming table. The character encoding is UTF-16, which matches the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001') in the font's Naming Table. For example, if the font Naming Table contains two name records for the full font name (Name ID 4) - one in English - United States (LCID = X'0409') and one in German - Standard (LCID = X'0407'), both in the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001'), each of these names, encoded in UTF-16BE, is carried in a FQN type X'01' triplet on the BRS.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See "Comment Triplet X'65'" on page 385.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur more than once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'7E'**—*Data-object Font Linked Font Identifier*. This triplet may be specified on a BRS to indicate the following: • If the BRS envelopes a TrueType/OpenType font (TTF/OTF) file, the FQN type X'7E' triplet specifies a linked font for the base font. The order in which the FQN type X'7E' triplets are specified determines the order in which the linked fonts are processed. • If the BRS envelopes a TrueType Collection (TTC) file, the FQN type X'7E' triplet specifies a linked font for the base font that is identified with the immediately preceding FQN type X'6E' triplet. Note that if the base font is specified in multiple languages using multiple FQN type X'6E' triplets, each instance of the FQN type X'6E' triplet must be followed by the sequence of FQN type X'7E' triplets that identify the linked fonts for the base font. The identifier may be specified in the following format. • If FQNFmt = X'00', the identifier is a character-encoded name. The character string that identifies the font must be the *full font name* specified in a name record in the mandatory Naming Table of the font file. This parameter is specified in a name record with Name ID 4. An example of a full font name is *Times New Roman Bold*. The character encoding is UTF-16, which matches the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001') in the Naming Table. The byte order is big endian.

Using the BRS to Envelop Inline TrueType/OpenType Resources

TrueType/OpenType fonts (TTFs/OTFs), TrueType/OpenType fonts that are used as linked fonts, and TrueType/OpenType font collections (TTCs), may be carried in the resource group for a print file. This is called a print-file-level resource group, and these resources are said to be *inline*. When presentation servers search for a font that is referenced in the data stream, such a resource group is searched ahead of system-level resource libraries, and if an inline font is found it must be used in place of the system-level font. To support this hierarchy, presentation servers process a TrueType/OpenType font reference in an MDR for inline resources as follows:

1. The resource group - if present - is searched for a font (TTF/OTF) container or a collection (TTC) container that specifies a matching full font name.
   - A font container specifies the full font name using a FQN type X'01' triplet on the Begin Resource (BRS) structured field for the font container.
   - A collection container specifies the full font name of a font in the collection using a Data Object Font Base Font Identifier (X'6E') triplet on the BRS of the collection container.

   The first matching font container or collection container is used. If a collection containing the font is found, the complete TTC—if not already in the presentation device—is downloaded to the device, which must be able to index the required font in the collection. The font container or collection container may also specify one or more linked fonts for the referenced font.

- On a font container, linked fonts for the base font are specified with Data-object Font Linked Font Identifier (FQN type X'7E') triplets, which carry the full font name of the linked fonts, on the BRS of the font container.
- On a collection container, linked fonts are specified with Data-object Font Linked Font Identifier (FQN type X'7E') triplets that immediately follow the Data Object Font Base Font Identifier (X'6E') triplet for the base font on the BRS of the collection container. Note that if the base font is specified in multiple languages using multiple FQN type X'6E' triplets, each instance of the FQN type X'6E' triplet must be followed by the sequence of FQN type X'7E' triplets that identify the linked fonts for the base font.

The full font names for the linked fonts are used in turn to search the resource group for a font container or a collection container that carries a font that matches the full font name of the linked font. On a font container, the linked font name is matched against the FQN type X'01' triplet on the BRS; on a collection container it is matched against the FQN type X'6E' triplets on the BRS.

- The first matching font container or collection container is used, and its font is processed as a linked font for the base font. Multiple linked fonts may be specified, and the order in which they are specified on the BRS of the font container or collection container determines the order in which they are processed. The base font is always processed first, followed by the first-specified linked font, followed by the next-specified linked font, and so on. The last linked font is processed last.
- If a linked font cannot be found in either an inline font container or an inline collection container, the full font name of the linked font is used to index the RAT to locate the linked font in a resource library. If a specified linked font cannot be found in the resource group or in a resource library, a X'04' exception condition exists.

Only one level of linking is supported. That is, if a linked font specifies its own linked fonts, either with FQN type X'7E' triplets on its inline container or with linked font pointers in the RAT, these ‘secondary’ linked fonts are not processed as linked fonts for the original base font.

2. If a font matching the MDR reference is not found in an inline font container or in an inline collection container, the presentation server accesses the RAT with the full font name to locate the referenced font in a resource library. In this case, all linked fonts are specified in the RAT repeating group for the referenced font, and the order in which they are specified determines the order in which they are processed. Both inline linked fonts and library-based linked fonts are used, and the print-file-level resource group is always searched for linked fonts ahead of the resource library. The resource group search includes font containers, in which case the linked font name is matched against the FQN type X'01' triplet on the BRS of the font container, and collection containers, in which case the linked font name is matched against the FQN type X'6E' triplets on the BRS of the collection container.

BRS Exception Condition Summary

- A X’08’ exception condition exists when the Begin Resource structured field is not followed by a subsequent End Resource structured field in the same resource group.

Begin Resource Environment Group (BSG)

The Begin Resource Environment Group structured field begins a Resource Environment Group (REG), which defines a subset of the resources required for a document or for a group of pages in a document. The scope of the Resource Environment Group is the group of pages that follow, up to the next REG, which is a complete replacement for the current REG, or the end of the document, whichever occurs first.

Note: Resources that are mapped in a REG must still be mapped in the AEG for the page that uses the resources.

BSG (X'D3A8D9') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8D9'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	REGName		Name of the resource environment group	O	X'02'
8–*n*		Triplets		See “BSG Semantics” for triplet applicability.	O	X'10'

BSG Semantics

REGName Is the name of the resource environment group.

The document containing the Begin Resource Environment Group structured field must also contain a subsequent matching End Resource Environment Group structured field, or a X’08’ exception condition exists.

Triplets Appear as follows:

Triplets	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'65'	Comment	Optional. May occur more than once. Carries unarchitected data. See “Comment Triplet X'65'” on page 385.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

BSG Exception Condition Summary

- A X’08’ exception condition exists when a subsequent matching End Resource Environment Group structured field is not present in the document.

Color Attribute Table (CAT)

The Color Attribute Table structured field contains the data for a color attribute table resource object.

Note: The CAT structured field is used only in MO:DCA-L data streams.

CAT (X'D3B077') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B077'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	CATData		Up to 32759 bytes of color table data	O	X'00'

CAT Semantics

CATData Contains the color attribute table data. See “The Color Table Resource” on page 483 for a detailed description.

Container Data Descriptor (CDD)

The Container Data Descriptor structured field specifies control information for a presentation data object that is carried in an object container.

CDD (X'D3A692') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A692'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–11				Retired parameters; see "Retired Parameters" on page 515	M	X'06'
12–*n*		Triplets		See "CDD Semantics" for triplet applicability.	O	X'00'

CDD Semantics

Triplets Specify control information for object data. To be defined as required by the object data.

Notes:

1. For static presentation objects, a presentation space size is required for a scale-to-fit or scale-to-fill mapping of the object presentation space to the object area. See "Object Type Identifiers" on page 535 for information on how the presentation space size is specified by various objects. If the presentation space size is not specified by the object, the architected default is the presentation space size of the including page or overlay.
2. This structured field is not applicable to non-presentation objects and may be ignored if it appears in the object container for such objects.

End Active Environment Group (EAG)

The End Active Environment Group structured field terminates the definition of an Active Environment Group initiated by a Begin Active Environment Group structured field.

EAG (X'D3A9C9') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9C9'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	AEGName		Name of the active environment group	O	X'02'

EAG Semantics

AEGName Is the name of the active environment group being terminated. If a name is specified, it must match the name in the most recent Begin Active Environment Group structured field in the page or a X’01’ exception condition exists. If the first two bytes in AEGName contain the value X'FFFF', the name matches any name specified on the Begin Active Environment Group structured field that initiated the current definition.

A matching Begin Active Environment Group structured field must appear within the page at some location preceding the End Active Environment Group structured field, or a X’20’ exception condition exists.

EAG Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Active Environment Group structured field.
- A X’20’ exception condition exists when not preceded by a matching Begin Active Environment Group structured field.

End Bar Code Object (EBC)

The End Bar Code Object structured field terminates the current bar code object initiated by a Begin Bar Code Object structured field.

EBC (X'D3A9EB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9EB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	BCdoName		Name of the bar code data object	O	X'02'
8–*n*		Triplets		See "EBC Semantics" for triplet applicability.	O	X'10'

EBC Semantics

BCdoName Is the name of the bar code data object being terminated. If a name is specified, it must match the name in the most recent Begin Bar Code Object structured field in the page, overlay, or resource group, or a X'01' exception condition exists. If the first two bytes of BCdoName contain the value X'FFFF', the name matches any name specified on the Begin Bar Code Object structured field that initiated the current definition.

A matching Begin Bar Code Object structured field must appear within the containing structure at some location preceding the End Bar Code Object structured field, or a X'20' exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Bar Code Object structured field name and is used as the name of the bar code data object being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

EBC Exception Condition Summary

- A X'01' exception condition exists when a name is specified that does not match the name on the most recent Begin Bar Code Object structured field.
- A X'20' exception condition exists when the End Bar Code Object structured field is not preceded by a matching Begin Bar Code Object structured field.

End Color Attribute Table (ECA)

The End Color Attribute Table structured field terminates the Color Attribute Table resource object initiated by a Begin Color Attribute Table structured field.

Note: The ECA structured field is used only in MO:DCA-L data streams.

ECA (X'D3A977') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A977'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	CATName		Name of the color attribute table	O	X'02'
8–*n*		Triplets		See “ECA Semantics” for triplet applicability.	O	X'10'

ECA Semantics

CATName Is the name of the color attribute table being terminated. If a name is specified, it must match the name in the most recent Begin Color Attribute Table structured field in the resource group or a X’01’ exception condition exists. If the first two bytes of CATName contain the value X'FFFF', the name matches any name specified on the Begin Color Attribute Table structured field that initiated the current definition.

A matching Begin Color Attribute Table structured field must appear within the resource group at some location preceding the End Color Attribute Table structured field, or a X’20’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Color Attribute Table structured field name and is used as the name of the color attribute table being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

ECA Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Color Attribute Table structured field.

- A X′20′ exception condition exists when the End Color Attribute Table structured field is not preceded by a matching Begin Color Attribute Table structured field.

End Document Environment Group (EDG)

The End Document Environment Group structured field terminates the definition of a document environment group initiated by a Begin Document Environment Group structured field.

EDG (X'D3A9C4') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9C4'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	DEGName		Name of the document environment group	O	X'02'

EDG Semantics

DEGName Is the name of the document environment group being terminated. If a name is specified, it must match the name in the most recent Begin Document Environment Group structured field in the form map or a X’01’ exception condition exists. If the first two bytes in DEGName contain the value X'FFFF', the name matches any name specified on the Begin Document Environment Group structured field that initiated the current definition.

A matching Begin Document Environment Group structured field must appear at some location within the form map preceding the End Document Environment Group structured field, or a X’20’ exception condition exists.

EDG Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Document Environment Group structured field.
- A X’20’ exception condition exists when the End Document Environment Group structured field is not preceded by a matching Begin Document Environment Group structured field.

End Document Index (EDI)

The End Document Index structured field terminates the document index initiated by a Begin Document Index structured field.

EDI (X'D3A9A7') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9A7'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	IndxName		Name of the document index	O	X'02'
8–*n*		Triplets		See "EDI Semantics" for triplet applicability.	O	X'10'

EDI Semantics

IndxName Is the name of the document index being terminated. If a name is specified, it must match the name in the most recent Begin Document Index structured field in the print file or document, or a X'01' exception condition exists. If the first two bytes of IndxName contain the value X'FFFF', the name matches any name specified on the Begin Document Index structured field that initiated the current definition.

A matching Begin Document Index structured field must appear within the print file or document at some location preceding the End Document Index structured field, or a X'20' exception condition exists.

Triplets Appear as follows:

Triplet	**Type**	**Usage**
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Document Index structured field name and is used as the name of the document index being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

EDI Exception Condition Summary

- A X'01' exception condition exists when a name is specified that does not match the name on the most recent Begin Document Index structured field.
- A X'20' exception condition exists when the End Document Index structured field is not preceded by a matching Begin Document Index structured field.

End Document (EDT)

The End Document structured field terminates the MO:DCA document data stream initiated by a Begin Document structured field.

EDT (X'D3A9A8') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9A8'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	DocName		Name of the document	O	X'02'
8–*n*		Triplets		See "EDT Semantics" for triplet applicability.	O	X'10'

EDT Semantics

DocName Is the name of the document being terminated. If a name is specified, it must match the name in the most recent Begin Document structured field in the data stream or a X'01' exception condition exists. If the first two bytes of DocName contain the value X'FFFF', the name matches any name specified on the Begin Document structured field that initiated the current definition.

A matching Begin Document structured field must appear within the data stream at some location preceding the End Document structured field, or a X'20' exception condition exists.

Triplets Appear as follows:

Triplet	**Type**	**Usage**
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The only Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Document structured field name and is used as the name of the document being terminated.

Note: If a triplet is included on this structured field, the optional DocName positional parameter becomes mandatory.

EDT Exception Condition Summary

- A X'01' exception condition exists when a name is specified that does not match the name on the most recent Begin Document structured field.
- A X'20' exception condition exists when the End Document structured field is not preceded by a matching Begin Document structured field.

End Form Map (EFM)

The End Form Map structured field terminates the form map object initiated by a Begin Form Map structured field

EFM (X'D3A9CD') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9CD'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	FMName		Name of the form map	O	X'02'

EFM Semantics

FMName Is the name of the form map being terminated. If a name is specified, it must match the name in the most recent Begin Form Map structured field or a X’01’ exception condition exists. If the first two bytes of FMName contain the value X'FFFF', the name matches any name specified on the Begin Form Map structured field that initiated the current definition.

A matching Begin Form Map structured field must appear at some location preceding the End Form Map structured field, or a X’20’ exception condition exists.

EFM Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Form Map structured field.
- A X’20’ exception condition exists when the End Form Map structured field is not preceded by a matching Begin Form Map structured field.

End Graphics Object (EGR)

The End Graphics Object structured field terminates the current graphics object initiated by a Begin Graphics Object structured field.

EGR (X'D3A9BB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9BB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	GdoName		Name of the graphics data object	O	X'02'
8–*n*		Triplets		See "EGR Semantics" for triplet applicability.	O	X'10'

EGR Semantics

GdoName Is the name of the graphics data object being terminated. If a name is specified, it must match the name in the most recent Begin Graphics Object structured field in the containing page, overlay, or resource group, or a X'01' exception condition exists. If the first two bytes of GdoName contain the value X'FFFF', the name matches any name specified on the Begin Graphics Object structured field that initiated the current definition.

A matching Begin Graphics Object structured field must appear within the containing structure at some location preceding the End Graphics Object structured field, or a X'20' exception condition exists.

Triplets Appear as follows:

Triplet	**Type**	**Usage**
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Graphics Object structured field name and is used as the name of the graphics data object being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

EGR Exception Condition Summary

- A X'01' exception condition exists when a name is specified that does not match the name on the most recent Begin Graphics Object structured field.
- A X'20' exception condition exists when the End Graphics Object structured field is not preceded by a matching Begin Graphics Object structured field.

End Image Object (EIM)

The End Image Object structured field terminates the current image object initiated by a Begin Image Object structured field.

EIM (X'D3A9FB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9FB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	IdoName		Name of the image data object	O	X'02'
8–*n*		Triplets		See “EIM Semantics” for triplet applicability.	O	X'10'

EIM Semantics

IdoName Is the name of the image data object being terminated. If a name is specified, it must match the name in the most recent Begin Image Object structured field in the containing page, overlay, or resource group, or a X’01’ exception condition exists. If the first two bytes of IdoName contain the value X'FFFF', the name matches any name specified on the Begin Image Object structured field that initiated the current definition.

A matching Begin Image Object structured field must appear within the containing structure at some location preceding the End Image Object structured field, or a X’20’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Image Object structured field name and is used as the name of the image data object being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

EIM Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Image Object structured field.
- A X’20’ exception condition exists when the End Image Object structured field is not preceded by a matching Begin Image Object structured field.

End Medium Map (EMM)

The End Medium Map structured field terminates the medium map object initiated by a Begin Medium Map structured field

EMM (X'D3A9CC') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9CC'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	MMName		Name of the medium map	O	X'02'

EMM Semantics

MMName Is the name of the medium map being terminated. If a name is specified, it must match the name in the most recent Begin Medium Map structured field or a X’01’ exception condition exists. If the first two bytes of MMName contain the value X'FFFF', the name matches any name specified on the Begin Medium Map structured field that initiated the current definition.

A matching Begin Medium Map structured field must appear at some location preceding the End Medium Map structured field, or a X’20’ exception condition exists.

EMM Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Medium Map structured field.
- A X’20’ exception condition exists when the End Medium Map structured field is not preceded by a matching Begin Medium Map structured field.

End Overlay (EMO)

The End Overlay structured field terminates the overlay resource object initiated by a Begin Overlay structured field.

EMO (X'D3A9DF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9DF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	OvlyName		Name of the overlay	O	X'02'
8–*n*		Triplets		See “EMO Semantics” for triplet applicability.	O	X'10'

EMO Semantics

OvlyName Is the name of the overlay that is being terminated. If a name is specified, it must match the name in the most recent Begin Overlay structured field in the resource group or a X’01’ exception condition exists. If the first two bytes of OvlyName contain the value X'FFFF', the name matches any name specified on the Begin Overlay structured field that initiated the current definition.

A matching Begin Overlay structured field must appear within the resource group at some location preceding the End Overlay structured field, or a X’20’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Overlay structured field name and is used as the name of the overlay being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

EMO Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Overlay structured field.
- A X’20’ exception condition exists when the End Overlay structured field is not preceded by a matching Begin Overlay structured field.

End Named Page Group (ENG)

The End Named Page Group structured field terminates a page group that was initiated by a Begin Named Page Group structured field.

ENG (X'D3A9AD') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9AD'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	PGrpName		Name of the overlay	O	X'02'
8–*n*		Triplets		See “ENG Semantics” for triplet applicability.	O	X'10'

ENG Semantics

PGrpName Is the name of the page group that is being terminated. If a name is specified, it must match the name in the most recent Begin Named Page Group structured field in the document or a X’01’ exception condition exists. If the first two bytes of PGrpName contain the value X'FFFF', the name matches any name specified on the Begin Named Page Group structured field that initiated the current definition.

A matching Begin Named Page Group structured field must appear within the document at some location preceding the End Named Page Group structured field, or a X’20’ exception condition exists.

Triplets Appear in the End Named Page Group structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID name*. This GID overrides the End Named Page Group structured field name and is used as the name of the page group being terminated.

Note: If a triplet is included on this structured field, the optional PGrpName positional parameter becomes mandatory.

ENG Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Named Page Group structured field.
- A X’20’ exception condition exists when the End Named Page Group structured field is not preceded by a matching Begin Named Page Group structured field.

End Object Container (EOC)

The End Object Container structured field terminates an object container initiated by a Begin Object Container structured field.

EOC (X'D3A992') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A992'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	ObjCName		Name of the object container	O	X'02'
8–*n*		Triplets		See “EOC Semantics” for triplet applicability.	O	X'10'

EOC Semantics

ObjCName Is the name of the object container that is being terminated. If a name is specified, it must match the name in the most recent Begin Object Container structured field or a X’01’ exception condition exists. If the first two bytes of ObjCName contain the value X'FFFF', the name matches any name specified on the Begin Object Container structured field that initiated the current definition.

A matching Begin Object Container structured field must appear at some location preceding the End Object Container structured field, or a X’20’ exception condition exists.

Triplets Appear in the End Object Container structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID name*. This GID overrides the End Object Container structured field name and is used as the name of the object container being terminated.

Note: If a triplet is included on this structured field, the optional ObjCName positional parameter becomes mandatory.

EOC Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Object Container structured field.
- A X’20’ exception condition exists when the End Object Container structured field is not preceded by a matching Begin Object Container structured field.

End Object Environment Group (EOG)

The End Object Environment Group structured field terminates the definition of an Object Environment Group initiated by a Begin Object Environment Group structured field.

EOG (X'D3A9C7') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9C7'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	OEGName		Name of the object environment group	O	X'02'

EOG Semantics

OEGName Is the name of the object environment group that is being terminated. If a name is specified, it must match the name in the most recent Begin Object Environment Group structured field in the object or a X’01’ exception condition exists. If the first two bytes of OEGName contain the value X'FFFF', the name matches any name specified on the Begin Object Environment Group structured field that initiated the current definition.

A matching Begin Object Environment Group structured field must appear within the object at some location preceding the End Object Environment Group structured field, or a X’20’ exception condition exists.

EOG Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Object Environment Group structured field.
- A X’20’ exception condition exists when the End Object Environment Group structured field is not preceded by a matching Begin Object Environment Group structured field.

End Page (EPG)

The End Page structured field terminates the current presentation page definition initiated by a Begin Page structured field.

EPG (X'D3A9AF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9AF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	PageName		Name of the page	O	X'02'
8–*n*		Triplets		See "EPG Semantics" for triplet applicability.	O	X'10'

EPG Semantics

PageName Is the name of the page that is being terminated. If a name is specified, it must match the name in the most recent Begin Page structured field in the document or a X'01' exception condition exists. If the first two bytes of PageName contain the value X'FFFF', the name matches any name specified on the Begin Page structured field that initiated the current definition.

A matching Begin Page structured field must appear within the document at some location preceding the End Page structured field, or a X'20' exception condition exists.

Triplets Appear as follows:

Triplet	**Type**	**Usage**
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Page structured field name and is used as the name of the page being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

EPG Exception Condition Summary

- A X'01' exception condition exists when a name is specified that does not match the name on the most recent Begin Page structured field.
- A X'20' exception condition exists when the End Page structured field is not preceded by a matching Begin Page structured field.

End Page Segment (EPS)

The End Page Segment structured field terminates the page segment resource object initiated by a Begin Page Segment structured field.

EPS (X'D3A95F') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A95F'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	PsegName		Name of the page segment	O	X'02'

EPS Semantics

PsegName Is the name of the page segment that is being terminated. If a name is specified, it must match the name in the most recent Begin Page Segment structured field or a X’01’ exception condition exists. If the first two bytes of PsegName contain the value X'FFFF', the name matches any name specified on the Begin Page Segment structured field that initiated the current definition.

A matching Begin Page Segment structured field must appear at some location preceding the End Page Segment structured field, or a X’20’ exception condition exists.

EPS Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Page Segment structured field.
- A X’20’ exception condition exists when the End Page Segment structured field is not preceded by a matching Begin Page Segment structured field.

End Presentation Text Object (EPT)

The End Presentation Text Object structured field terminates the current presentation text object initiated by a Begin Presentation Text Object structured field.

EPT (X'D3A99B') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A99B'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	PTdoName		Name of the presentation text data object	O	X'02'
8–*n*		Triplets		See “EPT Semantics” for triplet applicability.	O	X'10'

EPT Semantics

PTdoName Is the name of the presentation text data object that is being terminated. If a name is specified, it must match the name in the most recent Begin Presentation Text Object structured field in the page, or overlay, or a X’01’ exception condition exists. If the first two bytes of PTdoName contain the value X'FFFF', the name matches any name specified on the Begin Presentation Text Object structured field that initiated the current definition.

A matching Begin Presentation Text Object structured field must appear within the containing structure at some location preceding the End Presentation Text Object structured field, or a X’20’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Presentation Text Object structured field name and is used as the name of the presentation text data object being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

EPT Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Presentation Text Object structured field.

- A X′20′ exception condition exists when the End Presentation Text Object structured field is not preceded by a matching Begin Presentation Text Object structured field.

End Resource Group (ERG)

The End Resource Group structured field terminates the definition of a resource group initiated by a Begin Resource Group structured field.

ERG (X'D3A9C6') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9C6'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	RGrpName		Name of the resource group	O	X'02'
8–*n*		Triplets		See “ERG Semantics” for triplet applicability.	O	X'10'

ERG Semantics

RGrpName Is the name of the resource group that is being terminated. If a name is specified, it must match the name in the most recent Begin Resource Group structured field in the print file, document, page, or data object, or a X’01’ exception condition exists. If the first two bytes of RGrpName contain the value X'FFFF', the name matches any name specified on the Begin Resource Group structured field that initiated the current definition.

A matching Begin Resource Group structured field must appear within the print file, document, page, or data object at some location preceding the End Resource Group structured field, or a X’20’ exception condition exists.

Triplets Appear as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the End Resource Group structured field name and is used as the name of the resource group being terminated.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

ERG Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Resource Group structured field.
- A X’20’ exception condition exists when the End Resource Group structured field is not preceded by a matching Begin Resource Group structured field.

End Resource (ERS)

The End Resource structured field terminates an envelope that is used to carry resource objects in external (printfile-level) resource groups. The envelope is initiated by a Begin Resource (BRS) structured field.

ERS (X'D3A9CE') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9CE'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	RSName		Name of the resource	O	X'02'

ERS Semantics

RSName Is the name of the resource being terminated. If a name is specified, it must match the name in the most recent Begin Resource structured field. If the first two bytes in RSName contain the value X'FFFF', the name matches any name specified on the Begin Resource structured field that initiated the current definition.

A matching Begin Resource structured field must appear within the resource group at some location preceding the End Resource structured field, or a X’20’ exception condition exists.

ERS Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Resource structured field.
- A X’20’ exception condition exists when the End Resource structured field is not preceded by a matching Begin Resource structured field.

End Resource Environment Group (ESG)

The End Resource Environment Group structured field terminates the definition of a Resource Environment Group initiated by a Begin Resource Environment Group structured field.

ESG (X'D3A9D9') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9D9'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	REGName		Name of the resource environment group	O	X'02'

ESG Semantics

REGName Is the name of the resource environment group being terminated. If a name is specified, it must match the name in the most recent Begin Resource Environment Group structured field in the document or a X’01’ exception condition exists. If the first two bytes in REGName contain the value X'FFFF', the name matches any name specified on the Begin Resource Environment Group structured field that initiated the current definition.

A matching Begin Resource Environment Group structured field must appear within the document at some location preceding the End Resource Environment Group structured field, or a X’20’ exception condition exists.

ESG Exception Condition Summary

- A X’01’ exception condition exists when a name is specified that does not match the name on the most recent Begin Resource Environment Group structured field.
- A X’20’ exception condition exists when the End Resource Environment Group structured field is not preceded by a matching Begin Resource Environment Group structured field.

Graphics Data (GAD)

The Graphics Data structured field contains the data for a graphics object.

GAD (X'D3EEBB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3EEBB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	UNDF	GOCAdat		Up to 32759 bytes of GOCA-defined data	O	X'00'

GAD Semantics

GOCAdat Contains the GOCA-defined data. See the MO:DCA environment appendixes in the *Graphics Object Content Architecture Reference* and the *Graphics Object Content Architecture for Advanced Function Presentation Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Graphics Data Descriptor (GDD)

The Graphics Data Descriptor structured field contains the descriptor data for a graphics object.

GDD (X'D3A6BB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A6BB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–*n*	UNDF	GOCAdes		Up to 32759 bytes of GOCA-defined descriptor data	O	X'00'

GDD Semantics

GOCAdes Contains the GOCA-defined descriptor data. See the MO:DCA environment appendixes in the *Graphics Object Content Architecture Reference* and the *Graphics Object Content Architecture for Advanced Function Presentation Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Image Data Descriptor (IDD)

The Image Data Descriptor structured field contains the descriptor data for an image data object.

IDD (X'D3A6FB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A6FB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	IOCAdes		Up to 32759 bytes of IOCA-defined descriptor data	O	X'00'

IDD Semantics

IOCAdes Contains the IOCA-defined descriptor data. See the MO:DCA environment appendix in the *Image Object Content Architecture Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Index Element (IEL)

The Index Element structured field identifies begin structured fields for use within a document index.

IEL (X'D3B2A7') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B2A7'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*		Triplets		See “IEL Semantics” for triplet applicability.	M	X'14'

IEL Semantics

Triplets

Appear in the Index Element structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'CA'**—*Index Element GID*, which is used as the name of this Index Element structured field.
X'2D'	Object Byte Offset	Mandatory. Must occur once. Specifies the offset, in bytes, from the beginning of the document to the indexed object. See “Object Byte Offset Triplet X'2D'” on page 353.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'02'	Fully Qualified Name	Optional. One of the following Fully Qualified Name types may appear on the Index Element structured field. • **X'0D'**—*Begin Page Group Name*. Specifies the name of the page group indexed by the Index Element structured field. • **X'87'**—*Begin Page Name*. Specifies the name of the page indexed by the Index Element structured field.
X'02'	Fully Qualified Name	Optional. May occur once. The Fully Qualified Name type that may appear is **X'8D'**—*Begin Medium Map Name*. For a page-level IEL, specifies the name of the medium map that is active for presenting the indexed page on a physical medium. For a page-group-level IEL, specifies the name of the medium map that is active for presenting the first page in the indexed page group on a physical medium.

Triplet	Type	Usage
X'56'	Medium Map Page Number	Optional. May occur once. For a page-level IEL, specifies the sequence number of the indexed page in the set of sequential pages controlled by the active medium map. For a page-group-level IEL, specifies the sequence number of the first page-group page in the set of sequential pages controlled by the medium map that is active at the beginning of the indexed page group. See “Medium Map Page Number Triplet X'56'” on page 374. If the Page Position Information (X'81') triplet is also specified on this IEL, it overrides the Medium Map Page Number (X'56') triplet.
X'57'	Object Byte Extent	Optional. May occur once. Specifies the extent, in bytes, of the indexed object. See “Object Byte Extent Triplet X'57'” on page 375.
X'58'	Object Structured Field Offset	Optional. May occur once. Specifies the offset, in structured fields, from the beginning of the document to the indexed object. See “Object Structured Field Offset Triplet X'58'” on page 376.
X'59'	Object Structured Field Extent	Optional. May occur once. Specifies the extent, in structured fields, of the indexed object. See “Object Structured Field Extent Triplet X'59'” on page 377.
X'5A'	Object Offset	Optional. May occur once for each object type counted. Specifies how many objects of a particular type precede the indexed object in the document. See “Object Offset Triplet X'5A'” on page 378.
X'5E'	Object Count	Optional. May occur once for each subordinate object type counted. Specifies how many subordinate objects of a particular type are contained within the indexed object. See “Object Count Triplet X'5E'” on page 381.
X'81'	Page Position Information	Optional. May occur once. For a page-level IEL, specifies the PGP repeating group that is used to view the page and its PMC overlay data. For a page-group-level IEL, specifies the PGP repeating group that is used to view the first page in the group. The PGP is specified in the medium map referenced by a FQN type X'8D'—Begin Medium Map Reference triplet. If the X'81' triplet is specified, it overrides a Medium Map Page Number (X'56') triplet. See “Page Position Information Triplet X'81'” on page 403.
X'83'	Presentation Control	Optional. May occur once. Specified on a page-level IEL to indicate whether the page is intended to be viewed. If this triplet is not specified, the architected default is that the page is intended to be viewed. See “Presentation Control Triplet X'83'” on page 405.

IEL Exception Condition Summary

- A X’01’ exception condition exists when multiple type X’CA’ (Index Element GID) Fully Qualified Name triplets appear.
- A X'01' exception condition exists when the same object type is counted in more than one X'5A' triplet.
- A X'01' exception condition exists when the same subordinate object type is counted in more than one X'5E' triplet.

Invoke Medium Map (IMM)

The Invoke Medium Map structured field identifies the medium map that is to become active for the document. An Invoke Medium Map structured field affects the document's current environment. The medium map's effect on current environment parameter values lasts until a new medium map is invoked.

The processing system's form map is searched for the specified medium map unless the IMM directly follows an internal medium map, in which case the IMM can reference and activate that internal medium map. An IMM that does not follow an internal medium map cannot be used to reference an internal medium map elsewhere in the document and is assumed to reference a medium map in the processing system's form map.

If a document does not invoke a medium map by name, and if it does not include an internal medium map, the first medium map in the selected form map controls document presentation.

IMM (X'D3ABCC') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABCC'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	MMPName		Name of the medium map to be invoked	M	X'0E'
8–*n*		Triplets		See "IMM Semantics" for triplet applicability.	O	X'10'

IMM Semantics

MMPName Is the name of the medium map.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.

Effect On Parameter Values

The parameter values contained in the structured fields within the invoked medium map replace those that were established previously by structured fields having the same code points.

Parameter Conflict Resolution

All conflicts with existing environment settings are resolved in favor of the medium map specified by the Invoke Medium Map structured field.

Application Note: Page groups are often processed in stand-alone fashion, that is, they are indexed, retrieved, and presented outside the context of the containing document. While the pages in the group are independent of each other and of any other pages in the document, their formatting on media depends on when the last medium map was invoked and on how many pages precede the BNG since this invocation. To make the media formatting of page groups self-contained, a medium map should be invoked at the beginning of the page group between the Begin Named Group (BNG) structured field and the first Begin Page (BPG) structured field. If this is not done, the presentation system may need to "play back" all pages between the invocation of the active medium map and the BNG to determine media formatting such as sheet-side and partition number for the first page in the group.

It is therefore *strongly* recommended that in environments where stand-alone page group processing is required or anticipated, page groups are built with an Invoke Medium Map (IMM) structured field specified after the BNG and before the first BPG. IBM AFP applications that generate page groups will support a user option that ensures that an IMM is specified after BNG and before the first BPG, and IBM AFP archive servers will expect an IMM there and may not present the page group correctly if none is found. However, note that this may cause the complete document to print differently.

A newer method to specify how a page or page group should be formatted involves use of the Page Position Information (X'81') triplet. This triplet may be specified on a BPG and indicates the repeating group in the PGP structured field in the active medium map that should be used to format the page.

Include Object (IOB)

An Include Object structured field references an object on a page or overlay. It 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.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

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	Exc
0–7	CHAR	ObjName		Name of the object	M	X'06'
8				Reserved; must be zero	M	X'06'
9	CODE	ObjType	X'5F', X'92', X'BB', X'EB', 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	X'06'
10–12	SBIN	XoaOset	-32768–32767	X-axis origin of the object area	M	X'06'
			X'FFFFFF'	Use the X-axis origin defined in the object		
13–15	SBIN	YoaOset	-32768–32767	Y-axis origin of the object area	M	X'06'
			X'FFFFFF'	Use the Y-axis origin defined in the object		
16–17	CODE	XoaOrent	X'0000', X'2D00', X'5A00', 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'06'
			X'FFFFFF'	Use the X-axis rotation defined in the object		

Offset	Type	Name	Range	Meaning	M/O	Exc
18–19	CODE	YoaOrent	X'0000', X'2D00', X'5A00', 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'06'
			X'FFFF'	Use the Y-axis rotation defined in the object		
Note: See "IOB Semantics" for valid combinations of the XoaOrent and YoaOrent values.						
20–22	SBIN	XocaOset	-32768–32767	X-axis origin for object content	M	X'06'
			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'06'
			X'FFFFFF'	Use the Y-axis origin defined in the object		
26	CODE	RefCSys	X'01'	Reference coordinate system: **X'01'** Page or overlay coordinate system	M	X'06'
27–*n*		Triplets		See "IOB Semantics" for triplet applicability.	M	X'14'

IOB Semantics

ObjName Is the name of the object being referenced. This name may be a file name or any other identifier associated with the object data.

ObjType Identifies the type of object being referenced.

Value	Description
X'5F'	Page segment object.

Note: The page segment must be a MO:DCA page segment. AFP migration page segments are not supported in the IOB. For a definition of MO:DCA page segments, see "Page Segment Objects" on page 77. For a definition of AFP page segments, see "AFP Page Segment" on page 519.

Application Note: For purposes of Print Services Facility resource management, the Object Environment Groups (OEGs) for all objects in the page segment must not contain MCF or MDR structured fields when the page segment is referenced with an IOB structured field.

X'92' Other object data. The object data to be included is a paginated presentation object whose format may or may not be defined by an IBM presentation architecture. The object data is characterized and identified by a mandatory Object Classification (X'10') triplet, which must specify the registered object-type OID for the object type and must characterize the object as being a presentation object. This triplet also specifies whether the object data is carried in a MO:DCA object container, whether it is unwrapped object data, or whether the container structure of the object data is unknown.

This value is not used for OCA objects since they are referenced using object-specific values for the ObjType parameter.

To ensure proper presentation of the object, the object-type OID must be supported by the MO:DCA-P presentation system. This means that the object-type OID is supported by the presentation server, and that it is either supported directly by the presentation device, or that it can be transformed by the server into a format that is directly supported by the presentation device.

Application Note: To see which object-type OIDs are supported by the presentation system, consult the product documentation. In particular, to see which object-type OIDs are supported by AFP presentation servers, see the *Advanced Function Presentation: Programming Guide and Line Data Reference*.

See “Non-OCA Object Types Supported by the IOB Structured Field” on page 543 for a list of object types that may be included with an IOB in MO:DCA-P data streams.

X'BB' Graphics (GOCA) object with MO:DCA object syntax as defined in “Graphics Objects” on page 90.

X'EB' Bar code (BCOCA) object with MO:DCA object syntax as defined in “Bar Code Objects” on page 88.

X'FB' Image (IOCA) object with MO:DCA object syntax as defined in “Image Objects” on page 94.

All others
Reserved

XoaOset 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 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 not included in the allowed range.

If the object does not specify the X-axis offset in an OEG, the architected default is X'000000'.

YoaOset 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 not included in the allowed range.

If the object does not specify the Y-axis offset in an OEG, the architected default is X'000000'.

XoaOrent 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).

YoaOrent 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).

Notes:

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

Table 16. IOB: Valid Values for XoaOrent and YoaOrent

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

2. If the object area orientation is such that the sum of the object area origin offset and the object area extent exceeds the size of the including presentation space in either the X or Y direction, all of the object area will not fit in the including presentation space. The including presentation space in this case is the page or overlay presentation space. If an attempt is made to actually present data in the portion of the object area that falls outside the including presentation space, that portion of the data is not presented, and a X'01' exception condition exists.

XocaOset Used in *position* and *position and trim* mappings to specify the offset along the X axis of the object area coordinate system, X_{oa}, to the X origin of the object content. The value for this parameter is expressed in terms of the number of object area 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.

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 not included in the allowed range.

If the object does not specify the X-axis offset in an OEG, the architected default is X'000000'.

YocaOset Used in *position* and *position and trim* mappings to specify the offset along the Y axis of the object area coordinate system, Y_{oa}, to the Y origin of the object content. The value for this parameter is expressed in terms of the number of object area 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.

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 not included in the allowed range.

If the object does not specify the Y-axis offset in an OEG, the architected default is X'000000'.

RefCSys Specifies the coordinate system used to position the object area.

Value	**Description**
X'00'	Retired for private use. **Architecture Note:** This value is used in AFP line-data environments to position and rotate the object area with respect to the current text (I,B) coordinate system. For more information, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.
X'01'	Page or overlay coordinate system
All others	Reserved

Triplets Appear in the Include Object structured field as follows:

Triplet	Type	Usage
X'10'	Object Classification	Mandatory for *ObjType* = X'92', other object data, in which case it must occur once. Specifies information used to characterize and identify the object data to be included. The included object must be a presentation object. See "Object Classification Triplet X'10'" on page 335.
X'4B'	Measurement Units	Mandatory if the IOB specifies an override for any of the following parameters: • XocaOset • YocaOset • XoaSize, specified in the Object Area Size (X'4C') triplet • YoaSize, specified in the Object Area Size (X'4C') triplet In this case, this triplet occurs once and defines the measurement units for the override values. See "Measurement Units Triplet X'4B'" on page 364.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID name*. This identifier overrides the Include Object structured field name and is used as the identifier of the object. The identifier may be specified in one - and only one - of the following formats: • If FQNFmt = X'00', the identifier is a character-encoded name. See "External Resource Naming Conventions" on page 80 for a description of the naming conventions used in AFP environments.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur more than once. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'DE'**—*Data Object External Resource Reference*. Specifies the external identifier of a resource object that is used by the object being included. The identifier is used by the presentation system to locate the resource object in the resource hierarchy. The identifier may be specified in one of the following two formats, but not in both formats: • If FQNFmt = X'00', the identifier is a character-encoded name. See “External Resource Naming Conventions” on page 80 for a description of the naming conventions used in AFP environments. • If FQNFmt = X'10', the identifier is an ASN.1 OID encoded using the definite short form. This format provides a unique and system-independent method to identify and reference an object. It may be used to select resources that are resident in, or have been captured by, the presentation device. Such an identifier is referred to as an *object OID*. If the resource is mapped with an MDR reference, the FQN type X'DE' triplet must specify the same reference using the same FQN format. If the included object also references the resource with an internal identifier, this identifier must be specified on the IOB with a FQN type X'BE' triplet that immediately follows the FQN type X'DE' triplet. The paired triplets map the internal identifier to the external identifier. Resources that are used by data objects that may themselves be processed as resources are called *secondary resources*. See “Secondary Resource Objects” on page 14. Note that, if the included object contains an OEG, the FQN X'DE'/X'BE' mappings on the IOB override any FQN X'DE'/X'BE' mappings on an MDR in the OEG; the mappings on the OEG MDR are ignored when the object is included with an IOB.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur more than once if the IOB also specifies FQN type X'DE' triplets. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'BE'**—*Data Object Internal Resource Reference*. Specifies the identifier of a resource object that is used by the object being included. The identifier is used internally by the included object to reference the resource. The identifier must be specified using FQNFmt X'00', which, for this FQN type, indicates that the data type is defined by the specific data object that generates the internal resource reference and is undefined (UNDF) at the MO:DCA data stream level. When specified, this triplet must *immediately* follow the FQN type X'DE' triplet that specifies the external identifier of the resource, or a X'04' exception condition exists. Resources that are used by data objects that may themselves be processed as resources are called *secondary resources*. See "Secondary Resource Objects" on page 14. Note that, if the included object contains an OEG, the FQN X'DE'/X'BE' mappings on the IOB override any FQN X'DE'/X'BE' mappings on an MDR in the OEG; the mappings on the OEG MDR are ignored when the object is included with an IOB.
X'04'	Mapping Option	Optional. May occur once. If present, defines the mapping of the object data to the object area. If the referenced object specifies an object environment group (OEG), this triplet overrides the corresponding triplet on the the mapping structured field of the OEG. The specified mapping option must be valid for the object or a X'02' exception condition exists. If the referenced object is a page segment, this triplet overrides the corresponding triplet on the mapping structured field of the OEG in all objects that comprise the page segment. The specified mapping option must be valid for all objects in the page segment or a X'02' exception condition exists. See "Mapping Option Triplet X'04'" on page 332. If this triplet is omitted, the mapping option specified in the object's OEG is used. If the object does not specify the mapping option in an OEG, the architected default mapping for the object is used. Note that for objects referenced with ObjType = X'92', the architected default mapping is scale-to-fit.

Triplet	Type	Usage
X'4C'	Object Area Size	Optional. May occur once. If present, specifies the size of the object area (XoaSize, YoaSize) into which the object data is mapped. If the referenced object specifies an Object Environment Group (OEG), this triplet overrides the corresponding triplet on the the Object Area Descriptor (OBD) structured field of the OEG. If the referenced object is a page segment, this triplet overrides the corresponding triplet on the OBD structured field in all objects that comprise the page segment. If this triplet is omitted, the object area size specified in the object's OEG is used. If the object does not specify the object area size in an OEG, the architected default is to use the presentation space size of the including page or overlay. See "Object Area Size Triplet X'4C'" on page 365. **Note:** For static presentation objects, a presentation space size is required for a scale-to-fit or scale-to-fill mapping of the object presentation space to the object area. See "Object Type Identifiers" on page 535 for information on how the presentation space size is specified by various objects. If the object does not specify the presentation space size, the architected default is the presentation space size of the including page or overlay.
X'4E'	Color Specification	Optional. May occur once. Specifies the color that is to be used as the default color, or the initial color, for the object as specified in the object's data descriptor. This triplet overrides the default color specified in the data descriptor or sets the color if none is specified. This triplet only overrides default colors specified for the object presentation space; it does not affect colors assigned to the object's object area. The triplet must specify the color space as X'40'—Standard OCA color space, and the IOB must specify one of the following object types: **X'5F'** Page segment **X'BB'** Graphics (GOCA) **X'EB'** Bar code (BCOCA) **X'FB'** Image (IOCA) If these conditions are not met, the triplet is ignored. See "Color Specification Triplet X'4E'" on page 367.
X'70'	Presentation Space Reset Mixing	Optional. May occur once. This triplet may not appear on the Include Object structured field with a Presentation Space Mixing Rule (X'71') triplet. If present with BgMxFlag=1, specifies that both background and foreground of the referenced object data presentation space overpaint the area of the page or overlay presentation space that lies beneath it. If the referenced object specifies an Object Environment Group (OEG), this triplet overrides the corresponding triplet on the the OBD structured field of the OEG. If the referenced object is a page segment, this triplet overrides the corresponding triplet on the OBD structured field in all objects that comprise the page segment. If this triplet is omitted, the triplet specified on the OBD of the object's OEG is used. If the object does not specify this triplet on the OBD in an OEG, the architected default is to use the default mixing rule, that is, this triplet is ignored. For a definition of mixing rules see "Mixing Rules" on page 46. See "Presentation Space Reset Mixing Triplet X'70'" on page 390.

Triplet	Type	Usage
X'71'	Presentation Space Mixing Rules	Optional. May occur once. This triplet may not appear on the Include Object structured field with a Presentation Space Reset Mixing (X'70') triplet. If present, specifies the mixing rules for color mixing foreground and background object data on the portion of the page or overlay presentation space that lies beneath the object area. If the referenced object specifies an Object Environment Group (OEG), this triplet overrides the corresponding triplet on the the OBD structured field of the OEG. If the referenced object is a page segment, this triplet overrides the corresponding triplet on the OBD structured field in all objects that comprise the page segment. If this triplet is omitted, the triplet specified on the OBD of the object's OEG is used. If the object does not specify this triplet on the OBD in an OEG, the architected default is to use the default mixing rule, that is, this triplet is ignored. For a definition of mixing rules see "Mixing Rules" on page 46. See "Presentation Space Mixing Rules Triplet X'71'" on page 392. **Implementation Note:** The Presentation Space Mixing Rules (X'71') triplet is currently not used in AFP environments.

Architecture Note: When the IOB structured field is used in a page definition object in AFP line-data environments, an Extended Resource Local Identifier (X'22') triplet must be specified with ResType=X'30'—IOB Reference. The same triplet is used on a Descriptor in the Page Definition to reference the IOB and cause the specified object to be included.

IOB Exception Condition Summary

A X'01' exception condition exists when:

- The value specified for YoaOrent is not 90 degrees greater rotation than the value specified for XoaOrent
- An attempt is made to present data outside the presentation space of the containing coordinate system
- The mapping option is position and an attempt is made to present data outside the object area presentation space
- A Presentation Space Reset Mixing triplet and a Presentation Space Mixing Rules triplet are specified.

A X'02' exception condition exists when:

- The mapping option specified in a Mapping Option triplet is not valid for one or more of the referenced objects.

Image Picture Data (IPD)

The Image Picture Data structured field contains the data for an image data object.

IPD (X'D3EEFB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3EEFB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	IOCAdat		Up to 32759 bytes of IOCA defined data	O	X'00'

IPD Semantics

IOCAdat Contains the IOCA defined data. See the MO:DCA environment appendix in the *Image Object Content Architecture Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Include Page (IPG)

The Include Page structured field references a page that is to be included in the document. The Include Page structured field may occur in document state, page-group state, or page state. In all three cases the referenced page is positioned on the media using the (X_m, Y_m) offsets specified in the PGP structured field in the active medium map. The referenced page must not contain another Include Page structured field.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a printfile that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

IPG (X'D3AFAF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AFAF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	PgName		Name of the page	M	X'06'
8–15				Reserved; must be zero	M	X'06'
16	BITS	IPgFlgs		Specify control information for the included page. See “IPG Semantics” for bit definitions.	M	X'06'
17–*n*		Triplets		See “IPG Semantics” for triplet applicability.	M	X'14'

IPG Semantics

PgName Is the name of the page being referenced. The page name is qualified, using the Fully Qualified Name (X'02') type X'83' triplet, with the name of the document that contains the page.

IPgFlgs Specify control information for the included page.

Bit **Description**

0 Format of included page, must be set to B'1'.

B'0' Reserved

B'1' The referenced page is carried in a document in an external (inline) resource group. Before this page can be included with the IPG, it must be processed with all required resources and saved in the presentation device. The processing includes the application of all text suppressions specified in the medium map that is active when the page is saved.

1–7 Reserved; all bits must be B'0'.

Triplets Appear in the Include Page structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'83'**—*Begin Document Name*. Specifies the name of the document that contains the referenced page.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Include Page structured field name and is used as the name of the page.
X'5A'	Object Offset	Optional. May occur once, with *ObjTpe*=X'AF' to specify that pages are the objects to be counted for the offset. Specifies how many pages in the referenced document precede the page to be included. The page offset is measured from the beginning of the referenced document, so that the first page has offset 0, the second page has offset 1, and the nth page has offset (*n*–1). When this triplet is specified, the page name, as specified by the *PgName* parameter or by the Fully Qualified Name type X'01' triplet, is ignored. See "Object Offset Triplet X'5A'" on page 378.

Notes:

1. Care must be taken when activating text suppressions on pages to be saved. The document that contains the pages to be saved must be processed with the same form map as the document that references the saved pages. However, unless the two documents have the identical structure with respect to pages, Invoke Medium Map (IMM) structured fields, and internal (inline) medium maps, the medium map that is active when the page is saved may specify different text suppressions than the medium map that is active when the page is included, which may yield unexpected results.
2. If the medium map specifies multiple copy subgroups with different text suppression activations, the presentation device must process and save a copy of the page for each set of text suppressions. When an IPG is processed for multiple copy subgroups, the presentation device uses the copy of the saved page whose text suppressions match those required by the current medium map.
3. The following rules apply to overlays when a page is processed and saved by the presentation device:
   - Page overlays are processed and saved with the page.
   - PMC overlays are not processed and saved with the page. They are applied to the page when it is included with an IPG as specified by the medium map that is active during page presentation.
   - Medium overlays are not processed and saved with the page. They are applied to the medium as specified by the medium map that is active during page presentation.

4. Overlays that are included on the saved page may overflow the saved page presentation space. Such overflow areas need to be saved with the page since they only cause an exception at presentation time if they contain data that overflows the medium presentation space. If an attempt is made to present overlay data that overflows the medium presentation space, that portion of the data is not presented and a X'01' exception condition exists.
5. The size of the page may exceed the size of the medium presentation space in either the X_m or Y_m direction. If an attempt is made to present data in the portion of the page that overflows the medium presentation space, that portion of the data is not presented and a X'01' exception condition exists.
6. A page that is included with an IPG may be indexed as follows:
   - If the IPG occurs in document state or in page-group state, the included page may be indexed using an offset to the location of the IPG in the document.
   - If the IPG occurs in page state, the included page becomes a part of the containing page, therefore only the containing page may be indexed using an offset to its location in the document.

Include Page Overlay (IPO)

The Include Page Overlay structured field references an overlay resource definition that is to be positioned on the page. A page overlay can be referenced at any time during the page state, but not during an object state. The overlay contains its own active environment group definition.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a printfile that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

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	Exc
0–7	CHAR	OvlyName		Name of the overlay resource	M	X'06'
8–10	SBIN	XolOset	-32768–32767	X-axis origin for the page overlay	M	X'06'
			X'FFFFFF'	Retired value		
11–13	SBIN	YolOset	-32768–32767	Y-axis origin for the page overlay	M	X'06'
			X'FFFFFF'	Retired value		
14–15	CODE	OvlyOrent	X'0000', X'2D00', X'5A00', X'8700'	The overlay's X-axis rotation from the X axis of the including page coordinate system: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	O	X'02'
16–*n*		Triplets		See "IPO Semantics" for triplet applicability.	O	X'10'

IPO Semantics

OvlyName Is the name of the overlay resource definition being referenced.

XolOset Specifies the offset along the X-axis of the including page coordinate system, X_{pg}, to the origin of the X axis for the page overlay coordinate system, X_{ol}. The value X'FFFFFF' is retired, therefore the offset value (-1) is not included in the allowed range. See the architecture note following the Triplets section. The value for this parameter is expressed in terms of the number of page coordinate system X-axis measurement units.

YolOset Specifies the offset along the Y axis of the including page coordinate system, Y_{pg}, to the origin of the Y axis for the page overlay coordinate system, Y_{ol}. The value X'FFFFFF' is retired, therefore the offset value (-1) is not included in the allowed range. See the architecture note following the Triplets section. The value for this parameter is expressed in terms of the number of page coordinate system Y-axis measurement units.

OvlyOrent Specifies the amount of rotation of the page overlay's X axis, X_{ol}, about the page overlay origin relative to the X axis, X_{pg}, of the including page coordinate system. 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: If the rotation is such that the sum of the page overlay origin offset and the page overlay extent exceeds the size of the including presentation space in either the X or Y direction, all of the object area will not fit on the including presentation space. The including presentation space in this case is the medium presentation space. If an attempt is made to actually present data in the portion of the page overlay that falls outside the including presentation space, that portion of the data is not presented, and a X'01' exception condition exists.

Application Note: 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.

Triplets Appear in the Include Page Overlay structured field as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'02'	Fully Qualified Name	Optional. May occur once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'01'**—*Replace First GID Name*. This GID overrides the Include Overlay structured field name and is used as the name of the overlay.
X'46'	Page Overlay Conditional Processing	Optional. May occur more than once. See "Page Overlay Conditional Processing Triplet X'46'" on page 361.
X'47'	Resource Usage Attribute	Optional. May occur once. See "Resource Usage Attribute Triplet X'47'" on page 363.

Note: If a triplet is included on this structured field, the optional positional parameter becomes mandatory.

The current environment of the page that included the overlay is restored when the Include Page Overlay has been completed.

Architecture Note: In AFP line data environments, the value X'FFFFFF' is supported for the XolOset and YolOset parameters to indicate that the X_p or Y_p position, respectively, defined by the current Line Descriptor (LND) in the page definition is to be used as the origin for the overlay. This value was also valid in pre-1992 AFP data streams to specify the current text print position and is supported by some print servers for migration of such data streams. However, this value is not valid in MO:DCA data streams and should not be generated by MO:DCA applications. To record support for this value by some AFP print servers and to limit any further use, this value is retired; see "Retired Parameters" on page 515.

IPO Exception Condition Summary

- A X'01' exception condition exists when:
  - Multiple Resource Usage Attribute (X'47') triplets appear
  - An attempt is made to present data outside the medium presentation space. See the note under *OvlyOrent* for details.

Include Page Segment (IPS)

The Include Page Segment structured field references a page segment resource object that is to be presented on the page or overlay presentation space. The IPS specifies a reference point on the including page or overlay coordinate system that may be used to position objects contained in the page segment. A page segment can be referenced at any time during page or overlay state, but not during an object state. The page segment inherits the active environment group definition of the including page or overlay.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a printfile that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

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	Exc
0–7	CHAR	PsegName		Name of the page segment resource	M	X'06'
8–10	SBIN	XpsOset	-32768–32767	X axis origin for positioning objects	M	X'06'
			X'FFFFFF'	Retired value		
11–13	SBIN	YpsOset	-32768–32767	Y-axis origin for positioning objects	M	X'06'
			X'FFFFFF'	Retired value		
14–*n*		Triplets		See “IPS Semantics” for triplet applicability.	O	X'10'

IPS Semantics

PsegName Is the name of the page segment resource object being referenced.

XpsOset Specifies the offset along the X axis of the including page coordinate system, X_{pg}, or the including overlay coordinate system, X_{ol}, to the reference point that may be used to position objects in the page segment. The value X'FFFFFF' is retired, therefore the offset value (-1) is not included in the allowed range. See the architecture note following the Triplets section. The value for this parameter is expressed in terms of the number of page or overlay coordinate system X-axis measurement units.

YpsOset Specifies the offset along the Y axis of the including page coordinate system, Y_{pg}, or the including overlay coordinate system, Y_{ol}, to the reference point that may be used to position objects in the page segment. The value X'FFFFFF' is retired, therefore the offset value (-1) is not included in the allowed range. See the architecture note following the Triplets section. The value for this parameter is expressed in terms of the number of page or overlay coordinate system Y-axis measurement units.

Triplets Appear as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.

Architecture Note: In AFP line data environments, the value X'FFFFFF' is supported for the XpsOset and YpsOset parameters to indicate that the X_p or Y_p position, respectively, defined by the current Line Descriptor (LND) in the Page Definition is to be used as the “origin” for the page segment. This value was also valid in pre-1992 AFP data streams to specify the current text print position and is supported by some print servers for migration of such data streams. However this value is not valid in MO:DCA data streams and should not be generated by MO:DCA applications. To record support for this value by some AFP print servers and to limit any further use, this value is retired, see “Retired Parameters” on page 515.

Application Note: For purposes of Print Services Facility resource management, the Object Environment Groups (OEGs) for all objects in the page segment must not contain MCF or MDR structured fields when the page segment is referenced with an IPS structured field.

IPS Exception Condition Summary

- A X’01’ exception condition exists when an attempt is made to present data outside the medium presentation space.

Link Logical Element (LLE)

A Link Logical Element structured field specifies the linkage from a source document component to a target document component. The LLE identifies the source and target and indicates the purpose of the linkage by specifying a link type. The link source and link target may be in the same document component or in different document components, and they need not be of the same document component type. The linkage may involve a complete document component, or it may be restricted to a rectangular area on the presentation space associated with the document component. The Link Logical Element structured field can be embedded in the document that contains the link source, in the document that contains the link target, in the document index for either document, or in any combination of these structures. Link Logical Element parameters do not provide any presentation specifications.

LLE (X'D3B490') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B490'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	LnkType	X'01'–X'03'	Link type: **X'01'** Navigation link **X'02'** Annotation link **X'03'** Append link	M	X'06'
1				Reserved; must be zero	M	X'06'
Two or three repeating groups in the following format:						
0–1	UBIN	RGLength	3–(*n*+1)	Total length of this repeating group	M	X'06'
2	CODE	RGFunct	X'01'–X'03'	Repeating group function: **X'01'** Link attribute specification **X'02'** Link source specification **X'03'** Link target specification	M	X'06'
3–*n*		Triplets		See “LLE Semantics” for triplet applicability.	O	X'10'

LLE Semantics

LnkType Specifies the purpose of the link.

Value	Description
X'01'	Navigation link. Specifies the linkage from a source document component to a contextually-related target document component. Navigation links may be used to support applications such as hypertext and hypermedia.

X'02' Annotation link. Specifies the linkage from a source document component to a target document component that contains an annotation for the source.

X'03' Append link. Specifies the linkage from the end of a source document component to a target document component that contains an append to the source.

All others Reserved

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

RGFunct Identifies the function of the repeating group:

Value	**Description**
X'01'	The repeating group specifies general attributes of the link.
X'02'	The repeating group specifies the source of the link.
X'03'	The repeating group specifies the target of the link.

Every Link Logical Element structured field must contain one repeating group that specifies the source of the link, and one repeating group that specifies the target of the link. Every Link Logical Element structured field may optionally contain one additional repeating group that specifies attributes of the link.

The optional attribute repeating group can be used to specify attributes and data that apply to the whole link, such as the name of the Link Logical Element structured field, the code page and character set used to encode character data in the Link Logical Element structured field, and parameter data to be associated with the link.

The source and target repeating groups specify the document components that are the source and target of the link and may further restrict the source and target to rectangular areas on the corresponding document component presentation spaces. The source and target repeating groups may qualify the name of a document component with the names of the document components that are higher in the document hierarchy. For example, if the target of the link is a page, the target repeating group may specify the name of the page, the name of the page group that contains the page, and the name of the document that contains the page group. If the names of the document components that are higher in the document hierarchy are not explicitly specified in the LLE repeating groups, they are inherited from the document components that contain the Link Logical Element structured field. For example, if a source repeating group only specifies an area, then the Link Logical Element structured field must be located within a page or overlay definition, and the name of the page or overlay, as well as the name of the document, are inherited by the source repeating group.

The inheritance of names is bypassed if the repeating group indicates that the source or target is located in the MO:DCA resource hierarchy. In that case, the source or target is located using the resource search order defined in "Resource Search Order" on page 29. The inheritance of names is also bypassed if the repeating group references the source or target with FQNFmt X'20' - URL. In that case, the source or target is a resource located on the Internet.

In general, source and target repeating groups may specify multiple document component names, however within each repeating group the identified document components must all be part of the same document hierarchy, and the actual source or target of the link is determined by the lowest specified member of that document component hierarchy.

If any positional processing is associated with the link source or link target on a page, such as the positioning of a cursor, processing starts at the location in the source or target that is closest to the page origin. For example, if the link target is specified to be an area on a page, positional processing starts at the corner of the area that is closest to the page origin. If the link target is specified to be a group of areas on a page, positional processing starts at the area corner that is closest to the page origin. If the link target is a page, positional processing starts at the page origin. Closest in this case is defined to be the minimum geometric distance. A given point (X,Y) on the page has a distance to the page origin defined by $\sqrt{(X^2+Y^2)}$, so that for a set of points, the point closest to the page origin is defined by the minimum $\sqrt{(X^2+Y^2)}$.

Table 17 shows which document components may be specified as link sources in a link source repeating group or as link targets in a link target repeating group.

Table 17. Link Sources and Link Targets

Component	Link Source	Link Target
Document	Yes	Yes
Page group	Yes	Yes
Page	Yes	Yes
Overlay	Yes	Yes
Process element (TLE)	Yes	Yes
Rectangular area	Yes	Yes
Other object data	Yes	Yes

Triplets Appear in Link Logical Element structured field repeating groups as shown in Figure 54 on page 202.

Link Attribute Repeating Group
- Coded Graphic Character Set Global Identifier (X′01′) triplet
- Fully Qualified Name (X′02′) triplet, type X'0C'—Process Element (LLE) Name
- Parameter Value (X′82′) triplet

Link Source Repeating Group
- Coded Graphic Character Set Global Identifier (X′01′) triplet
- Fully Qualified Name (X′02′) triplet, type X'09'—MO:DCA Resource Hierarchy Reference
- Fully Qualified Name (X′02′) triplet, type X'0A'—Begin Resource Group Reference
- Fully Qualified Name (X′02′) triplet, type X'0C'—Process Element (TLE) Name
- Fully Qualified Name (X′02′) triplet, type X'0D'—Begin Page Group Reference
- Fully Qualified Name (X′02′) triplet, type X'83'—Begin Document Reference
- Fully Qualified Name (X′02′) triplet, type X'87'—Begin Page Reference
- Fully Qualified Name (X′02′) triplet, type X'B0'—Begin Overlay Reference
- Fully Qualified Name (X′02′) triplet, type X'CE'—Other Object Data Reference
- Object Classification (X′10′) triplet
- Measurement Units (X′4B′) triplet
- Area Definition (X′4D′) triplet

Link Target Repeating Group
- Coded Graphic Character Set Global Identifier (X′01′) triplet
- Fully Qualified Name (X′02′) triplet, type X'09'—MO:DCA Resource Hierarchy Reference
- Fully Qualified Name (X′02′) triplet, type X'0A'—Begin Resource Group Reference
- Fully Qualified Name (X′02′) triplet, type X'0C'—Process Element (TLE) Name
- Fully Qualified Name (X′02′) triplet, type X'0D'—Begin Page Group Reference
- Fully Qualified Name (X′02′) triplet, type X'83'—Begin Document Reference
- Fully Qualified Name (X′02′) triplet, type X'87'—Begin Page Reference
- Fully Qualified Name (X′02′) triplet, type X'B0'—Begin Overlay Reference
- Fully Qualified Name (X′02′) triplet, type X'CE'—Other Object Data Reference
- Object Classification (X′10′) triplet
- Measurement Units (X′4B′) triplet
- Area Definition (X′4D′) triplet

Figure 54. Triplets in Link Attribute, Source, and Target Repeating Groups

Note that by specifying FQNFmt = X'20' - URL for the FQN format of the target name, the LLE can be used to link to resources on the Internet using a Uniform Resource Locator (URL).

Details on triplet semantics and on rules for including each triplet on the repeating groups are as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur multiple times in each repeating group. If in a link attribute repeating group, specifies the code page and character set for all character data in all three LLE repeating groups, unless overridden by a Coded Graphic Character Set Global Identifier triplet in a source or target repeating group, in which case the latter triplet specifies the code page and character set for that repeating group. If in a link source or link target repeating group, specifies the code page and character set for that repeating group. By specifying this triplet multiple times in a link source or link target repeating group, you can specify a unique code page and character set for the character data in every triplet on that repeating group.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once in a link source repeating group and once in a link target repeating group. The Fully Qualified Name type that may appear is **X'09'**—*MO:DCA Resource Hierarchy Reference*. If in a link source repeating group, specifies that the link source object is located in the MO:DCA resource hierarchy. If in a link target repeating group, specifies that the link target object is located in the MO:DCA resource hierarchy. See "Resource Search Order" on page 29.
X'02'	Fully Qualified Name	Optional. May occur once in a link source repeating group and once in a link target repeating group. The Fully Qualified Name type that may appear is **X'0A'**—*Begin Resource Group Reference*. If in a link source repeating group, specifies a resource group that contains the link source. If in a link target repeating group, specifies a resource group that contains the link target.
X'02'	Fully Qualified Name	Optional. May occur once in each repeating group. The Fully Qualified Name type that may appear is **X'0C'**—*Process Element Name*. If in a link attribute repeating group, specifies the name of the Link Logical Element. If in a link source repeating group, specifies the name of a Tag Logical Element that is the link source. If in a link target repeating group, specifies the name of a Tag Logical Element that is the link target.
X'02'	Fully Qualified Name	Optional. May occur once in a link source repeating group and once in a link target repeating group. The Fully Qualified Name type that may appear is **X'0D'**—*Begin Page Group Reference*. If in a link source repeating group, specifies a page group that is the link source or that contains the link source. If in a link target repeating group, specifies a page group that is the link target or that contains the link target.
X'02'	Fully Qualified Name	Optional. May occur once in a link source repeating group and once in a link target repeating group. The Fully Qualified Name type that may appear is **X'83'**—*Begin Document Reference*. If in a link source repeating group, specifies a document that is the link source or that contains the link source. If in a link target repeating group, specifies a document that is the link target or that contains the link target.
X'02'	Fully Qualified Name	Optional. May occur once in a link source repeating group and once in a link target repeating group. The Fully Qualified Name type that may appear is **X'87'**—*Begin Page Reference*. If in a link source repeating group, specifies a page that is the link source or that contains the link source. If in a link target repeating group, specifies a page that is the link target or that contains the link target.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once in a link source repeating group and once in a link target repeating group. The Fully Qualified Name type that may appear is **X'B0'**—*Begin Overlay Reference*. If in a link source repeating group, specifies an overlay that is the link source or that contains the link source. If in a link target repeating group, specifies an overlay that is the link target or that contains the link target.
X'02'	Fully Qualified Name	Optional. May occur once in a link source repeating group and once in a link target repeating group. The Fully Qualified Name type that may appear is **X'CE'**—*Other Object Data Reference*. If in a link source repeating group, specifies other object data that is the link source or that contains the area that is the link source. If in a link target repeating group, specifies other object data that is the link target or that contains the area that is the link target. The object data being linked may or may not be defined by an IBM presentation architecture. The object data is characterized and identified by a mandatory Object Classification (X'10') triplet, which also specifies whether the object data is carried in a MO:DCA object container, whether it is unwrapped object data, or whether the container structure of the object data is unknown. Note that if FQNFmt X'20' (URL) is used to specify a link source or target, the object type is defined by the URL itself and the Object Classification (X'10') triplet becomes optional.
X'10'	Object Classification	Mandatory if the Fully Qualified Name type X'CE', Other Object Data Reference, appears in a link source or a link target repeating group, in which case it must occur once in that repeating group. Otherwise this triplet is not allowed in a repeating group. Specifies information used to characterize and identify other object data. Note however that if FQN type X'CE' with FQNFmt X'20' (URL) is used to specify the link source or target, the object type is defined by the URL itself and the Object Classification (X'10') triplet becomes optional. See "Object Classification Triplet X'10'" on page 335.
X'4B'	Measurement Units	Optional if one or more Area Definition (X'4D') triplets are present in a link source or link target repeating group, in which case it may occur once in that repeating group. Specifies the units of measure to be used for positioning areas and for determining their size. If this triplet is omitted when an Area Definition triplet is present, the units of measure are specified by the document component on which the area is defined. See "Measurement Units Triplet X'4B'" on page 364.

Triplet	Type	Usage
X'4D'	Area Definition	Optional. May occur multiple times in a link source repeating group and multiple times in a link target repeating group. Defines a rectangular area on the presentation space of the lowest document component in the document hierarchy that is specified by the repeating group or that is inherited by the repeating group. If the repeating group does not explicitly specify an object, then the object specification is inherited from the document hierarchy. For example, if the LLE is located in a page, and if the repeating group does not specify any document component at the page level or at a lower level in the document hierarchy, then the area is defined on the presentation space for the page that contains the LLE. The units of measure for resolving the offset and size of the area are specified by a Measurement Units triplet, if present, or by the document component on which the presentation space is defined if the triplet is not present. When this triplet occurs multiple times on a link source repeating group, the logical union of the areas defines the link source. When this triplet occurs multiple times on a link target repeating group, the logical union of the areas defines the link target. See "Area Definition Triplet X'4D'" on page 366.
X'82'	Parameter Value	Optional. May occur multiple times in a link attribute repeating group. Used to pass parameter values to the link target. See "Parameter Value Triplet X'82'" on page 404.

LLE Exception Condition Summary

- A X'04' exception condition exists when the Area Definition triplet is present in a repeating group but the Measurement Units triplet is absent and the lowest identified document component in the document hierarchy does not define units of measure.

Map Bar Code Object (MBC)

The Map Bar Code Object structured field specifies how a bar code data object is to be mapped into its object area.

MBC (X'D3ABEB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABEB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One repeating group in the following format:						
0–1	UBIN	RGLength	5	Total length of this repeating group	M	X'06'
2–4		Triplets		Mapping Option triplet	M	X'14'

MBC Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Bar Code Object structured field as follows:

Triplet	Type	Usage
X'04'	Mapping Option	Mandatory. Must occur once in each repeating group. See "Mapping Option Triplet X'04'" on page 332. The valid mapping options for the MBC structured field are: **Value** **Description** **X'00'** Position **All others** Reserved

Note: If this structured field is not present in the data stream, the architected default is *position*.

MBC Exception Condition Summary

- A X'02' exception condition exists when a Mapping Option (X'04') triplet value other than X'00' is specified.
- A X'01' exception condition exists when the Map Bar Code Object structured field contains more than one repeating group.

Map Color Attribute Table (MCA)

The Map Color Attribute Table structured field maps a unique Resource Local ID to the name of a Begin Color Attribute Table structured field. A local ID may be embedded one or more times within an object’s data.

Note: The MCA structured field is used only in MO:DCA-L data streams.

MCA (X'D3AB77') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AB77'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One to 254 repeating groups in the following format:						
0–1	UBIN	RGLength	7–260	Total length of this repeating group	M	X'06'
2–*n*		Triplets		See “MCA Semantics” for triplet applicability.	M	X'14'

MCA Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear within each repeating group as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once in each repeating group. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'84'**—*Begin Resource Object Reference*, which must match the name on a Begin Color Attribute Table structured field or a X’01’ exception condition exists.
X'24'	Resource Local Identifier	Mandatory for image, not present for graphics. For image, this triplet must occur once in each repeating group. See “Resource Local Identifier Triplet X'24'” on page 350. The only resource type that may appear is **X'07'**—*Color Attribute Table*.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.

Within the same Map Color Attribute Table structured field, it is not permissible to map the same Resource Local ID to more than one color attribute table or a X’01’ exception condition exists. However, two or more repeating groups within the same Map Color Attribute Table structured field may be used to map different LIDs to the same color attribute table.

Note: If this structured field is not present in the data stream, the architected default LID is X'00'.

MCA Exception Condition Summary

- A X’02’ exception condition exists when:
  - A Fully Qualified Name (X’02’) triplet other than a type X'84' (Begin Resource Object Reference) appears within any repeating group
  - A Resource Local Identifier (X’24’) triplet type other than X'07' appears within any repeating group
- A X’01’ exception condition exists when:
  - A Begin Color Attribute Table structured field with the same name as that specified on the type X'84' (Begin Resource Object Reference) Fully Qualified Name triplet could not be located
  - Multiple type X'84' (Begin Resource Object Reference) Fully Qualified Name triplets appear within the same repeating group
  - Multiple type X'07' Resource Local Identifier triplets appear within the same repeating group
  - The same LID is mapped to more than one color attribute table within the same structured field

Medium Copy Count (MCC)

The Medium Copy Count structured field specifies the number of copies of each medium, or sheet, to be presented, and the modifications that apply to each copy. This specification is called a *copy group*. The MCC contains repeating groups that specify *copy subgroups*, such that each copy subgroup may be specified independently of any other copy subgroup. For each copy subgroup, the number of copies, as well as the modifications to be applied to each copy, is specified by the repeating group. If the modifications for a copy subgroup specify duplexing, that copy subgroup and all successive copy subgroups are paired such that the first copy subgroup in the pair specifies the copy count as well as the modifications to be applied to the front side of each copy, and the second copy subgroup in the pair specifies the same copy count as well as an independent set of modifications to be applied to the back side of each copy. The pairing of copy subgroups continues as long as duplexing is specified.

MCC (X'D3A288') Syntax

Structured Field Introducer				Structured Field Data
SF Length (2B)	ID = **X'D3A288'**	Flags (1B)	Reserved; X'0000'	

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
One to 128 repeating groups in the following format:						
0–1	UBIN	Startnum	1–32386	Starting copy number	M	X'06'
2–3	UBIN	Stopnum	1–32640	Ending copy number	M	X'06'
4				Reserved; must be zero	M	X'06'
5	CODE	MMCid	0–127	Medium Modification Control identifier	M	X'06'

MCC Semantics

Startnum The number of the first copy of the sheet for this copy subgroup. For the first copy subgroup this value must be 1. For other copy subgroups, this value must be one greater than the ending copy number of the preceding copy subgroup, or a X'01' exception condition exists.

Stopnum The number of the last copy of the sheet for this copy subgroup. This value must be greater than or equal to the value specified by *Startnum*, or a X'01' exception condition exists. The number of copies requested by the copy subgroup, called the copy count, which is defined by (*Stopnum*–*Startnum*) + 1, must be less than or equal to 255, or a X'02' exception condition exists. The total number of copies for the copy group, which is the sum of the copy counts for all copy subgroups, is equal to the value of *Stopnum* in the last copy subgroup.

MMCid Identifies a Medium Modification Control (MMC) structured field that specifies the modifications to be applied to all copies for the copy subgroup. A value of 0 selects an environment-specific set of default modifications.

MCC Exception Condition Summary

- A X′02′ exception condition exists when the copy count in a copy subgroup is greater than 255.
- A X′01′ exception condition exists when:
  - For all copy subgroups other than the first, the starting copy number in a copy subgroup is not 1 greater than the ending copy number in the preceding copy subgroup.
  - The ending copy number in a copy subgroup is not equal to or greater than the starting copy number in the same copy subgroup.

Map Container Data (MCD)

The Map Container Data structured field specifies how a presentation data object that is carried within an object container is mapped into its object area.

MCD (X'D3AB92') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AB92'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One repeating group in the following format:						
0–1	UBIN	RGLength	5	Total length of this repeating group	M	X'06'
2–4		Triplets		Mapping Option triplet	M	X'14'

MCD Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Container Data structured field as follows:

Triplet	Type	Usage
X'04'	Mapping Option	Mandatory. Must occur once. See "Mapping Option Triplet X'04'" on page 332. The valid mapping options for the MCD structured field are: **Value** **Description** **X'00'** Position **X'10'** Position and trim **X'20'** Scale to fit **X'30'** Center and trim **X'60'** Scale to fill **All others** Reserved

Notes:

1. If this structured field is not present in the data stream, the architected default for the mapping option is *scale to fit*.
2. A presentation space size is required for a scale-to-fit or scale-to-fill mapping of the object presentation space to the object area. See "Object Type Identifiers" on page 535 for information on how the presentation space size is specified by various objects. If the presentation space size is not specified by the object, the architected default is the presentation space size of the including page or overlay.
3. This structured field is not applicable to non-presentation objects. It may be ignored if it appears in the object container for such objects.

MCD Exception Condition Summary

- A X'01' exception condition exists when the Map Container Data structured field contains more than one repeating group.

Map Coded Font (MCF) Format 2

The Map Coded Font structured field maps a unique coded font resource local ID, which may be embedded one or more times within an object's data and descriptor, to the identifier of a coded font resource object. This identifier may be specified in one of the following formats:

- A coded font Global Resource Identifier (GRID)
- A coded font name
- A combination of code page name and font character set name

Additionally, the Map Coded Font structured field specifies a set of resource attributes for the coded font.

MCF (X'D3AB8A') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AB8A'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One to 254 repeating groups in the following format:						
0–1	UBIN	RGLength	7–(*n*+1)	Total length of this repeating group	M	X'06'
2–*n*		Triplets		See "MCF Semantics" for triplet applicability.	M	X'14'

MCF Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear within each repeating group as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. A Fully Qualified Name (X'02') triplet of any permitted type may appear only once in a repeating group. The Fully Qualified Name types permitted in a repeating group are: • **X'07'**—*Font Family Name* • **X'08'**—*Font Typeface Name* • **X'84'**—*Begin Resource Object Reference* • **X'85'**—*Code Page Name Reference* • **X'86'**—*Font Character Set Name Reference* • **X'8E'**—*Coded Font Name Reference* At a minimum, each repeating group must contain one of these triplets or triplet groups: • A single Fully Qualified Name type X'84' (Coded Font Reference) triplet • A Fully Qualified Name type X'85' (Code Page Name Reference) and a Fully Qualified Name type X'86' (Font Character Set Name Reference) triplet • A single Fully Qualified Name type X'8E' (Coded Font Name Reference) triplet See "Fully Qualified Name Triplet X'02'" on page 320. The type X'84' (Coded Font Reference) is not permitted in the same repeating group with the type X'8E' (Coded Font Name Reference), and neither is permitted in the same repeating group with a type X'85' (Code Page Name Reference) or a type X'86' (Font Character Set Name Reference). When the type X'84' (Coded Font Reference) identifies a font encoded using the EBCDIC Presentation double-byte encoding scheme (encoding scheme ID X'62*nn*') or the EBCDIC Presentation single-byte encoding scheme (encoding scheme ID X'61*nn*'), it is not permitted in the same repeating group with a Resource Section Number (X'25') triplet having a value other than X'00'. If a Fully Qualified Name type X'84' triplet specifies a font width in the global resource identifier (GRID), and if a vertical font size is not specified by a Font Descriptor (X'1F') triplet, this parameter may be used to generate the vertical font size, which is used to scale outline technology fonts to the desired point size. **Architecture Note:** If a coded font reference consists of only the GRID and does not contain a Font Descriptor triplet, it is assumed to have been generated by an application that was using integer point sizes. When the font width in such a font reference is used to calculate a specified vertical font size for scaling outline technology fonts, the calculated vertical font size is rounded to the nearest positive, non-zero, integer point size.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.

Triplet	Type	Usage
X'1F'	Font Descriptor Specification	Optional. May occur once in each repeating group. The specified vertical font size in this triplet may be used to scale an outline technology font to the desired point size and overrides any vertical font size that is calculated from a specified horizontal font size. If the vertical font size is not specified, the font width in the GRID may be used to calculate the specified vertical font size for scaling outline technology fonts. If a font width was not specified in the GRID, the specified horizontal font size in this triplet may be used to calculate the specified vertical font size for scaling outline technology fonts. If the specified vertical font size conflicts with the nominal vertical font size in the font object, the specified vertical font size overrides. A coded font reference may not always specify a vertical font size, such as when the reference does not include a GRID or a Font Descriptor triplet. In that case, the font object must provide the vertical font size for scaling an outline technology font. See "Font Descriptor Specification Triplet X'1F'" on page 341.
X'20'	Font Coded Graphic Character Set Global Identifier	Optional. May occur once in each repeating group. See "Font Coded Graphic Character Set Global Identifier Triplet X'20'" on page 345.
X'24'	Resource Local Identifier	Optional. May occur once in each repeating group. See "Resource Local Identifier Triplet X'24'" on page 350. The only resource type that may appear is **X'05'**—*Coded Font*. **Note:** If a resource LID is not specified in a Map Coded Font structured field, the architected default LID is X'00' and the architected default LID type is X'00'. **Application Note:** For purposes of PSF resource management, each MCF that maps a font in a data object OEG must have a corresponding MCF mapping the same font in the AEG for that page or overlay. The ID used in the AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG.
X'25'	Resource Section Number	Optional. May occur once in each repeating group. See "Resource Section Number Triplet X'25'" on page 351.
X'26'	Character Rotation	Optional. May occur once in each repeating group. See "Character Rotation Triplet X'26'" on page 352.

Triplet	Type	Usage
X'50'	Encoding Scheme ID	Optional. May occur once in each repeating group. See "Encoding Scheme ID Triplet X'50'" on page 371. The ESidCP parameter specifies the encoding scheme associated with the code page in the referenced font. Additionally, the ESidUD parameter may be specified to indicate the encoding scheme for the user data to be rendered with the referenced font. When the two encoding schemes do not match, the presentation system may need to transform the user data to match the encoding in the code page. Not all presentation systems support such transforms. To see which transforms are supported, consult your product documentation. See Table 18 on page 218 for the combinations of ESidCP and ESidUD that are valid for the MCF. **Note:** If this triplet is omitted, the architected default for the encoding scheme is EBCDIC Presentation for single-byte fonts and EBCDIC Presentation for double-byte fonts. The architected default for the user data encoding scheme is the code page encoding scheme; that is, it is assumed that the encoding for the user data matches the encoding in the font used to render the user data.
X'5D'	Font Horizontal Scale Factor	Optional. May occur once in each repeating group. Carries information that allows an outline technology font to be scaled anamorphically by specifying a horizontal scale factor. This horizontal scale factor is applied to the horizontal font dimension. If the font horizontal scale factor is the same as the specified vertical font size, the font scaling is uniform. If the font horizontal scale factor is not the same as the specified vertical font size, the font scaling is anamorphic; and the graphic characters are stretched or compressed in the horizontal direction relative to the vertical direction by the ratio of font horizontal scale factor divided by the specified vertical font size. If this triplet is omitted, the font horizontal scale factor defaults to the specified vertical font size and the scaling is uniform. A coded font reference may not always specify a vertical font size, such as when the reference does not include a GRID or a Font Descriptor triplet. In that case, if a Horizontal Scale Factor triplet is specified on the coded font reference, it is ignored. The vertical font size in the font object is then used to scale an outline technology font in the vertical direction, and the horizontal scale factor in the font object, if supplied, is used for anamorphic scaling. If a horizontal scale factor is not supplied in the font object, scaling is uniform. See "Font Horizontal Scale Factor Triplet X'5D'" on page 380.
X'84'	Font Resolution and Metric Technology	Optional. May occur once in each repeating group. Specifies metric information for a raster coded font. See page 406. Note that the presence of this triplet indicates that the MCF references a raster-technology coded font.

Application Note: In AFP environments, the following retired triplet is used on this structured field:

- Text Orientation (X'1D') triplet. See "Text Orientation Triplet X'1D'" on page 505.

MCF Usage Information

Only a Map Coded Font structured field can map a resource local ID to a pair of code page/font character set names.

The names of coded fonts, code pages, and font character sets can be specified in several ways. See the appropriate interchange set definition, “MO:DCA Presentation Interchange Set 1” on page 429, “MO:DCA Presentation Interchange Set 2” on page 444, or “MO:DCA Resource Interchange Set” on page 463 for the correct syntax of these names.

Multiple Resource Local Identifier (X’24’) triplet values (LIDs) may be mapped to the same font, but the same Resource Local Identifier (X’24’) triplet value may not be mapped to more than one font within the same structured field.

Double-byte Font References

The same Resource Local Identifier (X’24’) triplet value may be mapped to different subsections of the same double-byte font. When this is done, the following rules apply:

- All repeating groups associated with the double-byte font must be contiguous.
- Each repeating group must either default the LID value or contain a Resource Local Identifier (X’24’) triplet with the same value.
- Each repeating group must contain a Fully Qualified Name type X’85’ (Code Page Name Reference) and Fully Qualified Name type X’86’ (Font Character Set Name Reference).
- When the font uses the EBCDIC Presentation double-byte encoding scheme (encoding scheme ID X’62*nn*’), each repeating group must contain a Resource Section Number (X'25') triplet that specifies a valid double-byte section number in the range X'41' through X'FE'.
- Each Resource Section Number (X’25’) triplet value specified must be unique within the entire set of repeating groups associated with the double-byte font.
- A Character Rotation (X’26’) triplet may be specified in *any* of the repeating groups associated with the font and *need only* be specified in *one* of the repeating groups. However, if specified in more than one of the associated repeating groups, the value of all Character Rotation (X’26’) triplets must be identical.
- A Encoding Scheme ID (X’50’) triplet may be specified in *any* of the repeating groups associated with the font and *need only* be specified in *one* of the repeating groups. However, if specified in more than one of the associated repeating groups, the value of all Encoding Scheme ID (X’50’) triplets must be identical.
- A Font Horizontal Scale Factor (X’5D’) triplet may be specified in *any* of the repeating groups associated with the font and *need only* be specified in *one* of the repeating groups. However, if specified in more than one of the associated repeating groups, the value of all Font Horizontal Scale Factor (X’5D’) triplets must be identical.
- A Font Resolution and Metric Technology (X’84’) triplet may be specified in *any* of the repeating groups associated with the font and *need only* be specified in *one* of the repeating groups. If specified in more than one of the associated repeating groups, the last specified Font Resolution and Metric Technology (X’84’) triplet is used.

Using the X'50' Triplet to Specify Encoding

If the optional ESidUD parameter is included, the following ESidCP and ESidUD combinations are allowed in the X'50' triplet when specified in an MCF repeating group:

Table 18. Valid ESidCP/ESidUD Combinations for the MCF

ESidUD	ESidCP
X'7200'—UTF-16, including surrogates; byte order is big endian (UTF-16BE)	X'8200'—Unicode Presentation; byte order is big endian

Architecture Note: The following additional ESidUD/ESidCP combinations are supported in the AFP Line Data architecture when the X'50' triplet is specified on the MCF in a Page Definition. Note that for the combination ESidUD = X'7200' and ESidCP = X'2100', it is assumed that the user data only uses UTF-16 code points X'0020'–X'007F', since these are the only UTF-16 code points that transform to one-byte ASCII code points. Similarly, for the combination ESidUD = X'7807' and ESidCP = X'2100', it is assumed that the user data only uses UTF-8 code points X'20'–X'7F', since these are the only UTF-8 code points that transform to one-byte ASCII code points.

ESidUD	ESidCP
X'7200'—UTF-16, including surrogates; byte order is big endian (UTF-16BE)	X'2100'—PC-Data SBCS (ASCII-based)
X'7807'—UTF-8	X'2100'—PC-Data SBCS (ASCII-based)

MCF Exception Condition Summary

- A X'04' exception condition exists when any repeating group does not contain one of the following:
  - A Fully Qualified Name type X'84' (Coded Font Reference)
  - A Fully Qualified Name type X'85' (Code Page Name Reference) and a Fully Qualified Name type X'86' (Font Character Set Name Reference)
  - A Fully Qualified Name type X'8E' (Coded Font Name Reference)
- A X'02' exception condition exists when:
  - A Fully Qualified Name (X'02') triplet other than a type X'07' (Font Family Name), a type X'08' (Font Typeface Name), type X'84' (Coded Font Reference), type X'85' (Code Page Name Reference), type X'86' (Font Character Set Name Reference), or a type X'8E' (Coded Font Name Reference) appears within any repeating group.
  - A Resource Local Identifier (X'24') triplet type other than X'05' appears within any repeating group.
- A X'01' exception condition exists when any of the following conditions are encountered in *any* of the repeating groups:
  - A Fully Qualified Name type X'84' (Coded Font Reference) and a Fully Qualified Name of either type X'85' (Code Page Name Reference) or type X'86' (Font Character Set Name Reference)
  - A Fully Qualified Name type X'8E' (Coded Font Name Reference) and a Fully Qualified Name of either type X'85' (Code Page Name Reference) or type X'86' (Font Character Set Name Reference)
  - A Fully Qualified Name type X'84' (Coded Font Reference) and a Fully Qualified Name type X'8E' (Coded Font Name Reference)
  - A Fully Qualified Name type X'84' (Coded Font Reference) that identifies a font encoded using the EBCDIC Presentation double-byte encoding scheme

(encoding scheme ID X'62*nn*') or the EBCDIC Presentation single-byte encoding scheme (encoding scheme ID X'61*nn*'), and a Resource Section Number with a value other than X'00'

- A Fully Qualified Name type X'8E' (Coded Font Name Reference) that identifies a font encoded using the EBCDIC Presentation double-byte encoding scheme (encoding scheme ID X'62*nn*') or the EBCDIC Presentation single-byte encoding scheme (encoding scheme ID X'61*nn*'), and a Resource Section Number with a value other than X'00'
- Multiple Fully Qualified Names of the same type
- Multiple triplets of the same type, except Fully Qualified Name (X'02') triplet
- An Encoding Scheme ID where either the encoding scheme or the bytes-per-code-point indicator do not match the characteristics of the specified code page

- A X'01' exception condition exists when any of the following conditions are encountered *within the same* Map Coded Font structured field:
  - The Resource Local Identifier value is repeated in two or more repeating groups that do not map to the same double-byte font using a Fully Qualified Name type X'85' (Code Page Name Reference) and a Fully Qualified Name type X'86' (Font Character Set Name Reference).
  - The Resource Local Identifier value is repeated in two or more repeating groups that are not contiguous.
  - The Resource Local Identifier value is repeated in two or more repeating groups that do not each have a valid, unique Resource Section Number value.
  - The Resource Local Identifier value is repeated in two or more repeating groups that have different Character Rotation values.
  - The Resource Local Identifier value is repeated in two or more repeating groups that have different Encoding Scheme ID values.
  - The Resource Local Identifier value is repeated in two or more repeating groups that have different Font Horizontal Scale Factor values.

Medium Descriptor (MDD)

The Medium Descriptor structured field specifies the size and orientation of the medium presentation space for all sheets that are generated by the medium map that contains the Medium Descriptor structured field.

MDD (X'D3A688') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A688'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	XmBase	X'00'–X'01'	Medium unit base for the X axis: **X'00'** 10 inches **X'01'** 10 centimeters	M	X'06'
1	CODE	YmBase	X'00'–X'01'	Medium unit base for the Y axis: **X'00'** 10 inches **X'01'** 10 centimeters	M	X'06'
2–3	UBIN	XmUnits	1–32767	Medium units per unit base for the X axis	M	X'06'
4–5	UBIN	YmUnits	1–32767	Medium units per unit base for the Y axis	M	X'06'
6–8	UBIN	XmSize	1–32767	Medium extent for the X axis	M	X'06'
			X'000000'	X-axis extent not specified		
			X'FFFFFF'	Presentation process default		
9–11	UBIN	YmSize	1–32767	Medium extent for the Y axis	M	X'06'
			X'000000'	Y-axis extent not specified		
			X'FFFFFF'	Presentation process default		
12	BITS	MDDFlgs		Specify control information for the media. See “MDD Semantics” for bit definitions.	M	X'06'
13–*n*		Triplets		See “MDD Semantics” for triplet applicability.	O	X'10'

MDD Semantics

XmBase Specifies the unit base for the X axis of the medium coordinate system.

YmBase Specifies the unit base for the Y axis of the medium coordinate system.

Note: A X’01’ exception condition exists if the XmBase and YmBase values are not identical.

XmUnits Specifies the number of units per unit base for the X axis of the medium coordinate system.

YmUnits Specifies the number of units per unit base for the Y axis of the medium coordinate system.

XmSize Specifies the extent of the medium presentation space along the X axis. This is also known as the medium's size in the X-direction. A value of X'000000' indicates that the extent along the X axis is not specified and the current medium size in the X-direction is used. A value of X'FFFFFF' indicates that a presentation process default should be used for the X-axis extent.

YmSize Specifies the extent of the medium presentation space along the Y axis. This is also known as the medium's size in the Y-direction. A value of X'000000' indicates that the extent along the Y axis is not specified and the current medium size in the Y-direction is used. A value of X'FFFFFF' indicates that a presentation process default should be used for the Y-axis extent.

Application Note: The following parameter values match the MO:DCA–P IS/1 and IS/2 ranges for page size, as specified in the PGD.

- XmBase = YmBase = X'00' (10 inches)
- XmUnits = YmUnits = 2400 or 14400 (240 units per inch or 1440 units per inch)
- XmSize and YmSize are in the range of 1 to 5461 when using 240 units per inch, and 1 to 32767 when using 1440 units per inch

Larger medium extents can be specified by, for example, using measurement units of 240 units per inch and medium extents in the range 5462 to 32767.

MDDFlgs Specify control information for the media.

Bit **Description**

0 Medium orientation enablement for cut-sheet printers.

B'0' Do not pass the medium orientation specified on this structured field to cut-sheet printers; the medium orientation on such printers is always defined to be X'00' (portrait).

B'1' Pass the medium orientation specified on this structured field to cut-sheet printers.

If this parameter is not specified, the architected default for MDDFlgs bit 0 is B'0' (do not pass the medium orientation to cut-sheet printers). Note that the medium orientation is always passed to continuous-forms printers. It is always passed to cut-sheet printers when N-up presentation is active. Note also that a continuous-forms printer in cut-sheet emulation (CSE) mode is treated as a continuous-forms printer when processing the MDDFlgs parameter.

1–7 Reserved; all bits must be B'0'.

Triplets Appear in the Medium Descriptor structured field as follows:

Triplet	Type	Usage
X'68'	Medium Orientation	Optional. May occur once. Specifies the orientation of the medium presentation space on the physical medium. See "Medium Orientation Triplet X'68'" on page 386. If this triplet is not specified, the architected default for the medium orientation is X'00' (portrait).

Map Data Resource (MDR)

The Map Data Resource structured field 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.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same. For TrueType/OpenType fonts, optimal performance can be achieved by using UTF-16BE as the encoding scheme.

MDR (X'D3ABC3') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABC3'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One to 254 repeating groups in the following format:						
0–1	UBIN	RGLength	14–(*n*+1)	Total length of this repeating group	M	X'06'
2–*n*		Triplets		See “MDR Semantics” for triplet applicability.	M	X'14'

MDR Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Data Resource structured field repeating groups as follows. For examples of the triplet groups that can be specified for various types of MDR repeating groups, see Figure 56 on page 234.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once in each repeating group. Specifies the reference to the resource object. The GID is used to locate the resource object in the resource hierarchy, which may include the presentation device, and must match the identifier for an object or a X'01' exception condition exists. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name types that may appear are: • **X'84'**—*Begin Resource Object Reference*, which is used to map an IOCA image object. The GID is used to locate the resource object in the resource hierarchy, which may include the presentation device, and must match the identifier for an object or a X'01' exception condition exists. • **X'CE'**—*Other Object Data Reference*, which is used to map a data object whose format may or may not be defined by an IBM presentation architecture. The GID is used to locate the object in the resource hierarchy, which may include the presentation device, and must match the identifier for an object or a X'01' exception condition exists. This FQN type may not be used to map OCA objects, that is, IOCA, GOCA, BCOCA, or PTOCA objects. FQN type X'84' is used to map IOCA objects. • **X'DE'**—*Data Object External Resource Reference*, which is used to map a resource object that is used by a data object. The GID is used to locate the resource object in the resource hierarchy, which may include the presentation device, and must match the identifier for an object resource or a X'01' exception condition exists. Resources that are used by data objects that may themselves be processed as resources are called *secondary resources*. See "Secondary Resource Objects" on page 14. Note that in MO:DCA-P data streams, the FQNX'84' and FQNX'CE' triplets may not appear on an MDR that is specified in an OEG for a data object, or a X'02' exception condition exists. *Continued on next page*

Triplet	Type	Usage
X'02' (continued)	Fully Qualified Name	The reference in the FQN type X'84' and the FQN type X'CE' triplets may be specified in the following format: • If FQNFmt = X'00', the reference is made with a character-encoded name. See "External Resource Naming Conventions" on page 80 for a description of the naming conventions used in AFP environments. The reference in the FQN typeX'DE' triplet may be specified in one of the following two formats: • If FQNFmt = X'00', the reference is made with a character-encoded name. See "External Resource Naming Conventions" on page 80 for a description of the naming conventions used in AFP environments. • If FQNFmt = X'10', the reference is made with a ASN.1 OID encoded using the definite short form. This format provides a unique and system-independent method to identify and reference an object. It may be used to select objects that are resident in, or have been captured by, the presentation device. Such an identifier is referred to as an *object OID*. When FQNFmt X'10' (object OID) is used to reference a resource that is not resident in the device, the object itself must be carried in a valid MO:DCA structure that specifies the same OID on the Begin structured field, or a X'02' exception condition exists. When a FQN type X'DE' triplet with FQNFmt X'00' is used to reference a data-object font, the GID is a full font name that uniquely identifies the font. The encoding for this character string is specified by the X'01' triplet, which can be located either in this structured field or in the MO:DCA document hierarchy. **Implementation Note:** Not all AFP presentation servers support the inheritance of encoding scheme from higher levels of the document hierarchy, therefore the X'01' triplet should be specified directly on the MDR. See "Using the MDR to Map a TrueType/OpenType Font" on page 230. **Application Note:** When a full font name is specified in a Resource Access Table (RAT), the encoding for the name is UTF-16BE. This encoding is characterized by CCSID 1200 (X'04B0'). A performance benefit may be achieved if the full font name specified on the MDR—which is used to index the RAT—already uses this encoding, thereby eliminating the need for an encoding conversion. If an IOB is used to reference the mapped object, the IOB must specify the same reference, using the same FQNFmt, as the MDR.
X'10'	Object Classification	Mandatory if the repeating group specifies a Fully Qualified Name type X'CE'—Other Object Data Reference, or a Fully Qualified Name type X'DE'—Data Object External Resource Reference, in which case it must occur once in the repeating group and identifies the resource type. See "Object Classification Triplet X'10'" on page 335.

Triplet	Type	Usage
X'22'	Extended Resource Local Identifier	Mandatory in MO:DCA-L data streams. Must occur once in each repeating group when the MDR is used in MO:DCA-L data streams to map a resource with a FQN type X'84'—Begin Resource Object Reference. This triplet is not allowed in all other cases, and if specified, is ignored. See "Extended Resource Local Identifier Triplet X'22'" on page 348. The only Extended Resource Local Identifier type that may appear is **X'10'**—*Image*. Within the same Map Data Resource structured field, it is not permissible to map the same Resource Local ID to more than one resource object of the same type or a X'01' exception condition exists. However, two or more repeating groups within the same Map Data Resource structured field may be used to map different LIDs to the same resource object.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317. **Implementation Note:** Not all AFP presentation servers support the inheritance of encoding scheme from higher levels of the document hierarchy, therefore it is recommended that this triplet be specified directly on the MDR if required by a parameter such as the FQN type X'DE' triplet.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once in each repeating group that also specifies a FQN type X'DE' triplet, but only: • when the MDR is specified in the OEG of a data object • when the MDR references a data-object font and – the MDR is in the AEG for PTOCA text, or – the MDR is in the OEG for BCOCA or AFP GOCA Text, in which case this triplet is mandatory. When the MDR is in the AEG for BCOCA or AFP GOCA text, this triplet is also mandatory but the LID is not used; ID X'FE' may be specified in that case. This triplet is ignored in all other cases. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'BE'**—*Data Object Internal Resource Reference*. The identifier is used internally by the data object to reference the resource whose external identifier is specified by the FQN type X'DE' triplet. The identifier must be specified using FQNFmt X'00', which, for this FQN type, indicates that the data type is defined by the specific data object that generates the internal resource reference and is undefined (UNDF) at the MO:DCA data stream level. **Architecture Note:** 1. For data-object fonts referenced by AFP text (PTOCA), AFP graphics (GOCA), and AFP bar code (BCOCA) objects, the data type of the internal identifier is a CODE that consists of a one-byte local ID. 2. For purposes of Print Services Facility resource management, each MDR that is specified in an OEG for a data-object font must have a corresponding MDR mapping the same font in the AEG for the page or overlay. The local ID used in the page or overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG. When both the FQN type X'DE' and the FQN type X'BE' triplets are specified on an MDR repeating group, they map the internal resource identifier to the external resource identifier. Resources that are used by data objects that may themselves be processed as resources are called *secondary resources*. See "Secondary Resource Objects" on page 14.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur once in each repeating group. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is: **X'85'**—*Code Page Name Reference*. Only used when the MDR references a data-object font with the FQN type X'DE' triplet, in which case this triplet specifies the name of an IBM code page that defines the encoding in the user data. It is ignored in all other cases. **Application Note:** In AFP environments, the name consists of 8 characters and follows the naming conventions for AFP code pages defined in *Font Summary for AFP Font Collection*. An example of a code page name is T1V10500. Either this triplet or the X'20' triplet may be specified. If the MDR repeating group specifies both the FQN type X'85' triplet and a X'20' triplet, the FQN type X'85' triplet is ignored. **Application Notes:** 1. The referenced code page must map code points to IBM Graphic Character Global Identifiers (GCGIDs). The presentation device maps GCGIDs to the UTF-16 code points in the font. 2. If the user-data encoding is double-byte, the referenced code page must be a valid double-byte code page.
X'20'	Font Coded Graphic Character Set Global Identifier	Optional. May occur once in each repeating group. Only used when the MDR references a data-object font with the FQN type X'DE' triplet, in which case this triplet specifies the Code Page Global Identifier (CPGID) and Graphic Character Set Global Identifier (GCSGID) of an IBM code page that defines the encoding in the user data. It is ignored in all other cases. See “Font Coded Graphic Character Set Global Identifier Triplet X'20'” on page 345. Either this triplet or the FQN type X'85' triplet may be specified. If the MDR repeating group specifies both the FQN type X'85' triplet and a X'20' triplet, the FQN type X'85' triplet is ignored. **Application Notes:** 1. The referenced code page must map code points to IBM Graphic Character Global Identifiers (GCGIDs). The presentation device maps GCGIDs to the UTF-16 code points in the font. 2. If the user-data encoding is double-byte, the referenced code page must be a valid double-byte code page.

Triplet	Type	Usage
X'50'	Encoding Scheme ID	Optional. May occur once in each repeating group. Only used when the MDR references a data-object font and the encoding in the user data is different than the encoding in the referenced font. In that case this triplet specifies the encoding in the user data. The user data encoding can be specified in two ways: • With an IBM code page identifier—specified either as a CPGID in the X'20' triplet or as a name in the FQN type X'85' triplet—and an optional X'50' triplet with the ESidCP parameter that specifies . the encoding for the code page. The ESidUD parameter in the X'50' triplet is ignored in this case since the user data encoding is defined by the code page. • With the ESidUD parameter in the X'50' triplet and no IBM code page identifier. The ESidCP parameter in the X'50' triplet is ignored in this case. For a list of valid ESidUD and ESidCP combinations, see “Using the X'50' Triplet to Specify Encoding.” If the X'50' triplet is omitted and a code page is specified—either as a CPGID in the X'20' triplet or as a name in the FQN type X'85' triplet—the architected default is that the ESidUD and ESidCP parameters match the code page encoding. If the X'50' triplet is omitted and no code page is specified the architected default is that the ESidUD = ESidCP = X'7200' (UTF-16), which matches the encoding in the data object font. See “Encoding Scheme ID Triplet X'50'” on page 371.
X'8B'	Data-Object Font Descriptor	Optional. May occur once in each repeating group. Only used when the MDR references a data-object font with the FQN type X'DE' triplet, in which case this triplet specifies information used to render the font, and is mandatory. It is ignored in all other cases. See “Data-Object Font Descriptor Triplet X'8B'” on page 420.

Application Note: An non-OCA data object or an IOCA image object that is included on a page or overlay with an IOB may optionally be mapped with an MDR in the AEG for that page or overlay. If such a mapping is specified, the object is sent to the presentation device as a resource object and can be presented multiple times using an include command. Such an object is sometimes called a *hard* object. If a mapping is not specified, the object is sent to the presentation device as part of the page or overlay and is sometimes called a *soft* object.

Using the X'50' Triplet to Specify Encoding

Table 19 shows the ESidCP and ESidUD combinations that are allowed in the X'50' triplet when the MDR references a TrueType/OpenType font with EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001'):

Table 19. Valid ESidUD/ESidCP Combinations for the MDR

ESidUD	ESidCP
Not specified	X'2100'—PC-Data SBCS (ASCII)
Not specified	X'6100'—EBCDIC SBCS
Not specified	X'6200'—EBCDIC DBCS
X'7807'—UTF-8	Ignored

Using the MDR to Map a TrueType/OpenType Font

Font Name

When the MDR is used to map a data-object font resource that is a TrueType/OpenType font and specifies a FQN type X'DE' triplet with FQNFmt = X'00', the character string that identifies the font must be the *full font name* specified in a name record in the mandatory Naming Table of the font file. This parameter is specified in a name record with Name ID 4. An example of a full font name is *Times New Roman Bold*. Two characteristics of the full font name must be taken into account when using it to reference a TrueType/OpenType font: language and encoding.

- *Language*. The full font name may be specified in a number of languages. The language used for a given name record is specified with a language identifier (LCID). For example, English-United States is assigned LCID X'0409' (1033). The language used to specify the full font name in the FQN X'DE' triplet may be any of the languages specified in a name record for the full font name with the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001').
- *Encoding*. The encoding used to specify the character string in the FQN X'DE' triplet is defined by a Coded Graphic Character Set Global Identifier (X'01') triplet that precedes the FQN X'DE' triplet. This triplet may be specified on the MDR or on a structured field that is higher in the document hierarchy than the MDR: for example on the BPG for the page that contains the MDR or on the BDT for the document. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317 for a definition of the scoping rules for the X'01' triplet. Note that the encoding for the FQN X'DE' triplet need not match the encoding for the full font name in the font Naming Table.

Implementation Note: Not all AFP presentation servers support the inheritance of encoding scheme from higher levels of the document hierarchy, therefore it is recommended that this triplet be specified directly on the MDR if required by a parameter such as the FQN type X'DE' triplet.

Font Install Program

In general, the full font name does not provide sufficient information to find the font resource on a given platform. Additional information such as the file name is normally required to locate the font resource. The mapping from full font name to file name is provided for each platform that requires this by a font install program. This program builds a Resource Access Table (RAT) that must, at minimum, contain the following information:

- The full font name encoded in UTF-16. This full font name is specified multiple times in all languages used in the naming table. The UTF-16 encoding matches the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001') in the Naming Table. Figure 55 on page 231 shows the full font name of the MS Mincho font in two different languages.

Platform ID = 3
Encoding ID = 1
Language ID = 1033 (English - United States)
Field Value = 004D 0053 0020 004D 0069 006E 0068 006F
Example: ms mincho

Platform ID = 3
Encoding ID = 1
Language ID = 1041 (Japanese)
Field Value = FF2D FF33 0020 660E 671D
Example: ＭＳ 明朝

Figure 55. Example of a Full Font Name in Two Languages

- A mapping of the full font name—in each language—to the name of the file that contains the font. For example, if the Naming Table contains two name records for the full font name (Name ID 4), one in English-United States (LCID = X'0409') and one in German-Standard (LCID = X'0407'), both in the encoding defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001'), the font install table must map both language versions of this full font name to the same file name.
- If the font also has an object OID assigned and can therefore be resident in the printer, the mapping from full font name to font file name also includes the object OID for the font. This allows use of the resident version of the font and avoids a font download.
- If the font is contained in a TrueType Collection file (TTC), the full font name must be mapped to the file name of the TTC. A TTC consists of a collection of TrueType/OpenType font files which may share some of the font tables. The table directories for each font file are indexed from a single TTC Header Table. If the collection has an object OID assigned, the mapping from full font name to collection file name also includes the object OID for the collection. When a mapped TrueType/OpenType font is part of a TTC, the complete TTC (if not already in the presentation device) is downloaded to the device, which must be able to index the required font in the collection.
- If the font has linked fonts the RAT must link the full font name of the font to the full font names of the linked fonts. When a font has linked fonts, it is referred to as a *base font* to differentiate it from its linked fonts. Linked fonts are TTFs/OTFs that can be used to extend the character sets in a base font or to add user-defined characters (UDCs) to the base font. All linked fonts for a base font (if not already in the presentation device) are downloaded to the device and are treated as extensions to the base font by the device. The order in which the linked fonts are specified determines the order in which they are processed by the device. The base font is always processed first, followed by the first-specified linked font, followed by the next-specified linked font, and so on. The last linked font is processed last.

The Resource Access Table (RAT) used in AFP environments is defined in "The Resource Access Table (RAT)" on page 491.

TrueType/OpenType Font Resources in a Resource Library: When TrueType/OpenType fonts are installed in a resource library, they must not be wrapped with a MO:DCA object envelope such as BOC/EOC, that is, they must be installed in their raw source format. This allows the font resources to be used by all system components, particularly those that do not understand MO:DCA object envelopes such as BOC/EOC. Any of the necessary information that such an envelope normally provides, such as an object OID, is associated with the raw font resource by the Resource Access Table (RAT). The font install program must ensure that the TrueType/OpenType font resources are installed in this manner. BOC/EOC object containers for TrueType/OpenType font resources are only supported when such resources are placed into a print-file resource group, in which case they are mandatory.

Architecture Note: In AFP environments, when a TrueType/OpenType font resource is carried in a BOC/EOC container in an external (printfile-level) resource group, the container must be wrapped with a BRS/ERS envelope.

TrueType/OpenType Font Resources in an External (Printfile-level) Resource Group: TrueType/OpenType fonts (TTFs/OTFs), TrueType/OpenType fonts that are used as linked fonts, and TrueType/OpenType font collections (TTCs), may be carried in the resource group for a print file. This is called a print-file-level resource group, and these resources are said to be *inline*. When presentation servers search for a font that is referenced in the data stream, such a resource group is searched ahead of system-level resource libraries, and if an inline font is found it must be used in place of the system-level font. To support this hierarchy, presentation servers process a TrueType/OpenType font reference in an MDR for inline resources as follows:

1. The resource group, if present, is searched for a font (TTF/OTF) container or a collection (TTC) container that specifies a matching full font name.
   - A font container specifies the full font name using a FQN type X'01' triplet on the Begin Resource (BRS) structured field for the font container.
   - A collection container specifies the full font name of a font in the collection using a FQN type X'6E'—Data Object Font Base Font Identifier triplet on the BRS of the collection container.

   The first matching font container or collection container is used. If a collection containing the font is found, the complete TTC (if not already in the presentation device) is downloaded to the device, which must be able to index the required font in the collection. The font container or collection container may also specify one or more linked fonts for the referenced font.
   - On a font container, linked fonts for the base font are specified with FQN type X'7E'—Data-object Font Linked Font Identifier triplets, which carry the full font name of the linked fonts, on the BRS of the font container.
   - On a collection container, linked fonts are specified with FQN type X'7E' triplets that immediately follow the FQN type X'6E' triplet for the base font on the BRS of the collection container. Note that if the base font is specified in multiple languages using multiple FQN type X'6E' triplets, each instance of the FQN type X'6E' triplet must be followed by the sequence of FQN type X'7E' triplets that identify the linked fonts for the base font.

   The full font names for the linked fonts are used in turn to search the resource group for a font container or a collection container that carries a font that matches the full font name of the linked font. On a font container, the linked font name is matched against the FQN type X'01' triplet on the BRS; on a collection container it is matched against the FQN type X'6E' triplets on the BRS.

- The first matching font container or collection container is used, and its font is processed as a linked font for the base font. Multiple linked fonts may be specified, and the order in which they are specified on the BRS of the font container or collection container determines the order in which they are processed. The base font is always processed first, followed by the first-specified linked font, followed by the next-specified linked font, and so on. The last linked font is processed last.
- If a linked font cannot be found in either an inline font container or an inline collection container, the full font name of the linked font is used to index the RAT to locate the linked font in a resource library. If a specified linked font cannot be found in the resource group or in a resource library, a X'04' exception condition exists.

Only one level of linking is supported. That is, if a linked font specifies its own linked fonts, either with FQN type X'7E' triplets on its inline container or with linked font pointers in the RAT, these ‘secondary’ linked fonts are not processed as linked fonts for the original base font.

2. If a font matching the MDR reference is not found in an inline font container or in an inline collection container, the presentation server accesses the RAT with the full font name to locate the referenced font in a resource library. In this case, all linked fonts are specified in the RAT repeating group for the referenced font, and the order in which they are specified determines the order in which they are processed. Both inline linked fonts and library-based linked fonts are used, and the print-file-level resource group is always searched for linked fonts ahead of the resource library. The resource group search includes font containers, in which case the linked font name is matched against the FQN type X'01' triplet on the BRS of the font container, and collection containers, in which case the linked font name is matched against the FQN type X'6E' triplets on the BRS of the collection container.

MO:DCA–P Repeating Group Mapping an IOCA Image
- Fully Qualified Name (X’02’) triplet, type X'84'—*Begin Resource Object Reference*.

MO:DCA–L Repeating Group Mapping an IOCA Image
- Fully Qualified Name (X’02’) triplet, type X'84'—*Begin Resource Object Reference*.
- Extended Resource Local Identifier (X’22’) triplet.

MO:DCA–P Repeating Group Mapping a PDF Object
- Fully Qualified Name (X’02’) triplet, type X'CE'—*Other Object Data Reference*.
- Object Classification (X’10’) triplet.

MO:DCA–P Repeating Group Mapping a PDF Resource
- Fully Qualified Name (X’02’) triplet, type X'DE'—*Data Object External Resource Reference*.
- Object Classification (X’10’) triplet.

MO:DCA–P Repeating Group Mapping a TrueType/OpenType Font (user encoding = font encoding)
- Coded Graphic Character Set Global Identifier (X’01’) triplet.
- Fully Qualified Name (X’02’) triplet, type X'DE'—*Data Object External Resource Reference*.
- Fully Qualified Name (X’02’) triplet, type X'BE'—*Data Object Internal Resource Reference*.
- Object Classification (X’10’) triplet.
- Data-Object Font Descriptor (X’8B’) triplet.

MO:DCA–P Repeating Group Mapping a TrueType/OpenType Font (user encoding = UTF-8)
- Coded Graphic Character Set Global Identifier (X’01’) triplet.
- Fully Qualified Name (X’02’) triplet, type X'DE'—*Data Object External Resource Reference*.
- Fully Qualified Name (X’02’) triplet, type X'BE'—*Data Object Internal Resource Reference*.
- Object Classification (X’10’) triplet.
- Encoding Scheme ID (X’50’) triplet.
- Data-Object Font Descriptor (X’8B’) triplet.

MO:DCA–P Repeating Group Mapping a TrueType/OpenType Font (user encoding defined by EBCDIC/ASCII code page)
- Coded Graphic Character Set Global Identifier (X’01’) triplet.
- Fully Qualified Name (X’02’) triplet, type X'DE'—*Data Object External Resource Reference*.
- Fully Qualified Name (X’02’) triplet, type X'BE'—*Data Object Internal Resource Reference*.
- Object Classification (X’10’) triplet.
- Font Coded Graphic Character Set Global Identifier (X’20’) triplet.
- Encoding Scheme ID (X’50’) triplet.
- Data-Object Font Descriptor (X’8B’) triplet.

Figure 56. Examples of MDR Repeating Groups

MDR Exception Condition Summary

- A X’02’ exception condition exists when:
  - A Fully Qualified Name (X’02’) triplet other than a type X'84' (Begin Resource Object Reference), a type X’85’ (Code Page Name Reference), a type X’CE’ (Other Object Data Reference), a type X’DE’ (Data Object External Resource Reference), or a type X’BE’ (Data Object Internal Resource Reference) appears within any repeating group.
  - An Extended Resource Local Identifier (X’22’) triplet type other than X'10' appears within any repeating group.
  - The same resource reference is specified in more than one repeating group.

  - The resource reference is specified using FQNFmt X'10' (object OID), but the object either is not carried in a valid MO:DCA structure or is carried in a valid MO:DCA structure but does not have a matching object OID.
- A X’01’ exception condition exists when:
  - A Begin Image structured field with the same name as that specified on the type X'84' (Begin Resource Object Reference) Fully Qualified Name triplet cannot be located.
  - A resource with the same identifier as that specified on the type X’CE’ (Other Object Data Reference) Fully Qualified Name triplet or on the type X’DE’ (Data Object External Resource Reference) Fully Qualified Name triplet cannot be located.
  - The same repeating group contains an invalid number or combination of Fully Qualified Name triplets.
  - The same repeating group contains multiple Extended Resource Local Identifier (X'22') type X'10' triplets.
  - The same Resource LID is mapped to more than one resource object of the same type within the same structured field.

Medium Finishing Control (MFC)

The Medium Finishing Control structured field specifies the finishing requirements for physical media. Finishing can be specified for a media *collection* at the printfile level or at the document level by placing the MFC in the document environment group (DEG) of the form map. Finishing can be specified for a media collection at the medium map level by placing the MFC in a medium map. Finishing can be specified for individual media, or sheets, at the medium map level by placing the MFC in a medium map.

- When the MFC is specified in the document environment group (DEG) of the form map, its scope is specified to be one of the following:
  - The complete printfile
  - Each individual document in the printfile
  - A selected document in the printfile

  If the scope is the printfile, the MFC defines *printfile-level finishing*, and all media in the printfile are collected for finishing in a *printfile-level media collection*. The specified finishing operations are applied to the complete collection, that is, the complete printfile. Note that the printfile-level media collection excludes other material that may accompany the printfile, such as header pages, trailer pages, and message pages. Such material can be generated as a separate printfile. Therefore, it may be collected in a separate printfile-level media collection and processed with separate finishing operations.

  If the scope is each individual document in the printfile, the MFC defines *document-level finishing*, and all media in each document are collected for finishing in a *document-level media collection*. The specified 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 scope is a selected document in the printfile, the MFC defines *document-level finishing*, and all media in the selected document are collected for finishing in a *document-level media collection*. The specified finishing operations are applied to this single collection. 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.

  A single printfile-level MFC, a single document-level MFC for all documents, or multiple document-level MFCs for single documents can be specified in the DEG. If a printfile-level MFC and document-level MFCs are specified in the same DEG, document-level finishing is applied to the selected documents, and printfile-level finishing is applied to the complete print file.

  If a document is selected for finishing using an MFC whose scope is each document in the printfile, and if it is also selected by one or more MFCs whose scope is a single document, the finishing operations that apply to each document in the printfile are applied before the finishing operations that apply to a single document.
- When the MFC is specified in a medium map, its scope is specified to be one of the following:
  - Each medium, or sheet, generated by the medium map. When the scope is each medium in the medium map, the MFC defines *medium-map-level sheet finishing*, and the specified finishing operations are applied to each medium, *not* to the media collection.
  - The collection of media, or the collection of sheets generated by the medium map. In this case the MFC defines *medium-map-level group finishing*, and all

media generated by the medium map are collected for finishing in a *medium-map-level sheet finishing*. The specified finishing operations are applied to this single collection.

When an MFC is specified both in a medium map and in the DEG, both sets of finishing operations are applied according to their scope, as long as the operations are compatible. For rules on how finishing operations are nested, see “Finishing Operation Nesting Rules” on page 242. Note that not all combinations of finishing operations are compatible. Compatible combinations of finishing operations are presentation-device specific.

MFC (X'D3A088') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A088'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	BITS	MFCFlgs		See “MFC Semantics” on page 238 for the MFCFlgs parameter bit definitions.	M	X'06'
1				Reserved; must be zero	M	X'06'
2	CODE	MedColl	X'00'–X'02'	Boundary conditions for medium-map-level sheet collection **X'00'** No sheet collection processed at the medium map level **X'01'** Begin medium-map-level sheet collection **X'02'** Continue medium-map-level sheet collection	M	X'06'
3	CODE	MFCScpe	X'01'–X'05'	MFC Scope: **X'01'** Printfile-level MFC **X'02'** Document-level MFC, all documents **X'03'** Document-level MFC, selected document **X'04'** Medium-map-level MFC, each medium or sheet **X'05'** Medium-map-level MFC, collection of media or sheets	M	X'06'
4–*n*		Triplets		See “MFC Semantics” on page 238 for triplet applicability.	M	X'14'

MFC Semantics

MFCFlgs The following flags are defined:

Bit	Description
0	Activate Medium Finishing Control **B'0'** Process this structured field as a NoOp. **B'1'** Process this structured field as specified.
1–7	Reserved; all bits must be B'0'.

MedColl is a parameter that defines the boundary conditions for the media collection generated by this medium map. This parameter is only processed if MFCScpe = X'05'—medium-map-level MFC, collection of sheets. It is ignored in all other cases.

Value **Scope**

X'00' No sheet collection is to be processed at the medium map level. This value should be specified when MFCScpe is set to values other than X'05'—medium-map-level MFC, collection of sheets. If this value is specified when MFCScpe is set to X'05', a X'01' exception condition exists.

X'01' Begin medium-map-level sheet collection.

This causes a sheet eject to be generated and starts a medium-map-level sheet collection for the finishing operation specified on this MFC. Note that if a collection for this *same* finishing operation is already in progress from a previous medium map, that collection is terminated and the specified finishing operation is applied. The sheet collection that is started by this MFC continues until:

1. A medium map is invoked that does not contain an MFC with MFCScpe= X'05' and MedColl = X'02' (Continue) for this *same* operation.
2. A medium-map-level finishing operation with MFCScpe = X'05' that is nested outside this operation is applied.
3. End of document is reached.

When this sheet collection is terminated for any of the above reasons, the specified finishing operation is applied to the collection, and a sheet eject is generated.

X'02' Continue medium-map-level sheet collection.

This continues a medium-map-level sheet collection that was started for the *same* finishing operation by a previous medium map. The sheet collection that is continued by this MFC continues until:

1. A medium map is invoked that does not contain an MFC with MFCScpe= X'05' and MedColl = X'02' (Continue) for this *same* operation.
2. A medium-map-level finishing operation with MFCScpe = X'05' that is nested outside this operation is applied.
3. End of document is reached.

When a sheet collection is terminated for any of the above reasons, the specified finishing operation is applied to the collection, and a sheet eject is generated.

If the same finishing operation was not previously started, the continue operation request is ignored.

Note that the MFC that continues an operation need not be specified in the same order in the medium map as the MFC that started the operation.

All others
Reserved.

MFCScpe Is a parameter that defines the scope of the finishing operations specified by this MFC structured field.

Value	Scope
X'01'	Printfile-level MFC. The scope of this MFC is the complete printfile. All media in the printfile are collected for finishing in a printfile-level media collection, and the specified finishing operations are applied to this collection.
X'02'	Document-level MFC, all documents. The scope of this MFC is each individual document in the printfile. The media in each document are collected for finishing in a document-level media collection, and the specified finishing operations are applied to each collection individually.
X'03'	Document-level MFC, single document. The scope of this MFC is a single document in the printfile. The document is selected by specifying its position in the printfile using an Object Offset (X'5A') triplet. If this triplet is not specified, the first document in the printfile is selected. The media in this document are collected for finishing in a document-level media collection, and the specified finishing operations are applied to that collection.
X'04'	Medium-map-level MFC, each medium, or sheet. The scope of this MFC is each medium generated by the medium map, and the specified finishing operations are applied to each medium, or sheet, individually.
X'05'	Medium-map-level MFC, collection of media or sheets. The scope of this MFC is the set of media, or sheets, generated by the medium map. All sheets generated by this medium map are collected in a medium-map-level sheet collection, and the specified finishing operations are applied to this collection. The MedColl parameter specifies whether this MFC begins a collection (MedColl = X'01'), or continues a collection (MedColl = X'02').

All others
Reserved

When the MFC is specified in a DEG, the following values for MFCScpe are supported:

X'01'	Printfile-level MFC
X'02'	Document-level MFC, all documents
X'03'	Document-level MFC, single document

If any other value is specified, the MFC is ignored.

When the MFC is specified in a medium map, the following values for MFCScpe are supported:

X'04' Medium-map-level MFC, each medium.

X'05' Medium-map-level MFC, collection of media.

If any other value is specified, the MFC is ignored.

The MedColl and MFCScpe parameters affect the generation of sheet ejects when N-up processing is active. For a description of how sheet and partition ejects are handled when N-up processing is active and an MFC is specified in the medium map, see “Media Eject Control Triplet X'45'” on page 356.

Triplets Appear in the Medium Finishing Control structured field as follows:

Triplet	Type	Usage
X'85'	Finishing Operation	One occurrence of either this triplet or the UP3i Finishing Operation (X'8E') triplet is mandatory. May occur more than once. Specifies finishing operations to be applied to collected media. If this triplet is specified more than once, finishing operations are applied in the order in which the triplets are specified. See "Finishing Operation Triplet X'85'" on page 407. For rules on how finishing operations are nested, see "Finishing Operation Nesting Rules" on page 242. The following finishing operations may be specified when this triplet is specified on the MFC in a DEG: **X'01'** Corner Staple **X'02'** Saddle Stitch Out **X'03'** Edge Stitch **X'04'** Fold In **X'05'** Separation Cut **X'06'** Perforation Cut **X'08'** Center Fold In **X'0A'** Punch **X'12'** Saddle Stitch In If any other finishing operation is specified, this triplet is ignored. The following finishing operations may be specified when this triplet is specified on the MFC in a medium map with MFCScpe = X'04': **X'04'** Fold In **X'05'** Separation Cut **X'06'** Perforation Cut **X'07'** Z-fold **X'08'** Center Fold In **X'0A'** Punch If any other finishing operation is specified, this triplet is ignored. The following finishing operations may be specified when this triplet is specified on the MFC in a medium map with MFCScpe = X'05': **X'01'** Corner Staple **X'02'** Saddle Stitch Out **X'03'** Edge Stitch **X'04'** Fold In **X'05'** Separation Cut **X'06'** Perforation Cut **X'08'** Center Fold In **X'0A'** Punch **X'12'** Saddle Stitch In If any other finishing operation is specified, this triplet is ignored.

Triplet	Type	Usage
X'8E'	UP3i Finishing Operation	One occurrence of either this triplet or the Finishing Operation (X'85') triplet is mandatory. May occur more than once. Specifies finishing operations to be applied to collected media. If this triplet is specified more than once, finishing operations are applied in the order in which the triplets are specified. See the UP3i Finishing Operation triplet description. For rules on how finishing operations are nested, see “Finishing Operation Nesting Rules.” The UP3i Finishing Operation triplet can be specified on the MFC either in a DEG or in a medium map with all architected values for the MFCScpe parameter. There is no architected restriction on which UP3i finishing operations may be specified with MFCScpe = X'04' or MFCScpe = X'05'. However, the UP3i Specification as well as UP3i equipment may limit the scope of UP3i finishing operations; for further information consult the current UP3i Specification. This specification is available on the UP3i home page at *www.up3i.org*.
X'5A'	Object Offset	Optional. If MFCScpe=X'03', may occur once with ObjTpe=X'A8' to specify that documents are the objects to be counted. Specifies how many documents in the print file precede the document to be finished. The offset is measured from the beginning of the print file, so that the first document has offset 0, the second document has offset 1, and the nth document has offset (*n*−1). If this triplet is specified when MFCScpe=X'01', X'02', X'04', or X'05', it is ignored. See “Object Offset Triplet X'5A'” on page 378.

Finishing Operation Nesting Rules

When more than one finishing operation that involves a collection of media is specified for some portion of the printfile, a nesting of the operations is defined first by the scope of the operation (printfile, document, medium), and second by the order of the operation in the data stream. Finishing operations with an inherently broader scope, e.g. operations at the printfile level, are nested outside of finishing operations with an inherently narrower scope, for example, operations at the medium map level.

If more than one operation is specified with the same scope, for example, if two operations are specified at the medium map level, the order of the Finishing Operation (X'85') triplets and of the UP3i Finishing Operation (X'8E') triplets (whether specified on the same MFC or on different MFCs) defines the order of the nesting. In that case, the first finishing operation specified defines the outermost nesting, and the last finishing operation specified defines the innermost nesting.

The following defines how finishing operations are nested starting with the outermost nesting and ending with the innermost nesting.

- Printfile level finishing (outermost level), MFCScpe = X'01'
  - Document-level finishing: each document in the print file, MFCScpe = X'02'
    - Document-level finishing: a selected document in the print file, MFCScpe = X'03'
      - Medium-map-level finishing: collection of sheets (innermost level), MFCScpe = X'05'.

Nesting may in turn affect the scope of a finishing operation. When a finishing operation is applied, all finishing operations nested inside this operation are also applied. Finishing operations that are nested outside this operation are not affected. Note that nesting does not apply to medium-map-level sheet finishing (MFCScpe = X'04'). Such finishing is applied to individual sheets and does not involve starting, continuing, and ending a collection of sheets. Each medium map that is to generate such finishing must specify the operation explicitly.

Implementation Notes:

1. AFP Environments limit the number of finishing operations that can be nested at the medium map level to sixteen. This limit does not apply to nesting at the document or printfile level. For example, if two finishing operations are nested at the medium map level, and these operations are nested within one finishing operation at the document level, which in turn is nested within one finishing operation at the printfile level, the level of nesting counted against the AFP nesting limit is two.
2. In AFP environments, the nesting of identical finishing operations at the medium-map-level is not supported. Two finishing operations are considered identical if they are specified by the same triplet (either the Finishing Operation (X'85') triplet or the UP3i Finishing Operation (X'8E') triplet), and the triplet contents are identical.

Architecture Note: For some printers, the offset stacking function (X'D1*nn*' keyword on the MMC structured field), when invoked inside a document or printfile, cannot be combined with a finishing operation. In this case, the offset stacking request is ignored and the finishing operation is performed.

Architecture Note: Finishing operations may be applied to printfiles that contain a mixture of MO:DCA documents and non-MO:DCA data. The following rules specify how the scope of the finishing operations applies to a printfile that contains line-data and mixed-data documents, with or without BDT/EDT, as well as composed documents. For more information on line data and mixed data, see the *Advanced Function Presentation: Programming Guide and Line Data Reference*.

- If the MFC specifies print-file level finishing, all media in the printfile is collected for finishing in a printfile level media collection, and the finishing operations are applied to the complete collection, that is, the complete printfile.
- If the MFC specifies document-level finishing and selects all documents, the printfile 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 printfile 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.

- If the MFC specifies document-level finishing and selects a single document, the printfile 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.

MFC Exception Condition Summary

- A X'01' exception condition exists when:
  - The FOpCnt parameter in a Finishing Operation (X'85') triplet is non-zero but does not match the specified number of OpPos parameters.
  - The MedColl parameter is X'00' and the MFCScpe parameter is X'05'.

Map Graphics Object (MGO)

The Map Graphics Object structured field specifies how a graphics data object is mapped into its object area.

MGO (X'D3ABBB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABBB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One repeating group in the following format:						
0–1	UBIN	RGLength	5	Total length of this repeating group	M	X'06'
2–4		Triplets		Mapping Option triplet	M	X'14'

MGO Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Graphics Object structured field as follows:

Triplet	Type	Usage
X'04'	Mapping Option	Mandatory. Must occur once. See “Mapping Option Triplet X'04'” on page 332. The valid mapping options for the MGO structured field are: **Value** **Description** **X'10'** Position and trim **X'20'** Scale to fit **X'30'** Center and trim **X'50'** Retired mapping option; see “Retired Parameters” on page 515. **X'60'** Scale to fill **All others** Reserved

Note: If this structured field is not present in the data stream, the architected default is *scale to fit*.

MGO Exception Condition Summary

- A X’02’ exception condition exists when a Mapping Option (X’04’) triplet value of X'00' is specified.
- A X’01’ exception condition exists when the Map Graphics Object structured field contains more than one repeating group.

Map Image Object (MIO)

The Map Image Object structured field specifies how an image data object is mapped into its object area.

MIO (X'D3ABFB') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABFB'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One repeating group in the following format:						
0–1	UBIN	RGLength	5	Total length of this repeating group	M	X'06'
2–4		Triplets		Mapping Option triplet	M	X'14'

MIO Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Image Object structured field as follows:

Triplet	Type	Usage
X'04'	Mapping Option	Mandatory. Must occur once. See “Mapping Option Triplet X'04'” on page 332. The valid mapping options for the MIO structured field are: **Value** **Description** **X'10'** Position and trim **X'20'** Scale to fit **X'30'** Center and trim **X'41'** Migration mapping option: Image point-to-pel. See “Coexistence Parameters” on page 533 for a description. **X'42'** Migration mapping option: Image point-to-pel with double dot. See “Coexistence Parameters” on page 533 for a description. **X'50'** Migration mapping option: Replicate and trim. See “Coexistence Parameters” on page 533 for a description. **X'60'** Scale to fill **All others** Reserved

Note: If this structured field is not present in the data stream, the architected default is *scale to fit*.

MIO Exception Condition Summary

- A X’02’ exception condition exists when a Mapping Option (X’04’) triplet value of X'00' is specified.

- A X′01′ exception condition exists when the Map Image Object structured field contains more than one repeating group.

Medium Modification Control (MMC)

The Medium Modification Control structured field specifies the medium modifications to be applied for a copy subgroup specified in the Medium Copy Count (MCC) structured field.

MMC (X'D3A788') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A788'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	MMCid	1–127	Medium Modification Control identifier	M	X'06'
1	CODE		X'FF'	Constant data	M	X'06'
2–*n*	CODE	Zero or more keywords in ascending order, in the format shown in the following table. When keywords occur in pairs, the ordering applies to the first keyword.				

Keyword ID	Parameter Range	Meaning	M/O	Exc
X'0E'	X'01'–X'20', X'FF'	Horizontal print adjustment; retired for 3800.	O	X'02'
X'90'	X'01'–X'FF'. **Note:** X'00' is not valid with keyword X'9100'	Media destination selector—high	O	X'02'
X'91'	X'01'–X'FF'. **Note:** X'00' is not valid with keyword X'9100'	Media destination selector—low	O	X'02'
X'A0'	X'01'–X'FE'	Fixed medium information: a local identifier for the particular fixed medium information selected	O	X'02'
	X'FF'	Apply all currently supported fixed medium information identifiers		
X'A1'	X'00'	Fixed perforation cut. Apply a perforation cut at a fixed location on the physical medium.	O	X'02'
X'A2'	X'00'	Fixed separation cut. Apply a separation cut at a fixed location on the physical medium.	O	X'02'
X'B4'	X'00'–X'FF'	Presentation subsystem set-up ID: high-order byte	O	X'00'
X'B5'	X'00'–X'FF'	Presentation subsystem set-up ID: low-order byte	O	X'00'
X'D1'	X'00'–X'01'	Offset stack/edge mark change: **X'00'** No offset stack or edge mark change **X'01'** Apply offset stack or edge mark change	O	X'02'

Keyword ID	Parameter Range	Meaning	M/O	Exc
X'E0'	X'01'–X'02'	Media source selection format: **X'01'** Media source selector in Format 1 **X'02'** Media source selector in Format 2	O	X'02'
X'E1'	X'01'–X'04', X'41', X'64'	Media source selector, Format 1: **X'01'–X'04'** Media source ID **X'41'** Envelope media source **X'64'** Manual feed media source	O	X'02'
	X'01'–X'FF'	Media source selector, Format 2		
X'E8'	X'00'–X'FF'	Media type local ID: high-order byte	O	X'02'
X'E9'	X'00'–X'FF'	Media type local ID: low-order byte	O	X'02'
X'F1'	X'00'–X'01'	Forms flash; retired for 3800	O	X'02'
X'F2'	X'01'–X'7F'	Medium overlay local identifier	O	X'02'
X'F3'	X'01'–X'7F'	Text suppression local identifier	O	X'02'
X'F4'	X'01'– X'03'	Duplex control: **X'01'** Simplex **X'02'** Normal duplex **X'03'** Tumble duplex	O	X'02'
X'F8'	X'01'–X'FE', X'FF'	Print quality control: **X'01'** Lowest quality level **X'FE'** Highest quality level **X'FF'** Printer default	O	X'02'
X'F9'	X'00'–X'01'	Constant forms control: **X'00'** Inactive **X'01'** Active	O	X'02'
X'FC'	X'01'–X'04'	N-up format control: **X'01'** 1-up format **X'02'** 2-up format **X'03'** 3-up format **X'04'** 4-up format	O	X'02'

MMC Semantics

MMCid Medium Modification Control Identifier. The identifier for the modifications specified by this structured field. This identifier is specified in a repeating group in the Medium Copy Control (MCC) structured field.

Keyword X'0E*nn*'
Retired keyword for the 3800 printer. See “Retired Parameters” on page 515 for a description.

Keyword X'90*nn*'
Specifies the high-order portion of a two-byte media destination ID. The allowed range is X'00'—X'FF'. The value X'00' is not valid if keyword X'91' also specifies a value of X'00', that is, the media destination ID X'0000' is reserved. This keyword may appear once. If this keyword is not present, the high-order portion of the media destination ID is set to X'00'. If this keyword is not present and the X'91' keyword is not present, the media destination is not specified and a presentation environment default is used.

Note: If the copy subgroup that references this MMC belongs to a duplex copy-subgroup pair, the media destination specified by this keyword must match the media destination specified for the other copy subgroup in the pair.

Keyword X'91*nn*'
Specifies the low-order portion of a two-byte media destination ID. The allowed range is X'00'—X'FF'. The value X'00' is not valid if keyword X'90' also specifies a value of X'00', that is, the media destination ID X'0000' is reserved. This keyword may appear once. If this keyword is not present, the low-order portion of the media destination ID is set to X'00'. If this keyword is not present and the X'90' keyword is not present, the media destination is not specified and a presentation environment default is used.

Note: If the copy subgroup that references this MMC belongs to a duplex copy-subgroup pair, the media destination specified by this keyword must match the media destination specified for the other copy subgroup in the pair.

Keyword X'A0*nn*'
Specifies the local ID of fixed medium information that a printer or a printer-attached device applies to a sheet-side. This application is independent of data provided through the data stream, and does not mix with the print data provided in the data stream. Fixed medium information is applied either before or after the data stream information is presented.

Value	Description
X'00'—X'FE'	Select a particular local ID for fixed medium information to be applied to the sheet-side.
X'FF'	Select all currently-supported local IDs for fixed medium information to be applied to the sheet-side.

This keyword may appear multiple times and specify multiple local IDs for fixed medium information.

Note: All Medium Modification Control structured fields that are referenced by the same Medium Copy Count structured field must specify the same local IDs for fixed medium information.

Keyword X'A100'
Specifies a perforation cut at a fixed location on the physical medium according to the current setup of the printer or printer-attached device.

Note: All Medium Modification Control structured fields that are referenced by the same Medium Copy Count structured field must specify the same perforation cuts.

Keyword X'A200'
Specifies a separation cut at a fixed location on the physical medium according to the current setup of the printer or printer-attached device.

Note: All Medium Modification Control structured fields that are referenced by the same Medium Copy Count structured field must specify the same separation cuts.

Keyword X'B4*nn*'

Specifies the high-order portion of a two-byte presentation subsystem set-up ID. The allowed range is X'00'—X'FF'. This keyword must be paired with a X'B5*nn*' keyword that immediately follows it and that specifies the low-order portion of the two-byte presentation subsystem set-up ID. The X'B4*nn*'—X'B5*nn*' keyword pair may appear multiple times. If the keyword pair is not present, a presentation subsystem set-up ID is not specified. The set-up ID specified by the X'B4*nn*' and X'B5*nn*' keywords is compared against the set-up IDs generated by the presentation subsystem, which typically consists of the presentation device and pre/post processing devices. If a match is found, presentation is allowed to proceed. If there is no match, the required set-up is not active in the presentation subsystem and presentation is terminated.

Keyword X'B5*nn*'

Specifies the low-order portion of a two-byte presentation subsystem set-up ID. The allowed range is X'00'—X'FF'. This keyword must be paired with a X'B4*nn*' keyword that immediately precedes it and that specifies the high-order portion of the two-byte presentation subsystem set-up ID. The X'B4*nn*'—X'B5*nn*' keyword pair may appear multiple times. If the keyword pair is not present, a presentation subsystem set-up ID is not specified. The set-up ID specified by the X'B4*nn*' and X'B5*nn*' keywords is compared against the set-up IDs generated by the presentation subsystem, which typically consists of the presentation device and pre/post processing devices. If a match is found, presentation is allowed to proceed. If there is no match, the required set-up is not active in the presentation subsystem and presentation is terminated.

Application Notes:

1. When presentation is terminated, the print file is put into a state where it can be resubmitted when the presentation subsystem is reconfigured to generate the required set-up IDs.
2. Presentation Subsystem set-up IDs are intended to be specified for one or more documents in a print file. IBM therefore recommends that the same IDs are specified in all the medium maps in the form map.

Note: All Medium Modification Control structured fields that are referenced by the same Medium Copy Count structured field must specify the same presentation subsystem set-up IDs.

Keyword X'D1*nn*'

Specifies whether the sheets generated by the current medium map should be offset from the sheets generated by the previous medium map or whether the edge marks applied to sheets generated by this medium map should be changed from the edge marks applied to sheets generated by the previous medium map. This keyword applies to all sheets generated by the current medium map and needs to be specified only once. If this keyword is omitted, the default is X'00' (no offset, no change in edge marks).

The keyword values are defined as follows:

Value	Description
X'00'	No offset, no change in edge marks
X'01'	Apply offset or change edge marks

Note: When processing partition ejects with N-up presentation, multiple medium maps may be invoked while building a single sheet. In that case, only the first X'D1*nn*' keyword is processed for a sheet. All other X'D1*nn*' keywords specified in medium maps invoked for the *same* sheet are ignored.

Implementation Note: Print Servers that automatically issue a jog command between jobs and between multiple copies of a job may ignore the jog (X'D1*nn*') keyword in the medium map used for the first sheet of the user's printfile.

Table 20 shows how the jog control specified by this keyword is processed with N-up presentation and conditional media ejects when an existing medium map (MM) is replaced by a new medium map. The "Result" column defines whether the sheet processed with the new medium map is jogged with respect to the previous sheet and what type of media eject (sheet or partition) occurs when the new medium map is invoked. Note that in AFP environments a jog is accomplished with the generation of an IPDS jog command when the medium map that specifies the jog is first invoked.

Table 20. Sheet Jogging and Conditional Ejects

Jog Control in Existing MM	**Jog Control in New MM**	**Eject Control in New MM**	**Result**	
			Eject	**Jog**
No jog	Jog	Partition	New sheet	Jog
No jog	Jog	New sheet	New sheet	Jog
Jog	Jog	Partition	Partition	Jog
Jog	Jog	New sheet	New sheet	Jog
Jog	No jog	Partition	New sheet	No jog
Jog	No jog	New sheet	New sheet	No jog
No jog	No jog	Partition	Partition	No jog
No jog	No jog	New sheet	New sheet	No jog

Keyword X'E0*nn*'

Specifies the format of the media source selector (X'E1') keyword.

This keyword may appear once. If this keyword is omitted, the X'E1' keyword, if present, is specified in Format 1.

The keyword values are defined as follows:

Value	Description
X'01'	The X'E1' keyword is specified in Format 1.
X'02'	The X'E1' keyword is specified in Format 2.

Keyword X'E1*nn*'

Specifies the media source. This keyword is defined in several formats. The format is selected by a X'E0' keyword or is defaulted to Format 1 if the X'E0' keyword is omitted. This keyword may appear once. If this keyword is omitted, the media source is not specified and a presentation environment default is used.

Notes:

1. If the copy subgroup that references this MMC belongs to a duplex copy-subgroup pair, the media source specified by this keyword must match the media source specified for the other copy subgroup in the pair.
2. The selected media source may be an *inserter bin*. Inserter bins do not support printing from the data stream, therefore printing is suppressed when pages, PMC overlays, and medium overlays are processed with media from an inserter bin. When a requested media source, which may be an inserter bin, is not available, the presentation systems uses a default bin and ensures that it is not an inserter bin, therefore pages and overlays that are associated with an inserter bin are printed if the inserter bin is not available.

Application Notes:

1. In AFP environments, the default media source is normally the first media source reported by the printer in the IPDS XOH-OPC reply.
2. To cause the insertion of a single sheet from the inserter bin, the application generates a data stream with one (simplex printing) or two (duplex printing) “placeholder” pages that are processed with the medium map that selects an inserter bin as the media source. If the inserter bin is available, a sheet is inserted but these pages will not be printed on the inserted sheet. However, if the inserter bin is not available, the presentation system will use a default media source that is not an inserter bin and the placeholder pages will be printed. This method can be extended to inserting multiple sheets by specifying multiple placeholder pages in the data stream.
3. An application can also cause the insertion of one or more sheets without generating placeholder pages. This is done by specifying two consecutive Invoke Medium Map (IMM) structured fields in the data stream, where the first invoked medium map selects an inserter bin and specifies the constant front (keyword X'F901') function and simplex printing, and the second invoked medium map resumes page printing from a non-inserter bin. Multiple inserted sheets can be generated in this manner by specifying a copy count that is greater than one.

X'E1*nn*' Format 1

Specifies a value that identifies either a presentation device media source ID or the characteristics associated with a presentation device media source. The keyword values in Format 1 are defined as follows:

Value	Description
X'01'	Media source ID X'00'
X'02'	Media source ID X'01'
X'03'	Media source ID X'02'
X'04'	Media source ID X'03'
X'41'	Envelope media source
X'64'	Manual feed media source

X'E1*nn*' Format 2

Specifies a value that identifies a presentation device media source ID. The keyword values in Format 2 can be in the range X'01' to X'FF' and specify media source IDs whose values are one less than the keyword values:

Value	Description
X'01'	Media source ID X'00'
X'02'	Media source ID X'01'
	⋮
X'FE'	Media source ID X'FD'
X'FF'	Media source ID X'FE'

Keyword X'E8*nn*'

Specifies the high-order portion of a two-byte local ID to select a media type. The allowed range is X'00'—X'FF'. This keyword must be paired with a X'E9*nn*' keyword that immediately follows it and that specifies the low-order portion of the two-byte media type local ID. The X'E8*nn*'–X'E9*nn*' keyword pair may appear only once. The media type local ID is mapped to a media type name or media type OID in the Map Media Type (MMT) structured field. If it is mapped to both, the media type OID takes precedence. If this keyword pair is present, it overrides the media source specified with the X'E1*nn*' keyword unless the presentation device doesn't support media type selection, in which case a specified media source is used. If the keyword pair is not present, the media is selected from the media source specified with the X'E1*nn*' keyword. A registry of standard media types along with their OID is provided in "Media Type Identifiers" on page 544.

Keyword X'E9*nn*'

Specifies the low-order portion of a two-byte local ID to select a media type. The allowed range is X'00'—X'FF'. This keyword must be paired with a X'E8*nn*' keyword that immediately precedes it and that specifies the high-order portion of the two-byte media type local ID. The X'E8*nn*'–X'E9*nn*' keyword pair may appear only once. The media type local ID is mapped to a media type name or media type OID in the Map Media Type (MMT) structured field. If it is mapped to both, the media type OID takes precedence. If this keyword pair is present, it overrides the media source specified with the X'E1*nn*' keyword unless the presentation device doesn't support media type selection, in which case a specified media source is used. If the keyword pair is not present, the media is selected from the media source specified with the X'E1*nn*' keyword.

A registry of standard media types along with their OID is provided in “Media Type Identifiers” on page 544.

Implementation Note: AFP print servers will attempt to select the media type requested by the X'E8'/X'E9' keyword pair using the following priority:

1. Attempt to find an available media source containing the media type that matches the specified OID. The media source can not be an inserter bin.
2. Attempt to find an available media source containing the media type that matches the specified name. The media source can not be an inserter bin.
3. Attempt to find an available media source whose ID matches the ID specified in a X'E1' keyword on the MMC.
4. Use the presentation process defaults for finding an available media source.

Keyword X'F1*nn*'

Retired keyword for the 3800 printer. See “Retired Parameters” on page 515 for a description.

Keyword X'F2*nn*'

Specifies the local identifier of a medium overlay that is to be applied to all sheet-sides generated by this copy subgroup. This keyword may appear a maximum of eight times in an MMC structured field. The allowed ID range is X'01'–X'7F'. The local ID must be mapped to the external name of the medium overlay in a Map Medium Overlay (MMO) structured field.

Keyword X'F3*nn*'

Specifies the local identifier of a text suppression that is to be applied to all sheet-sides generated by this copy subgroup. This keyword may appear a maximum of eight times in an MMC structured field. The allowed ID range is X'01'–X'7F'.

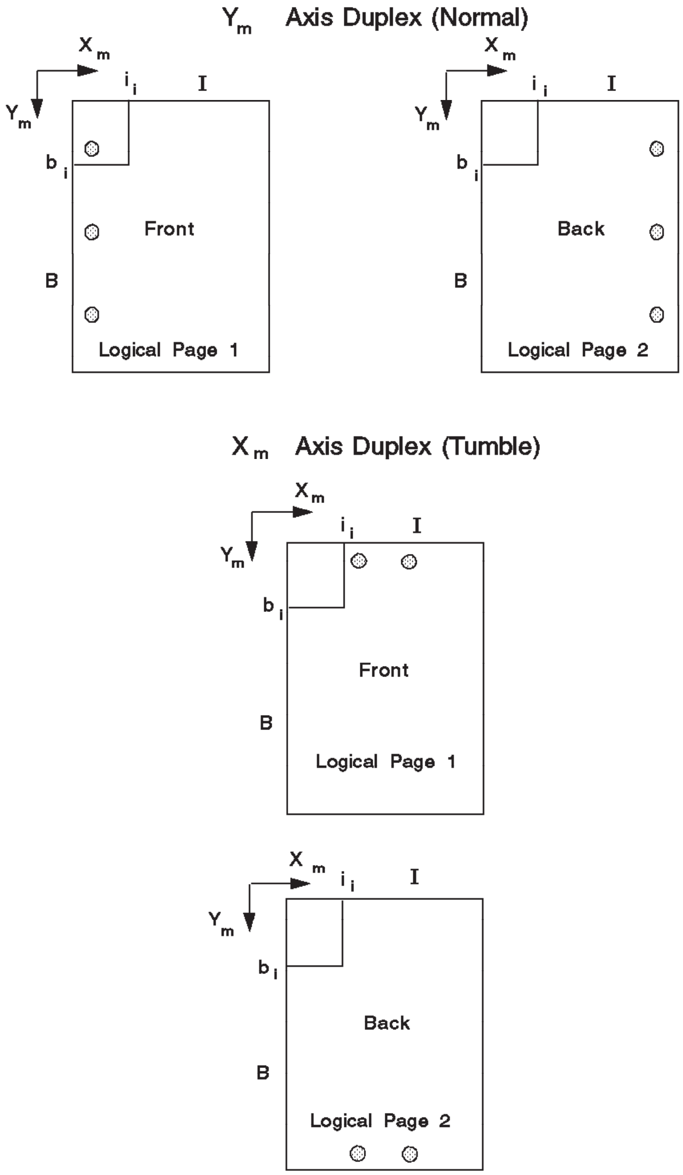
Keyword X'F4*nn*'

Specifies whether data is generated on the front side of the sheet (simplex) or on both sides of the sheet (duplex). If duplex is specified, the first copy subgroup in a pair generates the front sheet-side, and the second copy subgroup in the pair generates the back sheet-side. If this keyword is omitted, the default is X'01' (simplex).

The keyword values are defined as follows:

Value	Description
X'01'	Simplex
X'02'	Normal duplex. The media is turned around the Y_m axis.
X'03'	Tumble duplex. The media is turned around the X_m axis.

See Figure 57 on page 256 for a description of normal duplex and tumble duplex.



Note: The shaded circles in the illustration represent holes punched through the sheets of the two examples.

Figure 57. Normal Duplex and Tumble Duplex Printing

Note: All Medium Modification Control structured fields that are referenced by the same Medium Copy Count structured field must specify the same value for this keyword.

Keyword X'F8*nn*'

Specifies the level of print quality to be used on all sheet-sides generated by this copy subgroup. The mapping of print quality levels to physical print quality is presentation device dependent.

The allowed quality level range is X'01'–X'FF', and is defined as follows:

Value	**Description**
X'01'	Lowest print quality level
X'FE'	Highest print quality level

X'FF'	Device default print quality

Keyword X'F9*nn*'

Specifies whether both variable page data and medium overlay data or only medium overlay data should be generated on all sheet-sides generated by this copy subgroup. This functions is known as *constant forms control*. Note that PMC overlays are considered variable page data for this keyword. If this keyword is omitted, the default is X'00' (present both medium overlay data and variable page data).

The keyword values are defined as follows:

Value	Description
X'00'	Present both medium overlay data and variable page data
X'01'	Present only medium overlay data. If no medium overlays are specified for this copy subgroup, no data is presented on the sheet-sides generated by this copy subgroup.

Note: All Medium Modification Control structured fields that are referenced by the same Medium Copy Count structured field must specify the same value for this keyword.

Keyword X'FC*nn*'

Specifies the number of pages to be placed on a physical medium using N-up partitioning. In N-up partitioning, each side of the physical medium is divided into a number of equal-size partitions, where the number of partitions is indicated by the number N in “N-up”. If duplex is specified, the same N-up partitioning is applied to the back side as is applied to the front side. With simplex N-up partitioning, N pages are placed on the physical medium, and with duplex N-up partitioning, 2N pages are placed on the physical medium. Pages placed into partitions may be blank pages generated by setting PgFlgs bit 0 = B'1' in the Page Position (PGP) structured field repeating group.

Pages are placed into partitions using either a *default N-up page placement* or an *explicit N-up page placement*, as specified in the Page Position (PGP) structured field. In default N-up page placement, consecutive pages in the data stream are placed into consecutively-numbered partitions. In explicit N-up page placement, consecutive pages in the data stream are processed using consecutive PGP repeating groups and are placed into explicitly-specified partitions. For more information on page placement, see “Page Position (PGP) Format 2” on page 282.

Pages may be rotated within their partitions so that the page presentation space X axis is at a 0°, 90°, 180°, or 270° orientation with respect to the medium presentation space X axis. This rotation is specified in the Page Position structured field.

Pages are positioned within their partition relative to the partition origin using the offsets specified in the Page Position structured field. Modifications may be applied to pages before they are placed in their partition using the Page Modification Control (PMC) structured field. Figure 20 on page 61 shows the partitioning for wide continuous-forms media, narrow continuous-forms media, and cut-sheet media. Partitioning is not used with envelope media.

Figure 58 on page 290 through Figure 69 on page 296 show partition numbering for various media. This keyword may appear once.

The keyword values are defined as follows:

Value **Description**

X'01' 1-up partitioning. The medium presentation space is divided into one partition. One page (simplex) or two pages (duplex) are presented on the physical medium.

X'02' 2-up partitioning. The medium presentation space is divided into two partitions. Two pages (simplex) or four pages (duplex) are presented on the physical medium.

X'03' 3-up partitioning. The medium presentation space is divided into three partitions. Three pages (simplex) or six pages (duplex) are presented on the physical medium.

X'04' 4-up partitioning. The medium presentation space is divided into four partitions. Four pages (simplex) or eight pages (duplex) are presented on the physical medium.

Note: All Medium Modification Control structured fields that are referenced by the same Medium Modification Control structured field must specify the same value for this keyword.

Application Note: IPDS printers require that pages be contained within their partition if default N-up page placement is specified, otherwise an exception is generated. This restriction does not exist if explicit N-up page placement is specified. That is, pages may overflow their partition without necessarily causing an exception.

MMC Exception Condition Summary

- A X’02’ exception condition exists when an undefined keyword is encountered in an MMC structured field.

Map Medium Overlay (MMO)

The Map Medium Overlay structured field maps one-byte medium overlay local identifiers that are specified by keywords in the Medium Modification Control (MMC) structured field to external medium overlay names.

MMO (X'D3B1DF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B1DF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	RGLength	X'0C'	Length of each repeating group	M	X'06'
1–3				Reserved; must be zero	M	X'06'
Zero to 127 repeating groups in the following format:						
0	UBIN	OVLid	X'01'–X'7F'	Medium overlay local identifier	M	X'06'
1	BITS	Flags			M	X'06'
Bit 0			B'0'–B'1'	Raster indicator; retired for 3800		
Bits 1–7			B'0000000'	Reserved; must be zero		
2–3				Reserved; must be zero	M	X'06'
4–11	CHAR	OVLname		External name of medium overlay	M	X'06'

MMO Semantics

RGLength Length of each repeating group. Set to 12.

PsegName Medium overlay local identifier as specified by a keyword in an MMC structured field. The allowed range is X'01'–X'7F' and must be unique to each repeating group.

Flags

Bit	**Description**
0	Retired parameter for the 3800 printer. See “Retired Parameters” on page 515 for a description.
1–7	Reserved; must be zero.

OVLname External name of the medium overlay.

Map Media Type (MMT)

The Map Media Type structured field maps a Resource Local ID to the name or identifier of a media type.

MMT (X'D3AB88') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AB88'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One or more repeating groups in the following format:						
0–1	UBIN	RGLength	14–(*n*+1)	Total length of this repeating group	M	X'06'
8–*n*		Triplets		See “MMT Semantics” for triplet applicability.	M	X'14'

MMT Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Media Type structured field repeating groups as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once in each repeating group. May occur twice in each repeating group if one occurrence uses FQNFmt X'00' (name), and the other occurrence uses FQNFmt X'10' (OID). See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is **X'11'**—*Media Type Reference*. The media type reference may be specified in one of two ways: • If FQNFmt = X'00', the reference is made with a character-encoded name. **Architecture Note:** In the IPDS architecture, the media type name must be encoded using IBM code page 500, character set 640 (plus space character). It is strongly recommended that the same encoding be used in the FQN type X'11' triplet when FQNFmt = X'00', since not all print servers are able to process other encodings. Note that when the OID format is used to identify the media type, it is specified in hexadecimal format as defined in "Media Type Identifiers" on page 544. • If FQNFmt = X'10', the reference is made with a ASN.1 OID encoded using the definite short form. A registry of standard media types along with their OID is provided in "Media Type Identifiers" on page 544. If the FQN type X'11' triplet is specified twice in a repeating group, the FQNFmt X'10'—OID reference, takes precedence.
X'22'	Extended Resource Local Identifier	Mandatory. Must occur once in each repeating group. See "Extended Resource Local Identifier Triplet X'22'" on page 348. The only Extended Resource Local Identifier type that may appear is **X'40'**—*Media Type resource*. **Architecture Note:** The local IDs used with resource type X'40' are specified with a X'EB*nn* + X'E9*nn*' keyword pair on the MMC that can only carry a 2-byte ID. Therefore, the range for this resource type is restricted to 2-byte values.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.

Within the same medium map, you may not map the same Resource Local ID to more than one media type or a X'01' exception condition exists. The media type may be specified with an FQN type X'11' triplet using FQNFmt X'10' (OID reference), an FQN type X'11' triplet using FQNFmt X'00' (name reference), or both. Within the same medium map, different Resource Local IDs may be mapped to the same media type.

Implementation Note: AFP print servers will attempt to select the requested media type using the following priority:

1. Attempt to find an available media source containing the media type that matches the specified OID. The media source can not be an inserter bin.

2. Attempt to find an available media source containing the media type that matches the specified name. The media source can not be an inserter bin.
3. Attempt to find an available media source whose ID matches the ID specified in a X'E1' keyword on the MMC.
4. Use the presentation process defaults for finding an available media source.

MMT Exception Condition Summary

- A X′02′ exception condition exists when:
  - A Fully Qualified Name (X′02′) triplet other than a type X′11′ (Media Type Reference) appears within any repeating group.
  - An Extended Resource Local Identifier (X′22′) triplet type other than X'40' appears within any repeating group.
- A X′01′ exception condition exists when the same LID is mapped to more than one media type within the same structured field.

Map Page (MPG)

The Map Page structured field identifies a page that is to be merged with data specified for the current page by using an Include Page (IPG) structured field.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

MPG (X'D3ABAF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABAF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One repeating group in the following format:						
0–1	UBIN	RGLength	12–(*n*+1)	Total length of this repeating group	M	X'06'
2–*n*		Triplets		See “MPG Semantics” for triplet applicability.	M	X'14'

MPG Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Page structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Must occur once in each repeating group. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'83'**—*Begin Document Name*. Specifies the name of the document that contains the page to be included with the IPG.
X'02'	Fully Qualified Name	Mandatory. Must occur once in each repeating group. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'87'**—*Begin Document Name*. Specifies the name of the document that contains the page to be included with the IPG.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.

Triplet	Type	Usage
X'5A'	Object Offset	Optional. May occur once, with *ObjTpe*=X'AF', to specify that pages are the objects to be counted for the offset. Specifies how many pages in the referenced document precede the page to be mapped. The page offset is measured from the beginning of the referenced document, so that the first page has offset 0, the second page has offset 1, and the nth page has offset (*n*−1). When this triplet is specified, the page name, as specified by the Fully Qualified Name type X'87' triplet, is ignored. See "Object Offset Triplet X'5A'" on page 378.

MPG Exception Condition Summary

- A X′02′ exception condition exists when a Fully Qualified Name (X′02′) triplet other than a type X′87′ (Begin Page Reference) or a type X′83′ (Begin Document Reference) appears within the repeating group.
- A X′01′ exception condition exists when:
  - Multiple type X′87′ (Begin Page Reference) Fully Qualified Name triplets appear within the repeating group.
  - Multiple type X′83′ (Begin Document Reference) Fully Qualified Name triplets appear within the repeating group.

Map Page Overlay (MPO)

The Map Page Overlay structured field maps a Resource Local ID to the name of a Begin Overlay structured field. A Map Page Overlay structured field may contain from one to 254 repeating groups.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

MPO (X'D3ABD8') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABD8'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One to 254 repeating groups in the following format:						
0–1	UBIN	RGLength	11–(*n*)+1	Total length of this repeating group	M	X'06'
2–*n*		Triplets		See “MPO Semantics” for triplet applicability.	M	X'14'

MPO Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

Triplets Appear in the Map Page Overlay structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once in each repeating group. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is **X'84'**—*Begin Resource Object Reference* which must match the name on a Begin Overlay structured field or a X’01’ exception condition exists.
X'24'	Resource Local Identifier	Mandatory. Must occur once in each repeating group. See “Resource Local Identifier Triplet X'24'” on page 350. The only Resource Local Identifier type that may appear is **X'02'**—*Page Overlay*.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'46'	Page Overlay Conditional Processing	Optional. May occur more than once. See “Page Overlay Conditional Processing Triplet X'46'” on page 361.

Triplet	Type	Usage
X'47'	Resource Usage Attribute	Optional. May occur once. See “Resource Usage Attribute Triplet X'47'” on page 363.

Within the same Map Page Overlay structured field, you may not map the same Resource Local ID to more than one page overlay resource or a X′01′ exception condition exists. However, you may use two or more repeating groups within the same Map Page Overlay structured field to map different LIDs to the same page overlay resource.

MPO Exception Condition Summary

- A X′02′ exception condition exists when:
  - A Fully Qualified Name (X′02′) triplet other than a type X'84' (Begin Resource Object Reference) appears within any repeating group.
  - A Resource Local Identifier (X′24′) triplet type other than X'02' appears within any repeating group.
- A X′01′ exception condition exists when:
  - A Begin Overlay structured field with the same name as that specified on the FQN type X'84' triplet cannot be located.
  - Multiple FQN type X'84' triplets appear within the same repeating group.
  - Multiple type X'02' Resource Local Identifier (X′24′) triplets appear within the same repeating group.
  - Multiple Resource Usage Attribute (X′47′) triplets appear within the same repeating group.
  - The same LID is mapped to more than one page overlay within the same structured field.

Map Page Segment (MPS)

The Map Page Segment structured field identifies page segments that are required to present a page on a physical medium.

Application Note: To optimize print performance, it is strongly recommended that the same encoding scheme be used for a resource reference wherever in a print file that resource reference is specified. That is, the encoding scheme used for the resource include, the resource map, and the resource wrapper should be the same.

MPS (X'D3B15F') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B15F'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	RGLength	X'0C'	Length of each repeating group	M	X'06'
1–3				Reserved; must be zero	M	X'06'
Zero to 127 repeating groups in the following format:						
0–3				Reserved; must be zero	M	X'06'
4–11	CHAR	PsegName		External name of page segment	M	X'06'

MPS Semantics

RGLength Length of each repeating group. Set to 12.

PsegName External name of the page segment.

Application Note: A page segment included on a page or overlay with an IPS may optionally be mapped with an MPS in the AEG for that page or overlay. If such a mapping exists, the page segment is sent to the presentation device as a separate object and is called a *hard* page segment. If such a mapping does not exist, the page segment data is sent to the presentation device as part of the page or overlay and is called a *soft* page segment.

Map Suppression (MSU)

The Map Suppression structured field maps one-byte text suppression local identifiers to external text suppression names. Suppressible text is identified in presentation text objects with a local identifier and is bracketed with control sequences that specify the beginning and the end of the suppression. A text suppression is activated by specifying its local identifier in a Medium Modification Control (MMC) structured field in a medium map.

MSU (X'D3ABEA') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ABEA'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
Zero to 127 repeating groups in the following format:						
0–7	CHAR	SUPname		External name of text suppression	M	X'06'
8				Reserved; must be zero	M	X'06'
9	CODE	SUPid	X'01'–X'7F'	Text suppression local identifier	M	X'06'

MSU Semantics

SUPname External name of the text suppression.

SUPid Text suppression local identifier, as specified by a keyword in an MMC structured field. The allowed range is X'01'—X'7F'.

Note: The local ID may be mapped to more than one external name.

Architecture Notes:

1. When processing AFP line data with Pagedefs, the Descriptor structured fields can enable the text suppression function for a record, and, if so, assign an eight-byte external name to the suppression function. This external name is mapped to a local identifier using the MSU structured field. For more information on line data and Pagedefs, see the *Advanced Function Presentation: Programming Guide and Line Data Reference*.

No Operation (NOP)

The No Operation structured field performs no function.

NOP (X'D3EEEE') Syntax

Structured Field Introducer				Structured Field Data
SF Length (2B)	ID = **X'D3EEEE'**	Flags (1B)	Reserved; X'0000'	

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	UndfData		Up to 32,759 bytes of data with no architectural definition	O	X'00'

NOP Semantics

UndfData Is data that has no architectural definition.

The No Operation structured field may be specified within any begin-end domain.

Note: The No Operation structured field may be used to carry comments or any other type of unarchitected data. Although this is not recommended, it may also be used to carry semantic data in private or exchange data streams. However, because receivers of interchange data streams should ignore the content of No Operation structured fields, and because receiver-generator products are not required to propagate No Operation structured fields, no semantics should be attached to the data carried by the No Operation structured field in interchange data streams.

Object Area Descriptor (OBD)

The Object Area Descriptor structured field specifies the size and attributes of an object area presentation space.

OBD (X'D3A66B') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A66B'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–19		Triplets		See “OBD Semantics” for triplet applicability.	M	X'14'

OBD Semantics

Triplets Appear in the Object Area Descriptor structured field as follows:

Triplet	Type	Usage
X'43'	Descriptor Position	Mandatory. Must occur once. See “Descriptor Position Triplet X'43'” on page 355.
X'4B'	Measurement Units	Mandatory. Must occur once. See “Measurement Units Triplet X'4B'” on page 364.
X'4C'	Object Area Size	Mandatory. Must occur once. See “Object Area Size Triplet X'4C'” on page 365.
X'4E'	Color Specification	Optional. May occur once. Specifies a color for the object area. The color specification defines a color space, the syntax for specifying color values in the color space, and the actual color value. When this triplet is specified on an object area, the complete object area becomes foreground data that is colored with the specified color *before* any object data is added to the area. If the default mixing rules are used, the object area, once it becomes foreground data, overpaints (covers) any data that is underneath. **Note:** This triplet is not permitted on the OBD for presentation text that may optionally occur in the AEG for a page or overlay.
X'70'	Presentation Space Reset Mixing	Optional. May occur once. If this triplet specifies a reset to the color of medium (BgMxFlag=B'1'), the reset takes place at the point in the data stream where the triplet occurs. This triplet may not appear in the Object Area Descriptor structured field with a Presentation Space Mixing Rules triplet. See “Presentation Space Reset Mixing Triplet X'70'” on page 390.

Triplet	Type	Usage
X'71'	Presentation Space Mixing Rules	Optional. May occur once. This triplet may not appear in the Object Area Descriptor structured field with a Presentation Space Reset Mixing triplet. See "Presentation Space Mixing Rules Triplet X'71'" on page 392. **Implementation Note:** The Presentation Space Mixing Rules (X'71') triplet is currently not used in AFP environments.

Architecture Note: Triplets that affect the object area presentation space are processed in the order in which they occur on the OBD. For example, if a Presentation Space Reset Mixing (X'70') triplet on the OBD is followed by a Color Specification (X'4E') triplet, the object area is colored with the color specified in the X'4E' triplet and covers any data underneath it regardless of whether the X'70' triplet specified "reset to color of medium" or "do not reset to color of medium". If a Color Specification (X'4E') triplet is followed by a X'70' triplet, and if the X'70' triplet specified "reset to color of medium", the object area is colored with color of medium. If the X'70' triplet specified "do not reset to color of medium", the X'70' triplet does not change the object area and it remains foreground data colored with the color specified by the X'4E' triplet.

OBD Exception Condition Summary

- A X'01' exception condition exists when the OBD structured field contains both a Presentation Space Reset Mixing triplet and a Presentation Space Mixing Rules triplet.

Object Area Position (OBP)

The Object Area Position structured field specifies the origin and orientation of the object area, and the origin and orientation of the object content within the object area.

OBP (X'D3AC6B') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AC6B'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	OAPosID	X'01'–X'7F'	The object area position identifier	M	X'06'
One repeating group in the following format:						
1	UBIN	RGLength	23	Total length of this repeating group	M	X'06'
2–4	SBIN	XoaOset	-32768–32767	X-axis origin of the object area	M	X'06'
5–7	SBIN	YoaOset	-32768–32767	Y-axis origin of the object area	M	X'06'
8–9	CODE	XoaOrent	X'0000', X'2D00', X'5A00', 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'06'
10-11	CODE	YoaOrent	X'0000', X'2D00', X'5A00', 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'06'
Note: See "OBP Semantics" on page 273 for valid combinations of the XoaOrent and YoaOrent values.						
12				Reserved; must be binary zero	M	X'06'
13–15	SBIN	XocaOset	-32768–32767	X-axis origin for object content	M	X'06'
16–18	SBIN	YocaOset	-32768–32767	Y-axis origin for object content	M	X'06'
19–20	CODE	XocaOrent	X'0000'	The object content's X-axis rotation from the X axis of the object area coordinate system	M	X'06'
21–22	CODE	YocaOrent	X'2D00'	The object content's Y-axis rotation from the X axis of the object area coordinate system	M	X'06'

Offset	Type	Name	Range	Meaning	M/O	Exc
23	CODE	RefCSys	X'00', X'01', X'05'	Reference coordinate system: **X'00'** Page or overlay coordinate system; origin is defined by IPS structured field **X'01'** Page or overlay coordinate system; standard origin **X'05'** Retired value	M	X'06'

OBP Semantics

OAPosID Specifies an identifier for this Object Area Position structured field that is unique within the environment group. It is used to associate the Object Area Position structured field with the Object Area Descriptor structured field.

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

XoaOset Specifies the offset along the X axis, X_{pg} or X_{ol}, of the referenced coordinate system to the origin of the X axis, X_{oa}, for the object area coordinate system. The value for this parameter is expressed in terms of the number of referenced coordinate system X-axis measurement units. The reference coordinate system is described below under *RefCSys*.

YoaOset Specifies the offset along the Y axis, Y_{pg} or Y_{ol}, of the referenced coordinate system to the origin of the Y axis, Y_{oa}, for the object area coordinate system. The value for this parameter is expressed in terms of the number of referenced coordinate system Y-axis measurement units. The reference coordinate system is described below under *RefCSys*.

XoaOrent 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 reference coordinate system.

YoaOrent 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 reference coordinate system. The YoaOrent value must be 90 degrees greater than the XoaOrent value or a X'01' exception condition exists.

Notes:

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

Table 21. OBP: Valid values for XoaOrent and YoaOrent

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

2. If the object area orientation is such that the sum of the object area origin offset and the object area extent exceeds the size of the including presentation space in either the X or Y direction, all of the object area will not fit on the including presentation space. The including presentation space in this case is the page or overlay presentation space. If an attempt is made to actually present data in the portion of the object area that falls outside the including presentation space, that portion of the data is not presented, and a X’01’ exception condition exists.

XocaOset Specifies the offset along the X axis of the object area coordinate system, X_{oa}, to the X origin of the object content. The value for this parameter is expressed in terms of the number of object area coordinate system X-axis measurement units.

YocaOset Specifies the offset along the Y axis of the object area coordinate system, Y_{oa}, to the Y origin of the object content. The value for this parameter is expressed in terms of the number of object area coordinate system Y-axis measurement units.

Notes:

1. The object content is developed in the *data object presentation space*; within the context of this structured field the two terms are synonymous.
2. The XocaOset and YocaOset parameters are used only when a *position* or *position and trim* mapping is specified to map the object content to the object area. They are ignored for all other mappings.

XocaOrent Specifies the amount of rotation of the object content’s X axis about its defined origin relative to the X axis of the object area coordinate system.

YocaOrent Specifies the amount of rotation of the object content’s Y axis about its defined origin relative to the X axis of the object area coordinate system.

Note: If the object content orientation is such that the object content origin offset exceeds the size of the object area presentation space in either the X or Y direction, the object data will not fit on the object area presentation space. If the mapping option is position, that is X'00', and an attempt is made to actually present data outside the object area presentation space, that portion of the data is not presented, and a X’01’ exception condition exists.

RefCSys Specifies the coordinate system and origin used to position the object area.

Value	Description
X'00'	Used only if the object is part of a page segment. The reference coordinate system is the including page or overlay coordinate system. Object areas are positioned in this coordinate system with respect to a point (X_p, Y_p) or (X_{ol}, Y_{ol}) that is defined by the Include Page Segment (IPS) structured field.
X'01'	The reference coordinate system is the including page or overlay coordinate system. Object areas are positioned in

this coordinate system with respect to the standard origin defined by (X_p=0, Y_p=0) or (X_{ol}=0, Y_{ol}=0).

X'05' Retired value. See “Retired Parameters” on page 515.

All others
Reserved

OBP Exception Condition Summary

- A X’01’ exception condition exists when:
  - The value specified for YoaOrent is not 90 degrees greater rotation than the value specified for XoaOrent.
  - An attempt is made to present data outside the presentation space of the containing coordinate system.
  - The mapping option is position and an attempt is made to present data outside the object area presentation space.

Object Container Data (OCD)

The Object Container Data structured field contains the data for an object carried in an object container.

OCD (X'D3EE92') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3EE92'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	ObjCdat		Up to 32,759 bytes of object data	O	X'00'

OCD Semantics

ObjCdat Contains the object data.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Presentation Fidelity Control (PFC)

The Presentation Fidelity Control structured field specifies the user fidelity requirements for data presented on physical media and for operations performed on physical media. The scope of the Presentation Fidelity Control structured field is the document or print file controlled by the form map that contains this structured field.

PFC (X'D3B288') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B288'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0				Reserved; must be zero	M	X'06'
1	BITS	PFCFlgs		Flags	M	X'06'
Bit 0			B'0', B'1'	**B'0'** Reset fidelity controls to defaults and apply PFC controls **B'1'** Do not reset fidelity controls to defaults before applying PFC controls		
Bits 1–7			B'0000000'	Reserved; must be zero		
2–3				Reserved; must be zero	M	X'06'
4–*n*		Triplets		See "PFC Semantics" for triplet applicability.	O	X'10'

PFC Semantics

Triplets are used on the Presentation Fidelity Control structured field to define specific presentation fidelity requirements that are to be applied by the presentation process as data is presented on physical media. While triplets may be conceptually related, each triplet is processed independently of any other triplet. Therefore, it is the responsibility of the generator of the Presentation Fidelity Control structured field to ensure cross-triplet consistency. If a particular fidelity triplet is not specified on this structured field, or if this structured field is not specified, presentation process defaults are used to control the presentation fidelity.

PFCFlgs The following flags are defined:

Bit **Description**

0 Fidelity Control Activation

B'0' Reset all fidelity controls to their presentation process defaults, then apply fidelity controls specified by this PFC structured field

B'1' Leave all fidelity controls at their current setting, and additionally apply fidelity controls specified by this PFC structured field. If there is a conflict between an existing fidelity control and a new fidelity control, the last-specified fidelity control takes precedence.

1–7 Reserved; all bits must be B'0'.

Triplets Appear in the Presentation Fidelity Control structured field as follows:

Triplet	Type	Usage
X'74'	Toner Saver	May occur once. Used to activate and deactivate a toner saver mode for printing. See page 397.
X'75'	Color Fidelity	Optional. May occur once. Specifies the actions to be taken by the presentation process when a color exception is detected while processing the data stream. See page 399.
X'78'	Font Fidelity	May occur once. Specifies the actions to be taken by the presentation process when a font resolution exception is detected while processing the data stream. See page 401.
X'86'	Text Fidelity	Optional. May occur once. Specifies the actions to be taken by the presentation process when a text exception is detected while processing the data stream. See page 414.
X'87'	Media Fidelity	Optional. May occur once. Specifies the actions to be taken by the presentation process when a request for a specific media or a specific media bin cannot be satisfied. See page 416.
X'88'	Finishing Fidelity	Optional. May occur once. Specifies the actions to be taken by the presentation process when a finishing exception is detected while processing the data stream. See page 418.

Application Note: Some presentation platforms allow presentation fidelity parameters to be specified in the print request. For example, in the MVS environment, invalid character exceptions and positioning exceptions may be blocked with a data check parameter in the JCL. In the OS/400 environment, a print fidelity indicator may be used to specify whether absolute fidelity is required, so that the presentation process can determine how to continue following exceptions such as font not available, duplexing not available, media source not available, and data stream function not available. Print request fidelity specifications are outside the scope of the MO:DCA architecture. It is up to the print requestor to ensure that fidelity specifications in the form map are consistent and compatible with fidelity specifications in the print request. If there is a clear conflict between the fidelity specification in the form map and the fidelity specification in the print request, the presentation process may terminate processing of the print job.

Page Descriptor (PGD)

The Page Descriptor structured field specifies the size and attributes of a page or overlay presentation space.

PGD (X'D3A6AF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A6AF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	XpgBase	X'00'–X'01'	Page unit base for the X axis: **X'00'** 10 inches **X'01'** 10 centimeters	M	X'06'
1	CODE	YpgBase	X'00'–X'01'	Page unit base for the Y axis: **X'00'** 10 inches **X'01'** 10 centimeters	M	X'06'
2–3	UBIN	XpgUnits	1–32767	Page units per unit base for the X axis	M	X'06'
4–5	UBIN	YpgUnits	1–32767	Page units per unit base for the Y axis	M	X'06'
6–8	UBIN	XpgSize	1–32767	Page extent for the X axis	M	X'06'
9–11	UBIN	YpgSize	1–32767	Page extent for the Y axis	M	X'06'
12–14				Reserved; must be binary zero	M	X'06'
15–17		Triplets		See “PGD Semantics” for triplet applicability.	O	X'10'

PGD Semantics

XpgBase Specifies the unit base for the X axis of the page or overlay coordinate system.

YpgBase Specifies the unit base for the Y axis of the page or overlay coordinate system.

Note: A X’01’ exception condition exists if the XpgBase and YpgBase values are not identical.

XpgUnits Specifies the number of units per unit base for the X axis of the page or overlay coordinate system.

YpgUnits Specifies the number of units per unit base for the Y axis of the page or overlay coordinate system.

Application Note: Some AFP print servers require that the measurement units in the PGD match the measurement units in the Presentation Text Descriptor (PTD) when the latter is included in the AEG for a page. It is therefore strongly recommended that whenever the PTD is included in the AEG, the same measurement units are specified in both the PTD and PGD.

XpgSize Specifies the extent of the X axis of the page or overlay coordinate system. This is also known as the page or overlay's X-axis size.

YpgSize Specifies the extent of the Y axis of the page or overlay coordinate system. This is also known as the page or overlay's Y-axis size.

Note: If the sum of the page or overlay origin offset and the page or overlay extent exceeds the size of the including presentation space in either the X or Y direction, all of the page or overlay will not fit on the including presentation space. The including presentation space in this case is the medium presentation space. If an attempt is made to actually present data in the portion of the page or overlay that falls outside the including presentation space, that portion of the data is not presented, and a X'01' exception condition exists.

Application Note: The IS/1 and IS/2 interchange set definitions limit the page size to 22.75 inches in the X and Y directions. To specify a larger page size, 240 units per inch should be specified in the PGD for the page measurement units. Using a range of 1 to 32767, this allows a maximum page size in the X and Y directions of 136.5 inches, is supported by all IPDS printers, and keeps the complete page presentation space within the range of 2-byte addressing parameters in the IPDS and PTOCA architectures.

Triplets Appear in the Page Descriptor structured field as follows:

Triplet	Type	Usage
X'4E'	Color Specification	Optional. May occur once. Specifies a color for the page or overlay presentation space. The color specification defines a color space, the syntax for specifying color values in the color space, and the actual color value. When this triplet is specified on a page or overlay presentation space, the complete presentation space becomes foreground data that is colored with the specified color *before* any object data is added to the presentation space. If the default mixing rules are used, the page or overlay presentation space, when it becomes foreground data, overpaints (covers) any data that is underneath. See "Color Specification Triplet X'4E'" on page 367.
X'70'	Presentation Space Reset Mixing	If this triplet specifies a reset to the color of medium (BgMxFlag=B'1'), the reset takes place at the point in the data stream where the triplet occurs. This triplet may not appear in the Page Descriptor structured field with a Presentation Space Mixing Rules triplet. See "Presentation Space Reset Mixing Triplet X'70'" on page 390.

Triplet	Type	Usage
X'71'	Presentation Space Mixing Rules	May occur once. This triplet may not appear in the Page Descriptor structured field with a Presentation Space Reset Mixing triplet. See "Presentation Space Mixing Rules Triplet X'71'" on page 392. **Implementation Note:** The Presentation Space Mixing Rules (X'71') triplet is currently not used in AFP environments.

Architecture Note: Triplets that affect the page or overlay presentation space are processed in the order in which they occur on the PGD. For example, if a Presentation Space Reset Mixing (X'70') triplet on the PGD is followed by a Color Specification (X'4E') triplet, the presentation space is colored with the color specified in the X'4E' triplet and covers any data underneath it regardless of whether the X'70' triplet specified "reset to color of medium" or "do not reset to color of medium". If a Color Specification (X'4E') triplet is followed by a X'70' triplet, and if the X'70' triplet specified "reset to color of medium", the presentation space is colored with color of medium. If the X'70' triplet specified "do not reset to color of medium", the X'70' triplet does not change the presentation space and it remains foreground data colored with the color specified by the X'4E' triplet.

PGD Exception Condition Summary

- A X'01' exception condition exists when:
  - The XpgBase and YpgBase values are not identical.
  - An attempt is made to present data outside the medium presentation space. See the note under *YpgSize* for details.
  - The PGD structured field contains both a Presentation Space Reset Mixing triplet and a Presentation Space Mixing Rules triplet.

Page Position (PGP) Format 2

The Page Position structured field specifies the position and orientation of a page's presentation space on the medium presentation space for the physical medium. The PGP may be located in a medium map or in the document environment group of a form map. When present in the active medium map, it overrides a PGP in the document environment group of the form map. If N-up partitioning is specified by the Medium Modification Control structured field in the active medium map, the medium presentation spaces on the front and back sides of a sheet are divided into N partitions; and the Page Position structured field specifies the partition into which each page is mapped and with respect to which the page presentation space is positioned and oriented. The N-up page-to-partition mapping can be specified in two mutually exclusive ways:

- Default N-up page placement. Pages are processed in the order in which they appear in the data stream and are placed into consecutively-numbered partitions, that is, the first page is placed into partition 1, the second page is placed into partition 2, the third page is placed into partition 3, and the 4th page is placed into partition 4. Partition numbering for various media is shown in Figure 58 on page 290 to Figure 69 on page 296.
- Explicit N-up page placement. Pages are processed in the order in which they appear in the data stream and are placed into the partition that is explicitly specified by the repeating group for the page. Multiple pages may be placed into the same partition. If N-up simplex is specified, the Page Position structured field *must* contain N repeating groups, one for each page on the sheet-side. If N-up duplex is specified, the Page Position structured field *must* contain 2N repeating groups, one for each page on the sheet.

PGP (X'D3B1AF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B1AF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	Constant	X'01'	Reserved constant; must be X'01'	M	X'06'
One repeating group in the following format:						
0	UBIN	RGLength	X'0A'–X'0C'	Length of each repeating group	M	X'06'
1–3	SBIN	X_mOset	-32768–32767	X_m coordinate of page presentation space origin	M	X'06'
4–6	SBIN	Y_mOset	-32768–32767	Y_m coordinate of page presentation space origin	M	X'06'

Offset	Type	Name	Range	Meaning	M/O	Exc
7–8	CODE	PGorient	X'0000', X'2D00', X'5A00', X'8700'	The page presentation space X-axis rotation from the X axis of the medium presentation space: **X'0000'** 0° rotation **X'2D00'** 90° rotation **X'5A00'** 180° rotation **X'8700'** 270° rotation	M	X'06'
9	CODE	SHside	X'00'–X'01', X'10'–X'11', X'20'–X'21', X'30'–X'31', X'40'–X'41'	Sheet side and partition selection **X'00'** Page on front side if no N-up, default page placement on front side if N-up **X'01'** Page on back side if no N-up, default page placement on back side if N-up **X'10'** Explicit N-up page placement: partition 1, front side **X'11'** Explicit N-up page placement: partition 1, back side **X'20'** Explicit N-up page placement: partition 2, front side **X'21'** Explicit N-up page placement: partition 2, back side **X'30'** Explicit N-up page placement: partition 3, front side **X'31'** Explicit N-up page placement: partition 3, back side **X'40'** Explicit N-up page placement: partition 4, front side **X'41'** Explicit N-up page placement: partition 4, back side	M	X'06'
10	BITS	PgFlgs		Specify additional presentation controls for the partition. See "PGP Semantics" for PgFlgs bit definitions.	O	X'02'
11	CODE	PMCid	0–127	Page Modification Control identifier	O	X'02'
			X'FF'	Apply all modifications		

PGP Semantics

The Page Position structured field contains repeating groups that are used to map pages to the medium presentation space or to partitions on the medium presentation space. The number of repeating groups that may appear on the Page Position structured field is determined as follows:

- If N-up is not specified by the Medium Modification Control structured field in the active medium map, the Page Position structured field contains one repeating group for the front sheet-side for simplex printing, and two repeating groups, one for the front sheet-side and one for the back sheet-side for duplex printing. Each repeating group specifies the offset, orientation, and optional modifications for the page that is to be presented on the sheet-side. The page offset is measured with respect to the medium presentation space origin, and the page orientation is measured with respect to the medium presentation space X axis. Pages are processed sequentially as they appear in the data stream. For duplex printing, the front sheet-side is always processed before the back sheet-side, regardless of the order of the two repeating groups.
- If N-up is specified by the Medium Modification Control structured field in the active medium map and the default N-up page placement is desired, the Page Position structured field contains one repeating group for the front sheet-side for simplex printing, and two repeating groups, one for the front sheet-side and one for the back sheet-side for duplex printing. Each repeating group must specify default N-up page placement, and the specified page offset, page orientation, and page modifications apply to all pages placed on the sheet-side. The page offset is measured with respect to the origin of the partition into which the page is placed, and the page orientation is measured with respect to the medium presentation space X axis. Pages are processed sequentially as they appear in the data stream. For duplex printing, the front sheet-side is always processed before the back sheet-side, regardless of the order of the two repeating groups.
- If N-up is specified by the Medium Modification Control structured field in the active medium map and if explicit N-up page placement is desired, the Page Position structured field contains N repeating groups for simplex printing, and 2N repeating groups for duplex printing. Pages are processed sequentially as they appear in the data stream using consecutive PGP repeating groups. The first page is processed using the first repeating group, the second page is processed using the second repeating group, and so on. Each repeating group must specify a sheet-side, a partition number in the range from 1 to N, a page offset, and a page orientation. Each repeating group may also specify optional modifications to be applied to the page. Multiple repeating groups may specify the same partition number. The page offset is measured with respect to the origin of the partition specified by the repeating group. The page orientation is measured with respect to the medium presentation space X axis.

Notes:

1. The processing of PGP repeating groups is driven by pages in the data stream. If page n is the last page in a document, the repeating group used to present page n is the last repeating group that is processed. Similarly, if page *n* is followed by an IMM, the repeating group used to present page n is the last repeating group processed before the new medium map is invoked. As a result, if a PGP repeating group is to present a PMC overlay without any page data, placing it before the last repeating group that presents page data will ensure that this repeating group is processed and the PMC overlay is presented.
2. Pages can be placed in the partitions that correspond to default page placement but still be individually offset, oriented, and modified by specifying explicit page placement and sequential partition numbers in the repeating groups. For example, for 2-up duplex, the first repeating group specifies SHside = X'10', the second repeating group specifies SHside = X'20', the third repeating group specifies SHside = X'11', and the fourth repeating group specifies SHside = X'21'.

RGLength Length of each repeating group. Set to 10, 11, or 12.

X_mOset Offset of the page's presentation space origin along the X_m axis of the medium presentation space using the measurement units specified in the Medium Descriptor structured field. If N-up partitioning is specified by the Medium Modification Control structured field in the active medium map, the offset is measured from the partition origin.

Y_mOset Offset of the page's presentation space origin along the Y_m axis of the medium presentation space using the measurement units specified in the Medium Descriptor structured field. If N-up partitioning is specified by the Medium Modification Control structured field in the active medium map, the offset is measured from the partition origin.

PGorient Specifies the amount of clockwise rotation of the page presentation space X axis, X_p, about the page presentation space origin, relative to the X_m axis of the medium presentation space. The rotation of the Y axis of the page presentation space is always 90° greater than the rotation of the X axis. The allowed rotations are:

Value	Description
X'0000'	0° rotation
X'2D00'	90° rotation
X'5A00'	180° rotation
X'8700'	270° rotation

Note: If the page rotation is such that the sum of the page origin offset and the page extent exceeds the size of the including medium presentation space in either the X_m or Y_m direction, all of the page presentation space will not fit on the medium presentation space. If an attempt is made to actually present data in the portion of the page presentation space that falls outside the medium presentation space, that portion of the data is not presented, and a X'01' exception condition exists.

SHside Specifies the sheet side to which the repeating group applies and the manner in which pages are placed on the sheet side. If N-up partitioning is specified by the Medium Modification Control structured field in the active medium map, this parameter specifies the N-up page placement. It may specify the default N-up page placement, where pages are placed into consecutive partitions, or it may specify explicit N-up page placement, where pages are placed into explicitly-specified partitions.

Value	Description
X'00'	Single page placed on front sheet-side if no N-up specified, default page placement on front sheet-side if N-up specified.
X'01'	Single page placed on back sheet-side if no N-up specified, default page placement on back sheet-side if N-up specified.

Note: If default N-up page placement is specified for the front sheet-side, it must also be specified for the back sheet-side. With default N-up page placement, one repeating group (simplex) or two repeating groups (duplex) are specified, and the specified offset and orientation apply to all pages mapped to the sheet-side.

X'10' Explicit N-up page placement; page is mapped to partition 1, front sheet-side.

X'11' Explicit N-up page placement; page is mapped to partition 1, back sheet-side.

X'20' Explicit N-up page placement; page is mapped to partition 2, front sheet-side.

X'21' Explicit N-up page placement; page is mapped to partition 2, back sheet-side.

X'30' Explicit N-up page placement; page is mapped to partition 3, front sheet-side.

X'31' Explicit N-up page placement; page is mapped to partition 3, back sheet-side.

X'40' Explicit N-up page placement; page is mapped to partition 4, front sheet-side.

X'41' Explicit N-up page placement; page is mapped to partition 4, back sheet-side.

Application Note: IPDS printers require that pages be contained within their partition if default N-up page placement is specified, otherwise an exception is generated. This restriction does not exist if explicit N-up page placement is specified, that is, pages may overflow their partition without necessarily causing an exception.

PgFlgs Specify additional presentation controls for the partition. Bits 0–2 of this parameter are used only if N-up is specified by the Medium Modification Control structured field in the active medium map. If N-up is not specified and this parameter is present, bits 0–2 are ignored, and the architected default for PgFlgs bits 0–2 is B'000' (present variable page data, present PMC overlays, position PMC overlays with respect to the page origin).

Bit **Description**

0 Variable page data:

B'0' Present variable page data in the partition

B'1' Do not present variable page data in the partition. This causes a blank page to be presented in the partition.

1 PMC overlays:

B'0' Present PMC overlays in partition

B'1' Do not present PMC overlays in partition

2 PMC overlay position:

B'0' The offset specified for PMC overlays is measured with respect to the page origin using the measurement units specified in the PMC structured field. If no measurement units are specified in the PMC, the measurement units specified in the MDD structured field are used.

B'1' The offset specified for PMC overlays is measured with respect to the partition origin using the measurement units specified in the PMC structured field. If no measurement units are specified in the PMC, the measurement units specified in the MDD structured field are used. The measurement of the PMC overlay offset is done with the page in the 0° rotation. This fixes the position of the overlay origin with respect to the page origin along the X_{pg} and Y_{pg} axes, or along extensions of the X_{pg} and Y_{pg} axes in the *negative* direction. If a non-zero degree page rotation is specified, each PMC overlay is positioned by rotating the page coordinate system, extending the X_{pg} and Y_{pg} axes in the negative direction, and placing the PMC overlay origin in the extended (X_{pg},Y_{pg}) coordinate system at the same position, relative to the page, that it occupied in the 0° page rotation.

3 Page view control:

B'0' The data presented by this repeating group is intended for viewing. This is the architected default if the PgFlgs parameter is not specified.

B'1' The data presented by this repeating group is not intended for viewing.

4–7 Reserved; all bits must be B'0'.

Notes:

1. If this optional parameter is omitted, the PMCid parameter must be omitted as well and the architected default for PgFlgs bits 0–3 is B'0000', that is, present variable page data in the partition, present all PMC overlays in the active medium map in the partition, position PMC overlays with respect to the page origin, and view the data presented by this repeating group.
2. PMC overlays are page overlays whether they are positioned with respect to the page origin or the partition origin. PMC overlays rotate with the page if a non-zero page rotation is specified by the PGorient parameter. Media-level controls, such as the Constant Forms Control X'F9' keyword in the MMC, treat PMC overlays as variable page data.
3. The functions enabled at the page-level by bits 0–1 of this parameter are analogous to the functions provided by the Constant Forms Control (X'F9') keyword and the Medium Overlay Local ID (X'F2') keyword in the MMC at the medium-level. When the PgFlgs parameter, the X'F9' keyword, and the X'F2' keyword are present, they interact as follows:
   - The Constant Forms Control (X'F9') keyword is not supported with N-up explicit page placement and is ignored if it occurs. Similar functionality can be achieved for a sheet side by explicitly including the medium overlay as a PMC overlay on a partition without any variable page data.

When N-up with default page placement is specified, this keyword controls the application of variable page data that may include PMC overlays to a sheet side, while the PgFlgs parameter controls the application of variable page data and PMC overlays to a partition.

When the X'F9' keyword specifies that no variable page data is to be applied to the sheet side, it overrides the page-level specification in the PgFlgs parameter for that sheet side. The resulting effect is the same as if the PGP repeating group for partitions on that sheet side specified bits 0,1 = B'11' (do not present variable page data in the partitions and do not present PMC overlays in the partitions). In that case, the medium overlay is applied to the sheet side but neither variable page data nor PMC overlays are applied to any partition on the sheet side.

When the X'F9' keyword specifies that variable page data including PMC overlays can be applied to the sheet side, the PgFlgs parameter determines whether variable page data and PMC overlay data is placed into partitions on that sheet side.

- With default N-up page placement, if a sheet-side contains only constant data (MMC Constant Forms Control X'F9' keyword is specified or PGP PgFlgs bit 0 = B'1'), it is built as long as:
  - At least a single page is placed anywhere on that sheet; or
  - The other sheet-side also contains only constant data.
- The Medium Overlay Local ID (X'F2') keyword controls the application of medium overlays to the sheet side, while the PgFlgs parameter controls the application of PMC overlays to the page in a partition. These two overlay types are included or omitted *independently*.

  Note that medium overlays are only guaranteed to be presented on a sheet side if a page, which could be a blank page generated by setting PgFlgs bit 0 = B'1', is also presented on the sheet side, or if the Constant Forms Control (X'F9') keyword specifies X'01' (present only medium overlay data) for that sheet side.

  For example, if the PGP specifies explicit page placement but does not contain a repeating group for a back-side partition, and if the MMC for the back side copy subgroup calls out a medium overlay with the X'F2' keyword, this medium overlay will not be presented.
- In general, if the Constant Forms Control (X'F9') keyword is not specified for a sheet-side, any medium overlays specified for that sheet-side are only presented if at least a single page is placed on *the same* sheet-side. Note that this page could be a page with variable data, a blank page with only PMC overlays, or even a blank page without PMC overlays, as determined by the setting of the PgFlgs parameter.

Application Note: Bits 0–1 of the PgFlgs parameter can be used to place a blank page into a partition or to fill a partition with constant data specified in a PMC overlay.

PMCid Identifies a Page Modification Control (PMC) structured field in the active medium map that specifies modifications to be applied to the page before it is placed in the partition. If this parameter is not specified on a repeating group, or if the parameter specifies X'FF', all modifications specified by all PMCs in the active medium map are applied to the page. If this parameter is specified on a repeating group, only the modifications included by the selected PMC are applied to the page. If the medium map does not contain a PMC with the specified ID, no PMC modifications are applied. This parameter is used only if N-up is specified by the Medium Modification Control structured field in the active medium map. If N-up is not specified and this parameter is present, it is ignored, and all modifications specified by all PMCs in the active medium map are applied to the page.

Notes:

1. If the PMCid parameter is included in a repeating group, the optional PgFlgs positional parameter is mandatory for that repeating group.
2. All overlays included with a PMC structured field are presented on the page presentation space *before* any variable page data is presented.

Application Note: The N-up function provided by the PGP structured field provides powerful and flexible functionality for placing multiple pages on a single sheet. Not all of this functionality maps easily to a viewing environment, which is normally page-based. When creating N-up applications that are to be both printed and viewed, you should follow these guidelines:

- Do not use medium overlays. Medium overlays are tied to a sheet-side, not to a page, and should be replaced with PMC overlays, which can be tied to a page. If medium overlays are used, the page and PMC overlay position and rotation with respect to the medium origin must be preserved. This may generate blank space on the display screen and may even cause the page and PMC overlays to position or rotate off the screen. To avoid these problems, some viewing applications may not support medium overlays when presenting N-up data.
- Generate the PGP so that all data that must be displayed with a particular page is referenced by the PGP repeating group that is used to process the page.
- Avoid creating special effects by overlapping two or more pages since these effects will not be displayed by a page-based N-up viewing system.
- Avoid splitting page content across more than one page, since this would require a multi-page viewing capability.

PGP Exception Condition Summary

- A X’01’ exception condition exists when:
  - One repeating group specifies default N-up page placement and another repeating group specifies explicit N-up page placement.
  - The Page Position structured field contains an invalid number of repeating groups for the given N-up and simplex/duplex specification.

– Explicit N-up page placement is specified, but the active medium map does not specify N-up partitioning.

– A repeating group specifies invalid data, such as a back sheet-side partition when the active medium map specifies simplex, or partition #3 when the active medium map specifies 2-up.

Partition Numbering for N-up

Partition numbering for various media is shown in Figure 58 to Figure 69 on page 296. The numbering depends on whether 1-up, 2-up, 3-up, or 4-up is specified, and on how the medium presentation space is oriented on the physical medium. The medium presentation space orientation is specified by the Medium Orientation (X'68') triplet on the Medium Descriptor structured field to be Portrait (X'00'), Landscape (X'01'), Reverse Portrait (X'02'), Reverse Landscape (X'03'), Portrait 90 (X'04'), or Landscape 90 (X'05'). Note that when duplexing, the location of the partitions on the back sheet-side *relative* to the location of the partitions on the front sheet-side is dependent on whether normal duplexing (turning the media around the Y_m axis) or tumble duplexing (turning the media around the X_m axis) is specified.

Legend: The small circles in Figure 58 to Figure 69 on page 296 represent holes punched through the sheets and are intended to show how the sheets were flipped from front-side to back-side. All sheets have three holes punched along one of the long sides and one hole punched along the other long side. The small square indicates the medium origin, and the arrow indicates the direction of the medium X_m axis.

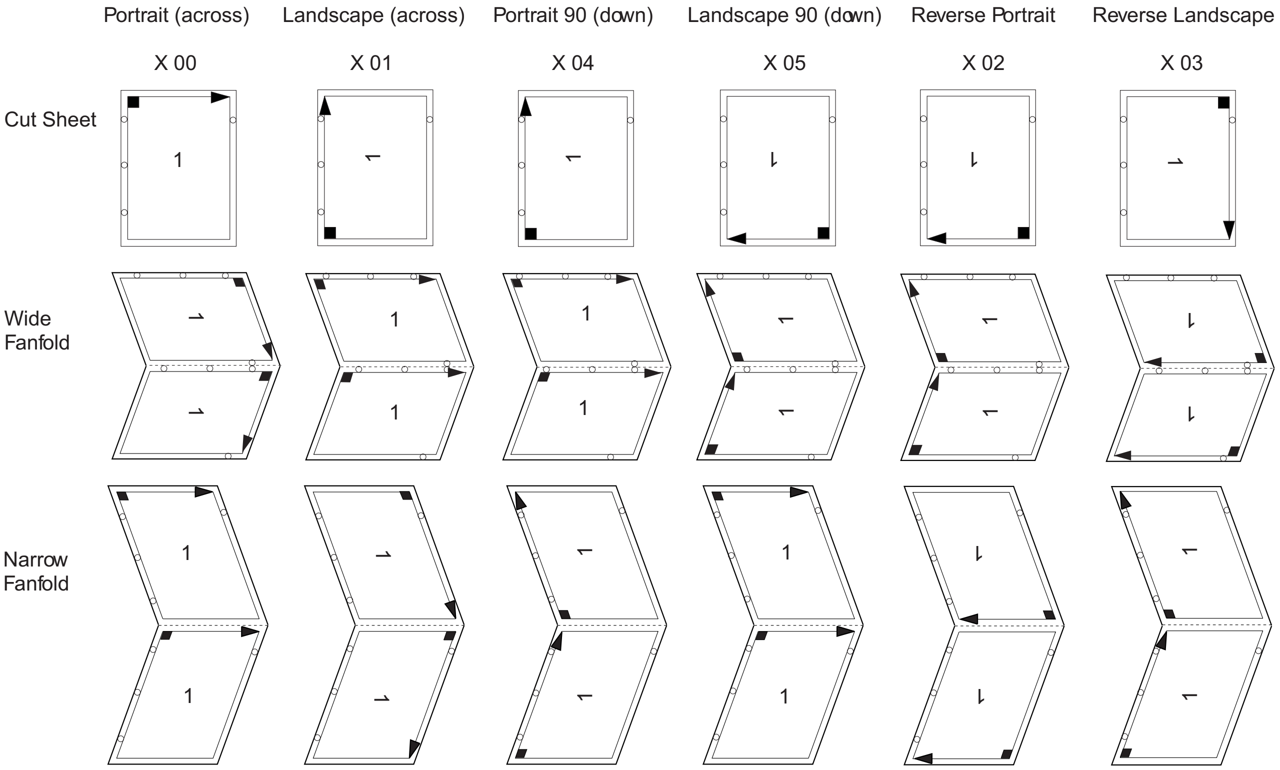


Figure 58. 1-up Partition Numbering, Front Sheet-Side

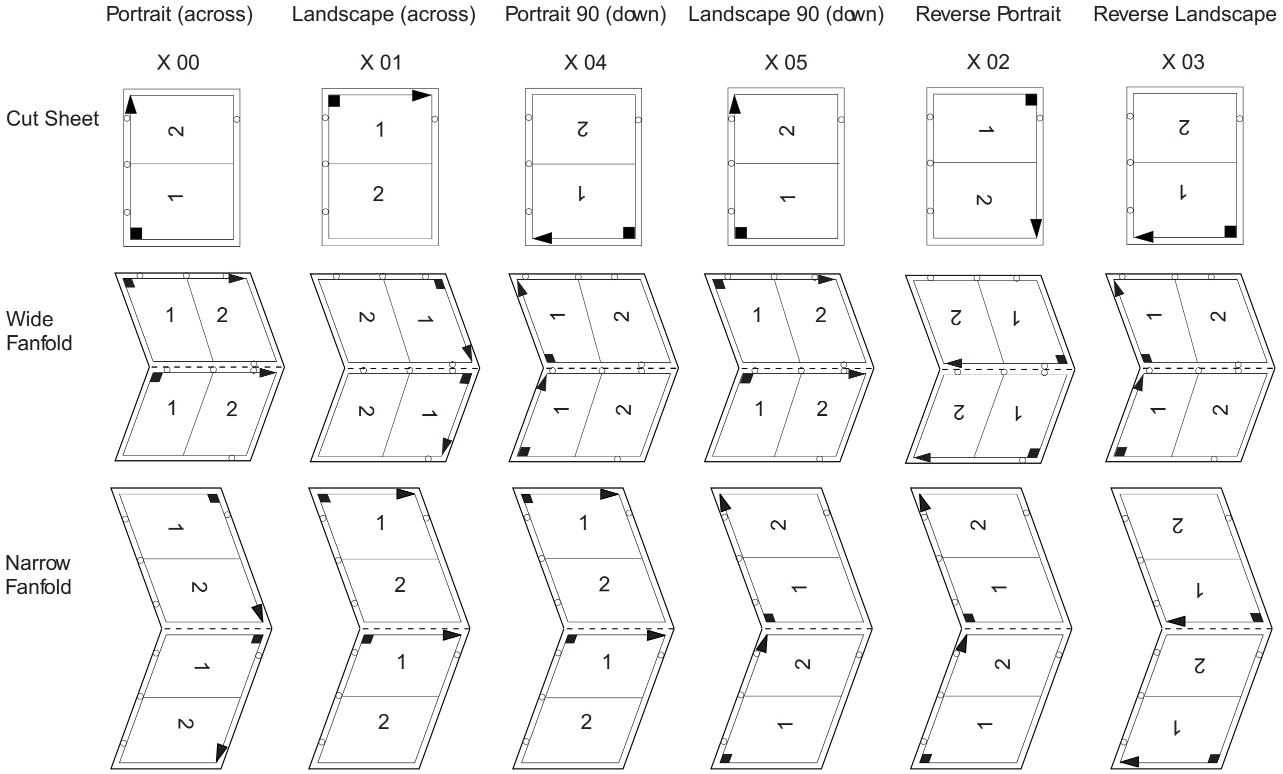


Figure 59. 2-up Partition Numbering, Front Sheet-Side

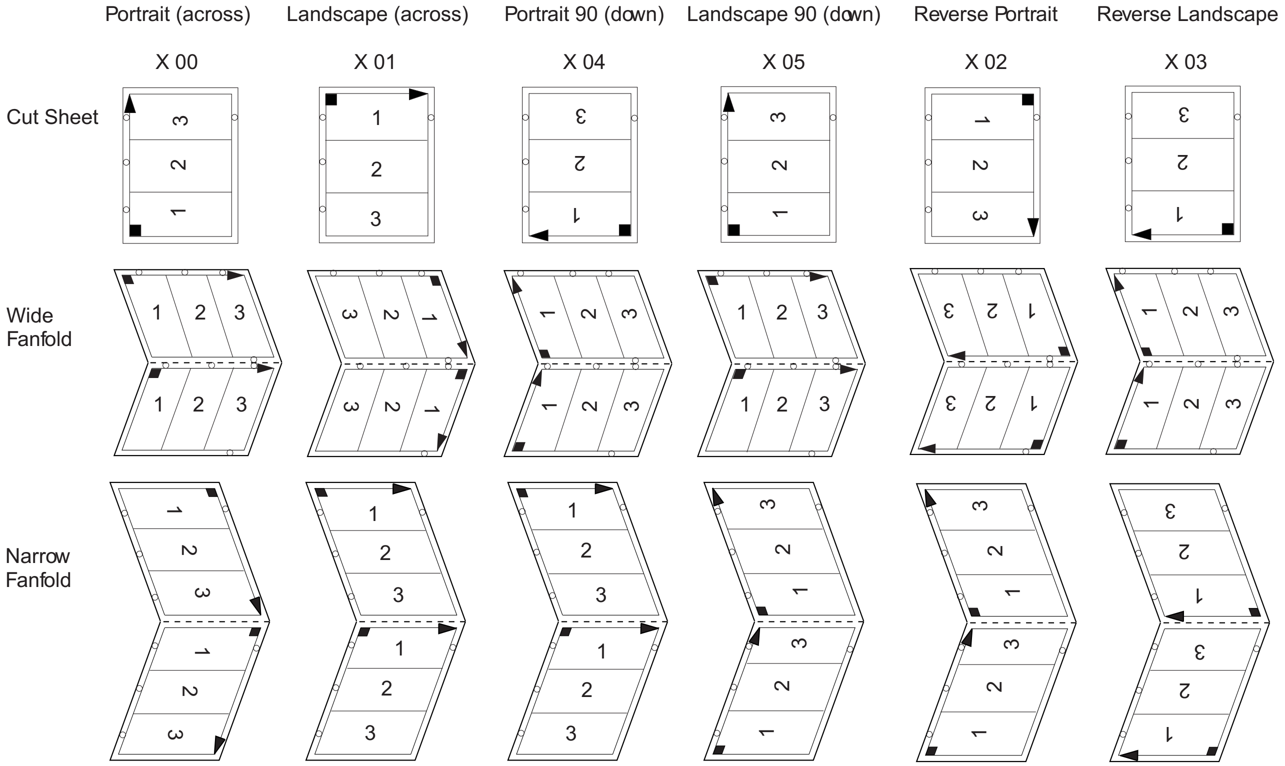


Figure 60. 3-up Partition Numbering, Front Sheet-Side

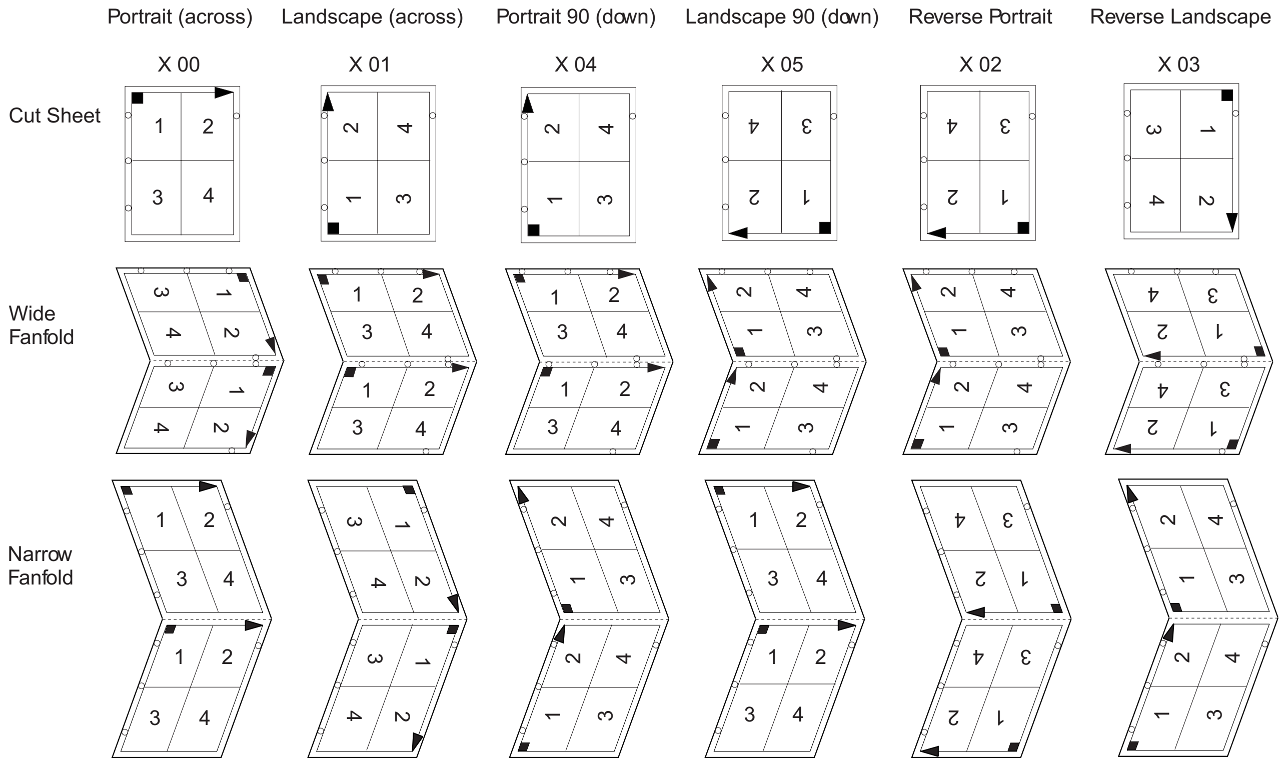


Figure 61. 4-up Partition Numbering, Front Sheet-Side

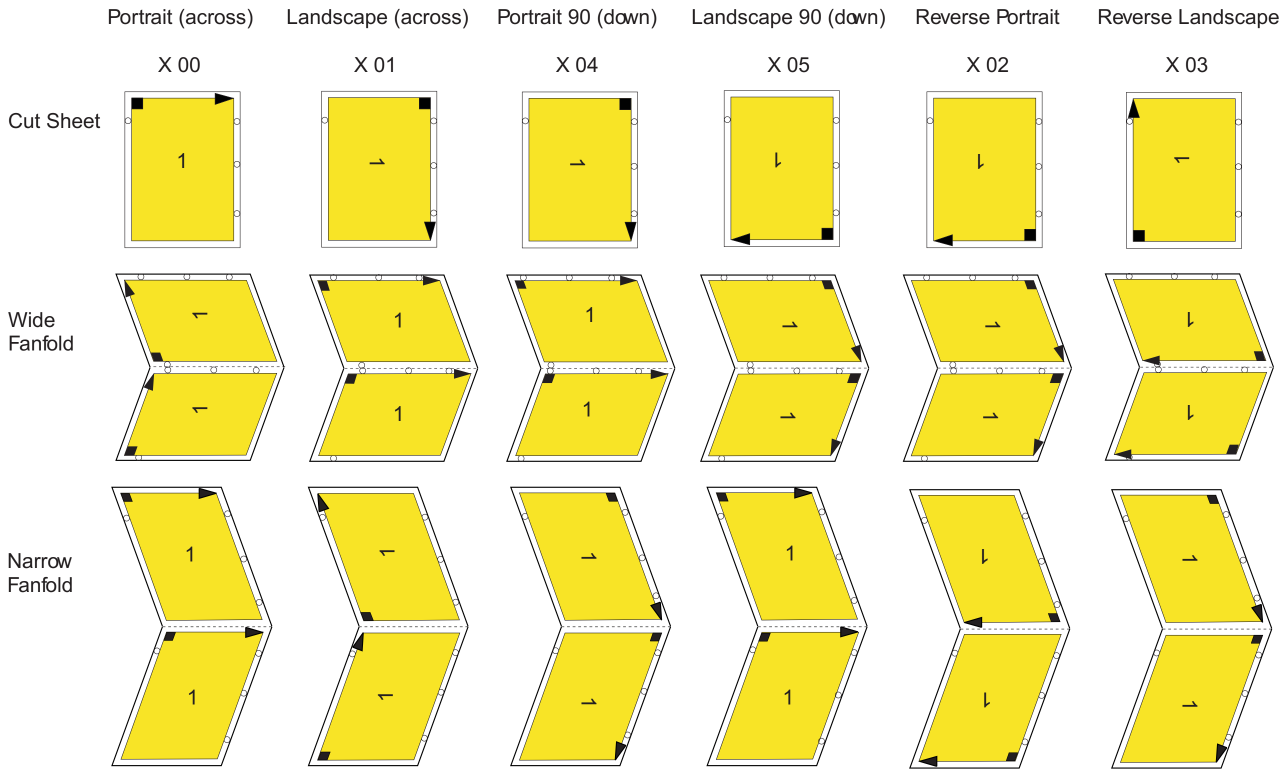


Figure 62. 1-up Partition Numbering, Back Sheet-Side, Normal Duplex

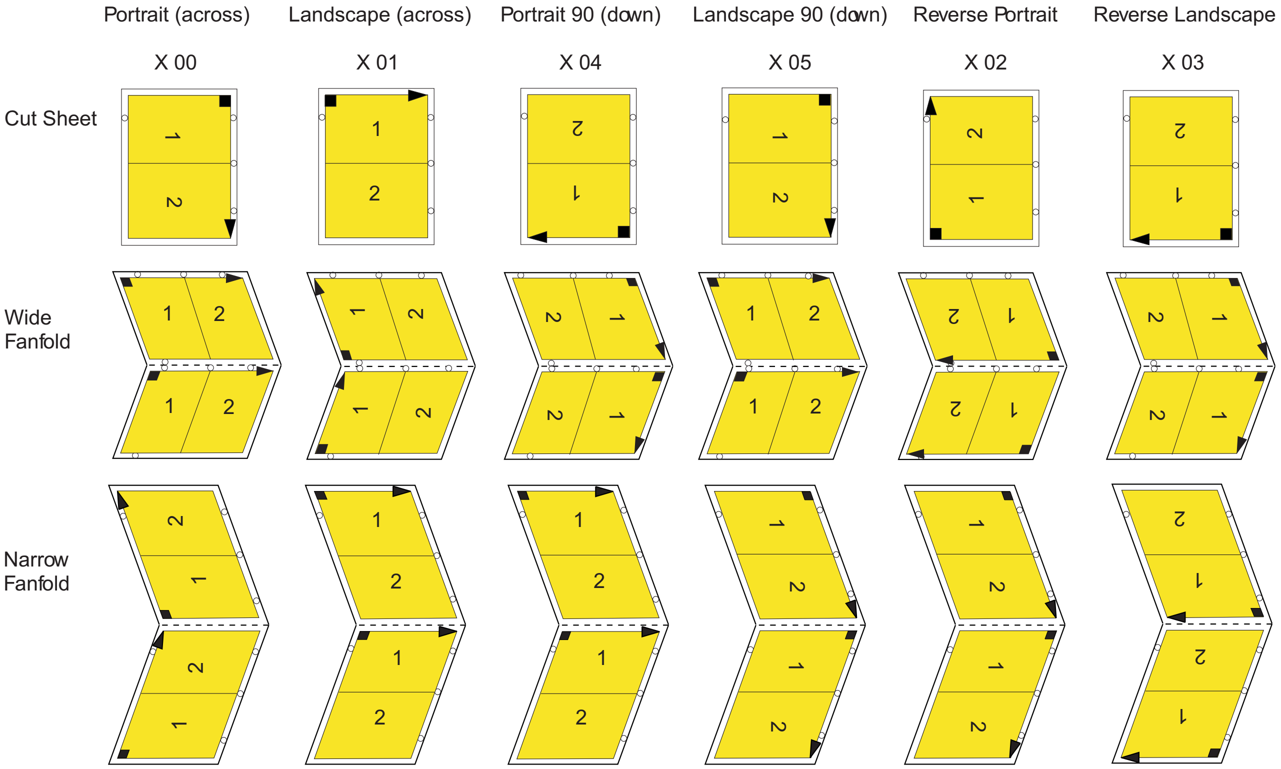


Figure 63. 2-up Partition Numbering, Back Sheet-Side, Normal Duplex

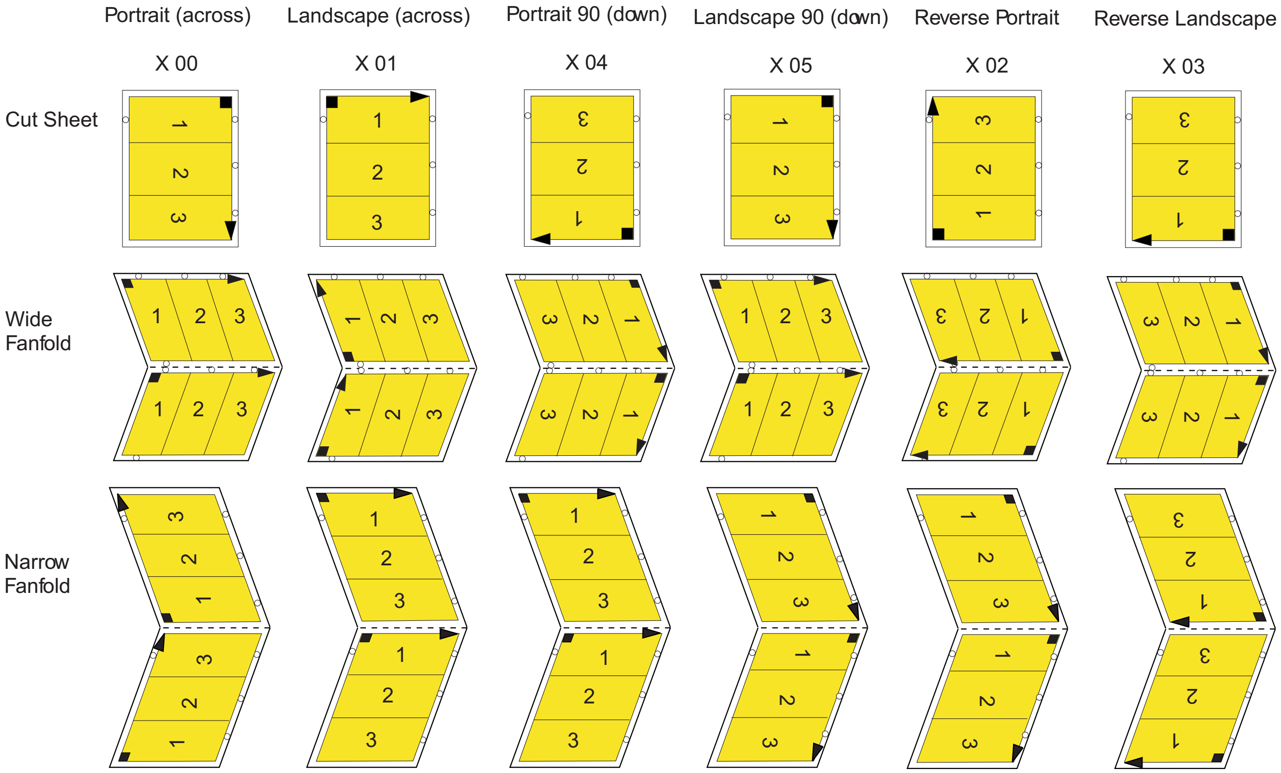


Figure 64. 3-up Partition Numbering, Back Sheet-Side, Normal Duplex

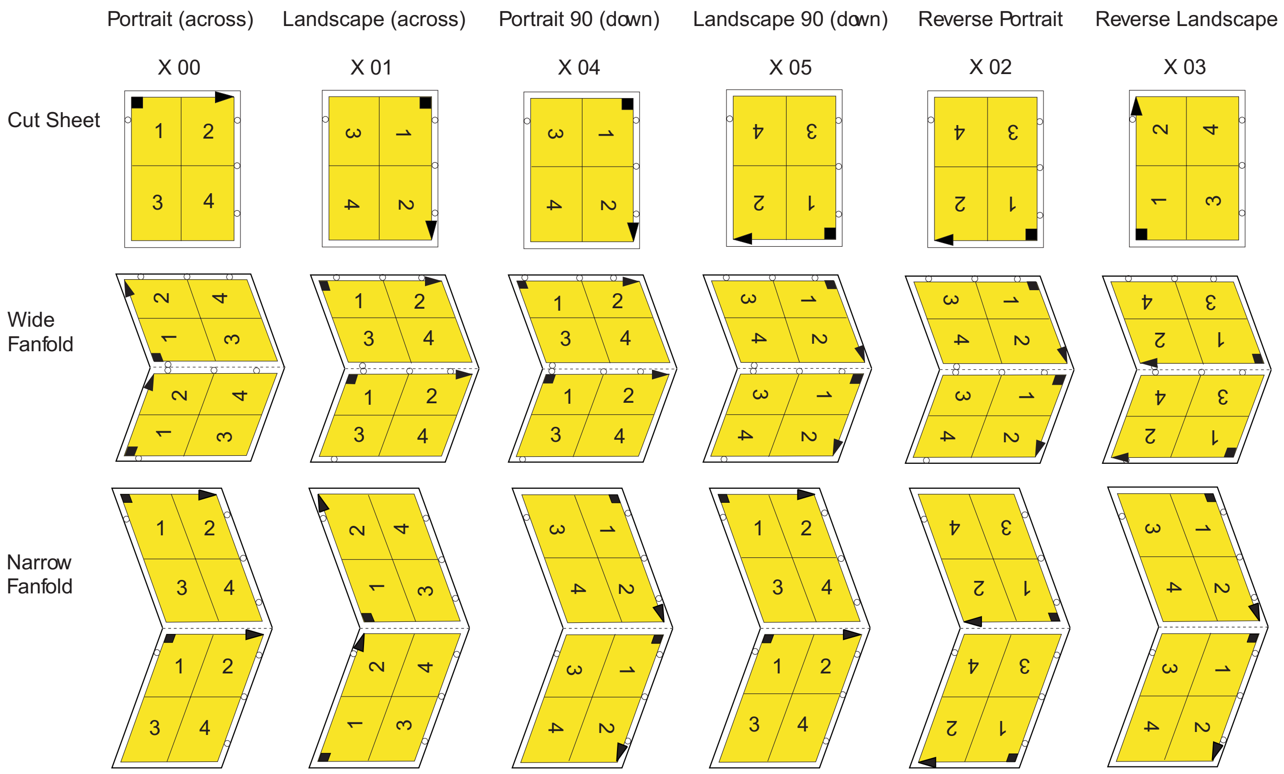


Figure 65. 4-up Partition Numbering, Back Sheet-Side, Normal Duplex

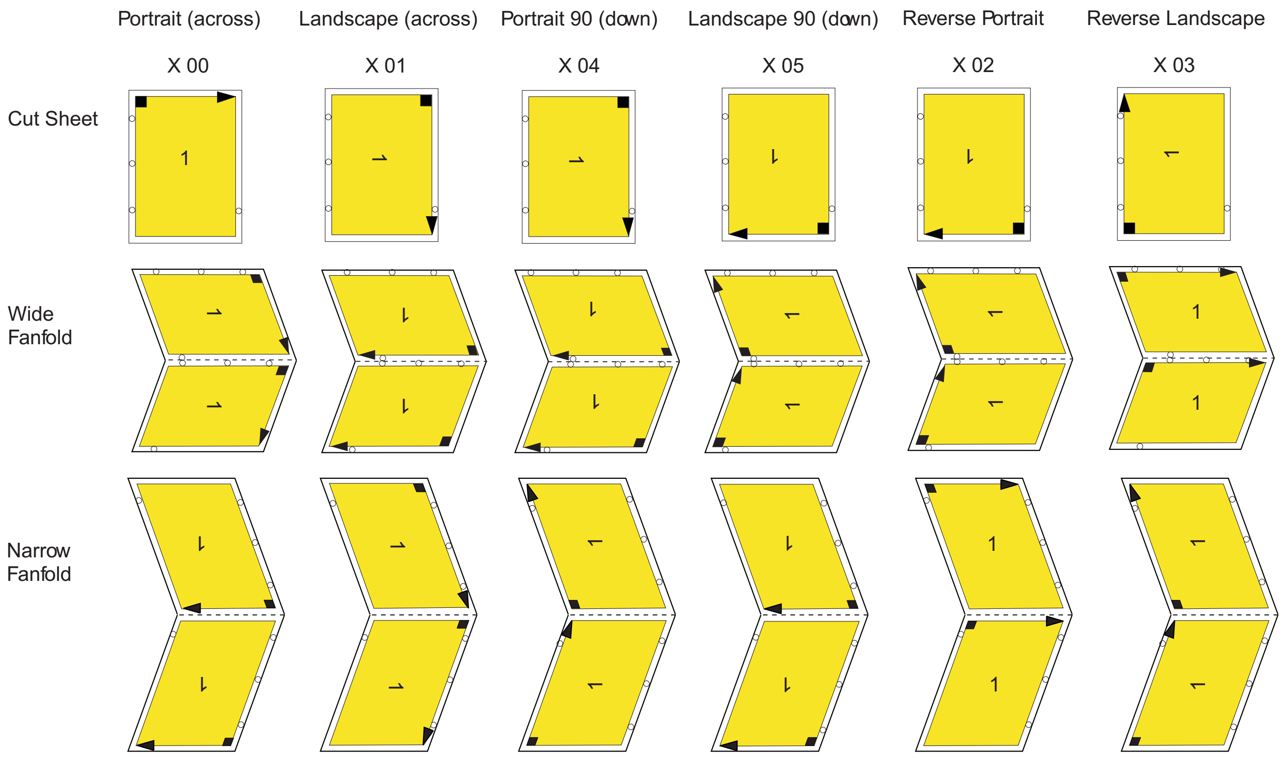


Figure 66. 1-up Partition Numbering, Back Sheet-Side, Tumble Duplex

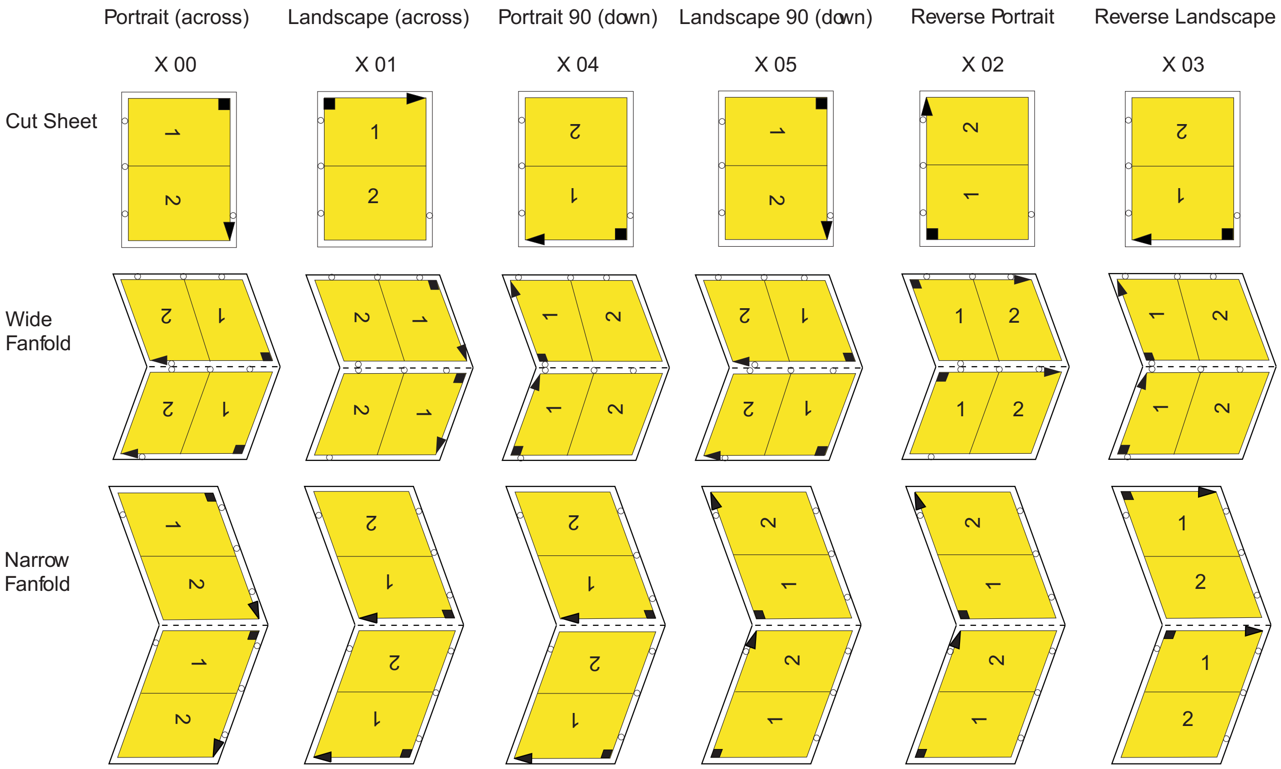


Figure 67. 2-up Partition Numbering, Back Sheet-Side, Tumble Duplex

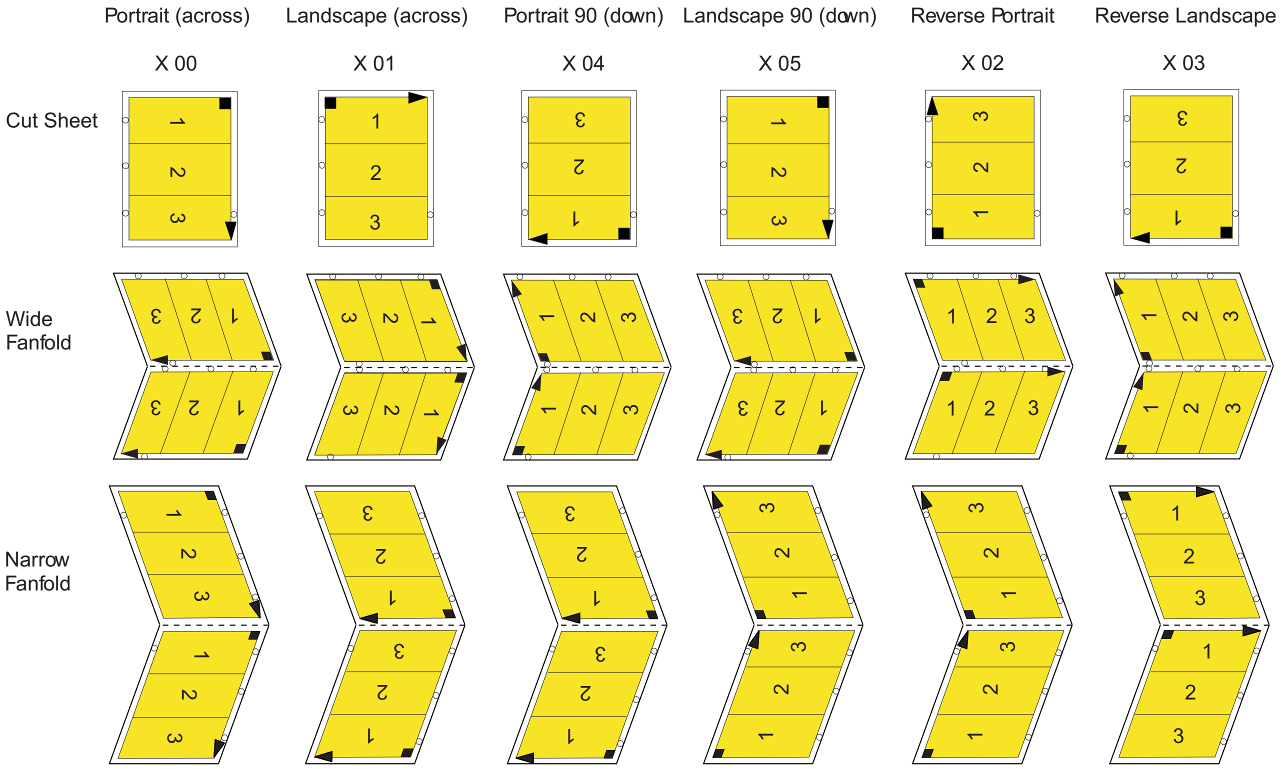


Figure 68. 3-up Partition Numbering, Back Sheet-Side, Tumble Duplex

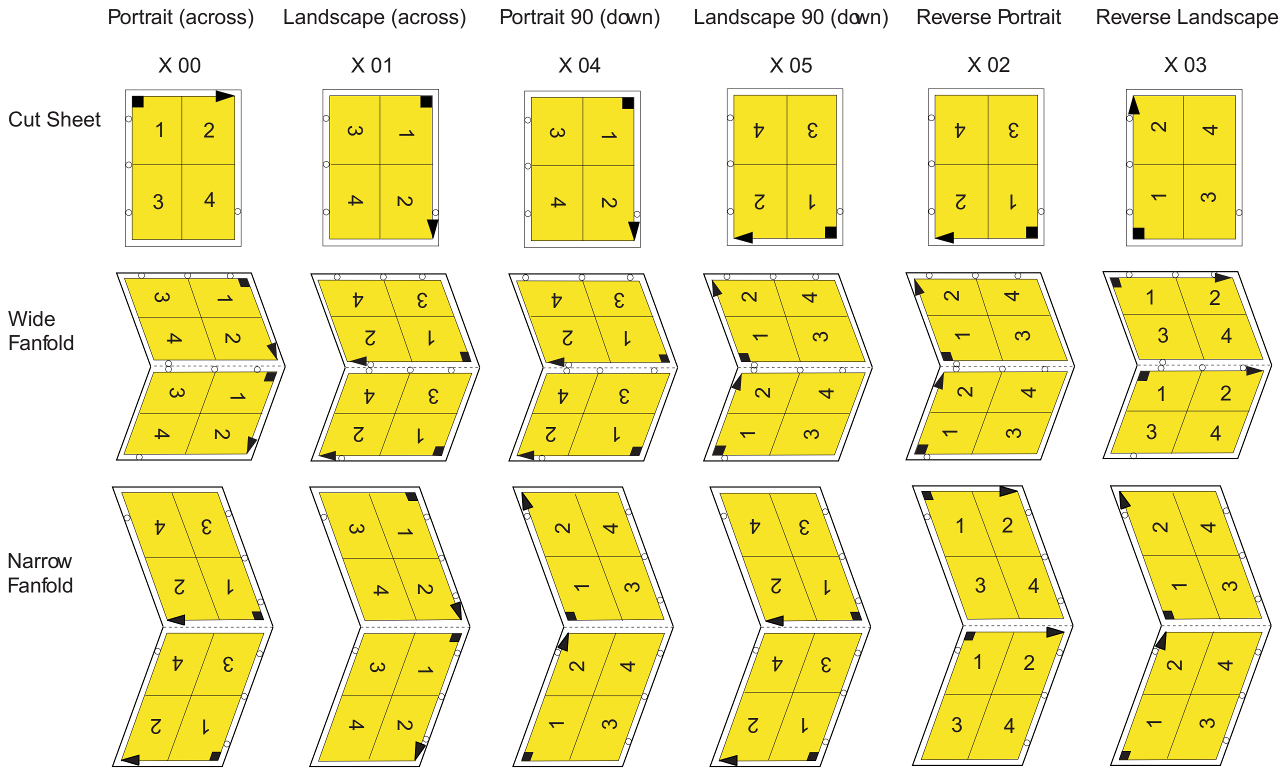


Figure 69. 4-up Partition Numbering, Back Sheet-Side, Tumble Duplex

Page Modification Control (PMC)

The Page Modification Control structured field specifies modifications to be applied to a page presented on the medium.

If the ID of a specific PMC is selected in the PGP structured field of the active medium map in N-up mode, only the modifications specified by that PMC are applied to pages placed on the medium. If a specific PMC is not selected in N-up mode, all modifications specified by all PMCs in the active medium map are applied to pages placed on the medium.

PMC (X'D3A7AF') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A7AF'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	PMCid	0–127	Page Modification Control identifier	M	X'06'
1				Reserved; must be zero	M	X'06'
2–*n*		Triplets		See “PMC Semantics” for triplet applicability.	O	X'10'

PMC Semantics

PMCid Page Modification Control Identifier. The identifier for the modifications specified by this structured field.

Triplets Appear in the Page Modification Control structured field as follows:

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur multiple times. Specifies encoding for structured field parameters defined with a CHAR data type. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.
X'4B'	Measurement Units	Optional. May occur once. Specifies the units of measure to be used for positioning included objects on the page. See “Measurement Units Triplet X'4B'” on page 364. If this triplet is omitted, the units of measure specified in the Medium Descriptor (MDD) that is in the same medium map as the PMC are used to position included objects on the page.
X'6C'	Resource Object Include	Optional. May occur more than once. Identifies an object to be included on the page at a specified position. See “Resource Object Include Triplet X'6C'” on page 388.

Note: Overlays that are included on a page using the PMC structured field are called *PMC overlays*. Each overlay included on a page with a PMC must first be mapped to a local ID with an MPO in the medium map containing the PMC.

Preprocess Presentation Object (PPO)

The Preprocess Presentation Object structured field 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 the same 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.

PPO (X'D3ADC3') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ADC3'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
One to 254 repeating groups in the following format:						
0–1	UBIN	RGLength	18–(*n*+1)	Total length of this repeating group	M	X'06'
2	CODE	ObjType	X'92', X'DF', X'FB'	Object type: **X'92'** Other object data **X'DF'** Overlay **X'FB'** Image (IOCA)	M	X'06'
3–4				Reserved; must be zero	M	X'06'
5	BITS	ObjOrent		Object orientations relative to media leading edge; see “PPO Semantics” for bit definitions	M	X'06'
6–8	SBIN	XocaOset	-32768–32767	X axis origin for object content	M	X'06'
			X'FFFFFF'	Not specified		
9–11	SBIN	YocaOset	-32768–32767	Y axis origin for object content	M	X'06'
			X'FFFFFF'	Not specified		
12–*n*		Triplets		See “PPO Semantics” for triplet applicability.	M	X'14'

PPO Semantics

RGLength Specifies the total length of the repeating group, including the length of the RGLength parameter itself.

ObjType Identifies the type of object being referenced.

Value	Description
X'92'	Other object data. The object data to be preprocessed is a non-OCA paginated presentation object. The object data is characterized and identified by a mandatory Object Classification (X'10') triplet, which must specify the registered OID for the object type and must characterize the object as being a presentation object. **Application Note:** To see which object type OIDs are supported by the presentation system, consult the product documentation. In particular, to see which object type OIDs are supported by AFP presentation servers, see the *Advanced Function Presentation: Programming Guide and Line Data Reference*.
X'DF'	Overlay object.
X'FB'	Image (IOCA) object with MO:DCA object syntax as defined in “Image Objects” on page 94.
All others	Reserved

ObjOrent Specifies one or more orientations, measured in a clockwise direction, of the X-axis of the object with respect to the leading edge of the media.

Application Note: Many factors, such as media selection, media side, media loading, media orientation, page rotation, and object area rotation affect the orientation of an object with respect to the media leading edge. Proper specification of this parameter may require visual inspection of physical output.

Bit	Description
0	0 degrees **B'0'** Do not preprocess the object at 0 degree orientation. **B'1'** Preprocess and cache the object at 0 degree orientation with respect to the leading edge of the media.
1	90 degrees **B'0'** Do not preprocess the object at 90 degree orientation. **B'1'** Preprocess and cache the object at 90 degree orientation with respect to the leading edge of the media.
2	180 degrees **B'0'** Do not preprocess the object at 180 degree orientation. **B'1'** Preprocess and cache the object at 180 degree orientation with respect to the leading edge of the media.

3 270 degrees

B'0' Do not preprocess the object at 270 degree orientation.

B'1' Preprocess and cache the object at 270 degree orientation with respect to the leading edge of the media.

4–7 Reserved; all bits must be B'0'.

If no orientations are specified, the object is preprocessed at a 0 degree orientation with respect to the leading edge of the media.

XocaOset Used in *position* and *position and trim* mappings to specify the offset along the X axis of the object area coordinate system, X_{oa}, to the X origin of the object content. The measurement units for this parameter are specified with a Measurement Units (X'4B') triplet. A value of X'FFFFFF' indicates that the X axis offset is not specified, therefore the offset value (−1) is not included in the allowed range. This parameter is ignored for ObjType = X'DF'—Overlay.

YocaOset used in *position* and *position and trim* mappings to specify the offset along the Y axis of the object area coordinate system, Y_{oa}, to the Y origin of the object content. The measurement units for this parameter are specified with a Measurement Units (X'4B') triplet. A value of X'FFFFFF' indicates that the Y axis offset is not specified, therefore the offset value (−1) is not included in the allowed range. This parameter is ignored for ObjType = X'DF'—Overlay.

Notes:

1. The object content is developed in the *data object presentation space*; within the context of this structured field the two terms are synonymous.
2. The XocaOset and YocaOset parameters are treated as a pair. If one is assigned the value X'FFFFFF' (not specified), the other is treated that way as well, regardless of its assigned value.

Triplets Appear in the Preprocess Presentation Object structured field repeating groups as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once in each repeating group. Specifies the reference to the resource object to be preprocessed. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name types that may appear are: **X'84'**—*Begin Resource Object Reference*, which is used to preprocess an overlay or an IOCA image object. The GID is used to locate the resource object in the resource hierarchy, which may include the presentation device, and must match the identifier for an object or a X’01’ exception condition exists. This FQN type is used with ObjType = X'DF'—Overlay, and with ObjType = X'FB'—IOCA image. **X'CE'**—*Other Object Data Reference*, which is used to preprocess a data object whose format may or may not be defined by an IBM presentation architecture. The GID is used to locate the object in the resource hierarchy, which may include the presentation device, and must match the identifier for an object or a X’01’ exception condition exists. This FQN type is used with ObjType = X'92'—other object data. The reference in the above FQN triplets may be specified in one—and only one—of the following formats: If FQNFmt = X'00', the reference is made with a character-encoded name. See “External Resource Naming Conventions” on page 80 for a description of the naming conventions used in AFP environments. The object reference must be specified in the same manner, using the same FQNFmt, as the MDR or MPO that maps the object as a resource.
X'10'	Object Classification	Mandatory if the repeating group specifies a Fully Qualified Name type X'CE'—Other Object Data Reference, in which case it must occur once in the repeating group and identifies the object type to be preprocessed. See “Object Classification Triplet X'10'” on page 335.

Triplet	Type	Usage
X'4B'	Measurement Units	Mandatory if the PPO specifies any of the following parameters: • XocaOset • YocaOset • XoaSize, specified in the Object Area Size (X'4C') triplet • YoaSize, specified in the Object Area Size (X'4C') triplet, In which case this triplet occurs once and defines the measurement units for the parameter values. This triplet is ignored for ObjType = X'DF'—Overlay. See “Measurement Units Triplet X'4B'” on page 364. **Note:** When the units of measure values specified on the PPO are different than the values specified on a subsequent IOB that includes the preprocessed object, the presentation device might calculate the sizes and offsets differently when processing the two structured fields, and—due to round-off errors—might not use the preprocessed version of the object. To avoid such problems, matching units of measure values should be specified on the PPO and the corresponding IOB.
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur more than once. See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur more than once. See "Fully Qualified Name Triplet X'02'" on page 320. The Fully Qualified Name type that may appear is: **DE**—*Data Object External Resource Reference*, Specifies the external identifier of a resource object that is used by the object to be preprocessed. The identifier is used by the presentation system to locate the resource object in the resource hierarchy. The identifier may be specified in one of the following two formats, but not in both formats: If FQNFmt = X'00', the identifier is a character-encoded name. See "External Resource Naming Conventions" on page 80 for a description of the naming conventions used in AFP environments. If FQNFmt = X'10', the identifier is an ASN.1 OID encoded using the definite short form. This format provides a unique and system-independent method to identify and reference an object. It may be used to select resources that are resident in, or have been captured by, the presentation device. Such an identifier is referred to as an *object OID*. If the data object that requires this resource is also processed as a resource, the term *secondary resource* is applied to the resource used by the data object. See "Secondary Resource Objects" on page 14. The secondary resource reference must be specified in the same manner, using the same FQNFmt, as the MDR that maps the secondary resource. If the object to be preprocessed also references the secondary resource with an internal identifier, this identifier must be specified on the PPO with a FQN type X'BE' triplet that immediately follows the FQN type X'DE' triplet. The paired triplets map the internal identifier to the external identifier.

Triplet	Type	Usage
X'02'	Fully Qualified Name	Optional. May occur more than once if the PPO also specifies FQN type X'DE' triplets. See “Fully Qualified Name Triplet X'02'” on page 320. The Fully Qualified Name type that may appear is: **X'BE'**—*Data Object Internal Resource Reference*. Specifies the identifier of a resource object that is used by the object being preprocessed. The identifier is used internally by the object to be preprocessed to reference the secondary resource. The identifier must be specified using FQNFmt X'00', which, for this FQN type, indicates that the data type is defined by the specific data object that generates the internal resource reference and is undefined (UNDF) at the MO:DCA data stream level. If the data object that requires this resource is also processed as a resource, the term *secondary resource* is applied to the resource used by the data object. See “Secondary Resource Objects” on page 14. When specified, this triplet must *immediately* follow the FQN type X'DE' triplet that specifies the external identifier of the secondary resource, or a X’04’ exception condition exists.
X'04'	Mapping Option	Optional. May occur once. This triplet is ignored for ObjType = X'DF'—Overlay. If present, defines the mapping of the object presentation space to the object area. The specified mapping option must be valid for the object or a X'02' exception condition exists. See “Mapping Option Triplet X'04'” on page 332.
X'4C'	Object Area Size	Optional. May occur once. This triplet is ignored for ObjType = X'DF'—Overlay. If present, specifies the size of the object area (XoaSize, YoaSize) into which the object data is mapped. See “Object Area Size Triplet X'4C'” on page 365.

Processing Rules

The purpose of the PPO is to improve system printing throughput by allowing the printer to preprocess and cache resource objects that are preloaded. If the resource is subsequently included using an IOB or IPO, a presentation-ready bit map is available. The following considerations need to be taken into account when selecting an object for preprocessing. Note that the efficiency of preprocessing is presentation-device and presentation-environment dependent.

Preprocessing overlays: Only the orientation parameter is required; all other presentation parameters, if specified, are ignored. If a subsequent include specifies one of the preprocessed orientations, the cached version of the overlay is used. The preprocessed and cached version of an overlay might not be used if any portion of the overlay exceeds the printable area when it is included.

Preprocessing data objects: A mapping that specifies how the object presentation space is mapped to the object area is required for preprocessing. For preprocessing, the mapping may be specified on the PPO with a Mapping Option (X'04') triplet. If this triplet is omitted, the mapping specified in the object’s OEG is used. If the object does not specify the mapping in an OEG, the architected default mapping for the object is used. Note that for objects referenced with ObjType = X'92' and ObjType = X'FB', the architected default mapping is scale to fit. Only the following mapping options are supported for preprocessing.

Scale-to-fit or scale-to-fill: If the mapping is scale-to-fit or scale-to-fill, the object is preprocessed into an object area size (which is required for these mappings) and cached.

For preprocessing, the object area size may be specified on the PPO with an Object Area Size (X'4C') triplet. If this triplet is omitted, the object area size specified in the object's OEG is used. If the object does not specify the object area size in an OEG, the presentation space size of the object is used. If a subsequent include specifies the same mapping, one of the preprocessed orientations, and the same object area size, the cached version of the object is used.

See "Object Type Identifiers" on page 535 for information on how the object presentation space size is specified by various non-OCA objects.

Position, position-and-trim, or center-and-trim: If the mapping is position, position-and-trim, or center-and-trim, the object is first preprocessed at the size of the object presentation space.

If a presentation window is specified by the PPO—which is defined by an object area size for center-and-trim and both an object area size and object content offset for position and position-and-trim—the preprocessed object is positioned, trimmed if required, and cached. No caching occurs if the mapping is position and there is an overflow of the object area. If a subsequent include specifies the same mapping, one of the preprocessed orientations, and the same window, the cached version of the object is used.

If a window is not specified by the PPO, the preprocessed object is cached at its presentation space size. If a subsequent include specifies any of these three mappings, one of the preprocessed orientations, and a presentation window, the cached version of the object is processed at print time—with a potential performance penalty—and trimmed if required. If the mapping is position, an exception is detected if there is an overflow of the object area.

Limitations: The PPO supports most presentation parameters that may be in effect when the preprocessed object is actually presented. However there are presentation parameters that may be in effect at presentation time that were not taken into account when the object was preprocessed. In such cases the preprocessed and cached object is not used for presentation and the system throughput improvement is not realized. Examples of such presentation parameters are:

- Specification of an unsupported preprocessing mapping, such as a migration image mapping, on the include structured field
- Specification of a color override on the include structured field, such as use of the Color Specification (X'4E') triplet to override a default OCA color
- Invocation of a non-reset Color Mapping Table
- Specification of a non-default print quality (objects are always preprocessed at default print quality)
- Activation of a text suppression for overlays (overlays are always preprocessed without text supressions).

PPO Exception Condition Summary

A X'02' exception condition exists when:

- A Fully Qualified Name (X'02') triplet other than a type X'84' (Coded Font Reference), a type X'BE' (Data Object Internal Resource Reference), type X'CE' (Other Object Data Reference), or a type X'DE' (Data Object External Resource Reference) appears within any repeating group.
- The resource reference is specified using FQNFmt X'10' (object OID), but the object either is not carried in a valid MO:DCA structure or is carried in a valid MO:DCA structure but does not have a matching object OID.

A X'01' exception condition exists when:

- A resource with the same identifier as that specified on the type X'84' (Coded Font Reference), Fully Qualified Name triplet, or on the type X'CE' (Other Object Data Reference) Fully Qualified Name triplet, or on the type X'DE' (Data Object External Resource Reference) Fully Qualified Name triplet was not previously mapped in the same resource group or could not be located.
- The same repeating group contains an invalid number or combination of Fully Qualified Name triplets.

Presentation Text Data Descriptor (PTD) Format 2

The Presentation Text Data Descriptor structured field contains the descriptor data for a presentation text data object.

PGD (X'D3B19B') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B19B'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–*n*	UNDF	PTOCAdes		Up to 32,759 bytes of PTOCA-defined descriptor data	O	X'00'

PTD Semantics

PTOCAdes Contains the PTOCA-defined text descriptor. See the MO:DCA environment appendix in the *Presentation Text Object Content Architecture Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Application Note: When the PTD is included in the AEG for a page, some AFP print servers require that the measurement units in the PTD match the measurement units in the Page Descriptor (PGD). It is therefore strongly recommended that whenever the PTD is included in the AEG, the same measurement units are specified in both the PTD and PGD.

Presentation Text Data (PTX)

The Presentation Text Data structured field contains the data for a presentation text data object.

PTX (X'D3EE9B') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3EE9B'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	PTOCAdat		Up to 32,759 bytes of PTOCA-defined data	O	X'00'

PTX Semantics

PTOCAdat Contains the PTOCA-defined text descriptor. See the MO:DCA environment appendix in the *Presentation Text Object Content Architecture Reference* for detailed information.

Note: The number of data bytes allowed in this structured field may be restricted by an interchange set.

Tag Logical Element (TLE)

A Tag Logical Element structured field assigns an attribute name and an attribute value to a page or page group. The Tag Logical Element structured field may be embedded directly in the page or page group, or it may reference the page or page group from a document index. When a Tag Logical Element structured field references a page or is embedded in a page following the active environment group, it is associated with the page. When a Tag Logical Element structured field references a page group or is embedded in a page group following the Begin Named Page Group structured field, it is associated with the page group. When a Tag Logical Element structured field is associated with a page group, the parameters of the Tag Logical Element structured field are inherited by all pages in the page group and by all other page groups that are nested in the page group. The scope of a Tag Logical Element is determined by its position with respect to other TLEs that reference, or are embedded in, the same page or page group. The Tag Logical Element structured field does not provide any presentation specifications and therefore has no effect on the appearance of a document when it is presented.

TLE (X'D3A090') Syntax

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A090'**	Flags (1B)	Reserved; X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*		Triplets		See "TLE Semantics" for triplet applicability.	M	X'14'

TLE Semantics

Triplets Appear in the Tag Logical Element structured field as follows:

Triplet	Type	Usage
X'02'	Fully Qualified Name	Mandatory. Must occur once. The Fully Qualified Name type that may appear is **X'0B'**—*Attribute Name*. Specifies the attribute name of the tag logical element. See "Fully Qualified Name Triplet X'02'" on page 320.
X'36'	Attribute Value	Mandatory. Must occur once. Specifies the attribute value of the tag logical element. See "Attribute Value Triplet X'36'" on page 354.

Triplet	Type	Usage
X'01'	Coded Graphic Character Set Global Identifier	Optional. May occur multiple times. If present, specifies the code page and character set for interpretation of subsequent character strings in the TLE. If not present, the including object specifies the code page and character set for interpretation of character strings in the TLE. By including the triplet multiple times, you can specify a unique code page and character set for the character data in every triplet on the TLE. See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317.
X'02'	Fully Qualified Name	Optional. One of the following Fully Qualified Name types may appear if the Tag Logical Element structured field references a page or page group from a document index: • **X'87'**—*Begin Page Name*. Specifies the name of the page that is referenced by the tag logical element. • **X'0D'**—*Begin Page Group Name*. Specifies the name of the page group that is referenced by the tag logical element.
X'02'	Fully Qualified Name	Optional. May occur once. The Fully Qualified Name type that may appear is **X'0C'**—*Process Element Name*. Specifies the name of the tag logical element.
X'80'	Attribute Qualifier	May occur once. Specifies an attribute qualifier for the tag logical element. See "Attribute Qualifier Triplet X'80'" on page 402.

Chapter 6. MO:DCA Triplets

This chapter:

- Describes the format, syntax, and semantics for each MO:DCA triplet
- Describes the purpose of each MO:DCA triplet parameter
- Identifies values that can be given to triplet parameters

General Information

Triplets appear after all fixed parameters in a structured field. Some structured fields may contain repeating groups of triplets. Each repeating group contains a length parameter followed by one or more triplets. An optional triplet may not appear at all, in which case a default value is used when a value is needed.

In general, when a triplet description refers to the structured field in which it appears, it refers to it as *the structured field*. When the description refers to a structured field other than the one in which it appears, it refers to that structured field by its proper name, such as *Begin Document structured field*.

Triplet Format

A triplet is a self-identifying parameter that contains three components: the length of the triplet, an ID identifying the triplet, and the associated parameters. The general format for the triplet data structure is shown below.

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3–254	Length of the triplet, including the length of Tlength	M	X'06'

Triplet Format

Offset	Type	Name	Range	Meaning	M/O	Exc
1	CODE	Tid	X'01'–X'02', X'04', X'10', X'18', X'1F', X'20'–X'22', X'24'-X'26', X'2D', X'36', X'43', X'45'–X'47', X'4B'–X'4E', X'50', X'56'–X'5A', X'5D'–X'5E', X'62', X'65', X'68', X'6C', X'70'–X'72', X'74'–X'75', X'78', X'80'–X'85', X'86', X'87', X'88', X'8B', X'8E'	Identifies the triplet: **X'01'** Coded Graphic Character Set Global Identifier **X'02'** Fully Qualified Name **X'04'** Mapping Option **X'10'** Object Classification **X'18'** MO:DCA Interchange Set **X'1F'** Font Descriptor Specification **X'20'** Coded Graphic Character Set Global Identifier **X'21'** Object Function Set Specification **X'22'** Extended Resource Local Identifier **X'24'** Resource Local Identifier **X'25'** Resource Section Number **X'26'** Character Rotation **X'2D'** Object Byte Offset **X'36'** Attribute Value **X'43'** Descriptor Position **X'45'** Media Eject Control **X'46'** Page Overlay Conditional Processing **X'47'** Resource Usage Attribute **X'4B'** Measurement Units **X'4C'** Object Area Size **X'4D'** Area Definition **X'4E'** Color Specification **X'50'** Encoding Scheme ID **X'56'** Medium Map Page Number **X'57'** Object Byte Extent **X'58'** Object Structured Field Offset **X'59'** Object Structured Field Extent	M	X'10'

Offset	Type	Name	Range	Meaning	M/O	Exc
1 (cont.)	CODE	Tid		**X'5A'** Object Offset **X'5D'** Font Horizontal Scale Factor **X'5E'** Object Count **X'62'** Object Date and Time Stamp **X'65'** Comment **X'68'** Medium Orientation **X'6C'** Resource Object Include **X'70'** Presentation Space Reset Mixing **X'71'** Presentation Space Mixing Rule **X'72'** Universal Date and Time Stamp **X'74'** Toner Saver **X'75'** Color Fidelity **X'78'** Font Fidelity **X'80'** Attribute Qualifier **X'81'** Page Position Information **X'82'** Parameter Value **X'83'** Presentation Control **X'84'** Font Resolution and Metric Technology **X'85'** Finishing Operation **X'86'** Text Fidelity **X'87'** Media Fidelity **X'88'** Finishing Fidelity **X'8B'** Data-Object Font Descriptor **X'8E'** UP3i Finishing Operation	M	X'10'
2–*n*		Contents		Contents of the triplet as identified by the MO:DCA architecture	M	X'06'

Triplet Syntax

The syntax for triplet data is the same as for structured field data. Refer to “How to Read the Syntax Diagrams” on page iv for a description of this syntax.

Triplet Semantics

Tlength Specifies the total length of the triplet, including the one-byte Tlength field. It contains a numeric value of UBIN type that ranges from 3 to 254, expressed in bytes.

Tid Identifies the triplet identifier. Permitted values are listed in the syntax table. If the value of Tid is not one of those listed in the Range column, a X’10’ exception condition exists.

Note: In all subsequent triplet syntax tables the *Exc* field contains the value X'00' for the Tid data element. This follows from the assumption that the Tid must be correct or it would not have been identified.

Contents Contains the triplet data elements. The number of data elements and the length of each is dependent on the triplet identifier.

Architected defaults are identified in the semantic description of the individual parameters. When an architected default exists for an entire triplet, the default is documented at the end of the semantic description for that triplet.

Coded Graphic Character Set Global Identifier Triplet X'01'

Certain structured fields within the data stream carry parameters that consist of a character string, such as a name. These parameters are specified to have a CHAR data type. For example the name type parameter on the Include Page Overlay structured field can be used as an identifier for a component, and as a viewable identifier to be recorded whenever the processor of the data stream associates an exception condition with the component. The Coded Graphic Character Set Global Identifier (CGCSGID) triplet is used to establish the values of the code page and character set for interpretation of all structured field parameters having a CHAR data type, such as name parameters, except where such parameters define a fixed encoding. An example of a parameter that defines its own encoding is the character string specified with a Fully Qualified Name (X'02') triplet using FQNFmt = X'20'—URL, which is encoded using the US-ASCII coded character set. The character set is specified with a Graphic Character Set Global ID (GCSGID), and the code page is specified with a Code Page Global ID (CPGID). Alternatively, the Coded Graphic Character Set Global Identifier triplet may be used to identify a Coded Character Set Identifier (CCSID) as defined and registered by the Character Data Representation Architecture (CDRA). The CCSID can be resolved to identify the value of the code page and character set for interpretation of parameters with a CHAR data type. See the *Character Data Representation Architecture Reference and Registry*, SC09-2190, for detailed information.

The scope of the Coded Graphic Character Set Global Identifier triplet is defined as follows:

- The most recent occurrence of a X'01' triplet on a structured field establishes the code page and character set used to interpret all subsequent parameters within that structured field with a CHAR data type.
- If the structured field syntax allows parameters with a CHAR data type to be positioned before the allowed triplets, then the first occurrence of a X'01' triplet on that structured field establishes the code page and character set to be used to interpret such parameters.
- If X'01' triplets appear on a Begin structured field, the last X'01' triplet specified establishes the code page and character set used to interpret all parameters with CHAR data type on all structured fields that lie between the Begin structured field and its corresponding End structured field, unless specifically overridden by a X'01' triplet on an enveloped structured field. Object names on an End structured field are always interpreted with the same code page and character set used for the object name on the corresponding Begin structured field.

Triplet X'01' Syntax: GCSGID/CPGID Form

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'01'	Identifies the Coded Graphic Character Set Global Identifier triplet	M	X'00'
2–3	CODE	GCSGID	X'0001'–X'FFFE'	Specifies the Graphic Character Set Global Identifier	M	X'06'
			X'FFFF'	Specifies the character set consisting of all characters in the code page		

Offset	Type	Name	Range	Meaning	M/O	Exc
4–5	CODE	CPGID	X'0001'–X'FFFE'	Specifies the Code Page Global Identifier	M	X'06'

Triplet X'01' Syntax: CCSID Form

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'01'	Identifies the Coded Graphic Character Set Global Identifier triplet	M	X'00'
2–3	CODE		X'0000'	Must be set to X'0000' to identify the CCSID form of the triplet	M	X'06'
4–5	CODE	CCSID	X'0000'–X'FFFF'	Coded Character Set Identifier defined by CDRA	M	X'06'

Triplet X'01' Semantics

GCSGID/CPGID Form

Tlength Contains the length of the triplet.

Tid Identifies the Coded Graphic Character Set Global Identifier triplet.

GCSGID Specifies the Graphic Character Set Global Identifier of the character set to be used in conjunction with the Code Page Global Identifier to identify the graphic characters that are represented by code points in any parameter with a data type of CHAR. The GCSGID may identify a subset or the maximal set of all of the graphic characters supported for the associated code page. Valid values for Graphic Character Set Global Identifiers are 1 through 65534. A value of 65535 (X'FFFF') indicates that a character set consisting of all characters that have assigned code points in the associated code page is to be used.

CPGID Specifies the Code Page Global Identifier of the code page to be used in conjunction with the character set to identify the graphic characters that are represented by code points in any parameter with a data type of CHAR. Valid values for Code Page Global Identifiers are 1 through 65534.

Note: The concatenation of the GCSGID and CPGID is currently referred to as the Coded Graphic Character Set Global Identifier (CGCSGID). In the past, it was also known as the Global Character Set Identifier (GCID).

CCSID Form

Bytes 2–3 Must be X'0000'. Identifies the CCSID form of the triplet.

CCSID Coded Character Set Identifier. Defined by the Character Data Representation Architecture. Can be resolved to specify the code page and character set for interpretation of parameters with CHAR data type. See the *Character Data Representation Architecture Reference and Registry*, SC09-2190, for detailed information.

Implementation Notes:

1. Most MO:DCA character strings are carried in Fully Qualified Name (FQN) triplets. This triplet limits the length of the data to 250 bytes. When such a character string is converted from one character encoding (such as single-byte EBCDIC) to another character encoding (such as double-byte UTF-16) the string may increase in length. When the new length exceeds the 250 byte triplet limit, AFP presentation servers generate an exception. Such encoding conversions are commonly used to compare object names that are specified in different encodings, therefore it is strongly recommended that object names that are specified using a single-byte encoding are limited to 125 characters or fewer.
2. There is better system support for encoding conversions using a CCSID instead of a CPGID + GCSGID combination to define the encoding of a character string, therefore it is recommended that the CCSID form of this triplet is used whenever possible.

Structured Fields Using Triplet X'01'

- "Begin Active Environment Group (BAG)" on page 104
- "Begin Bar Code Object (BBC)" on page 105
- "Begin Color Attribute Table (BCA)" on page 107
- "Begin Document (BDT)" on page 114
- "Begin Document Environment Group (BDG)" on page 111
- "Begin Document Index (BDI)" on page 112
- "Begin Form Map (BFM)" on page 116
- "Begin Graphics Object (BGR)" on page 118
- "Begin Image Object (BIM)" on page 120
- "Begin Medium Map (BMM)" on page 122
- "Begin Object Container (BOC)" on page 129
- "Begin Object Environment Group (BOG)" on page 133
- "Begin Overlay (BMO)" on page 124
- "Begin Page (BPG)" on page 134
- "Begin Named Page Group (BNG)" on page 126
- "Begin Page Segment (BPS)" on page 137
- "Begin Presentation Text Object (BPT)" on page 139
- "Begin Resource Group (BRG)" on page 141
- "Begin Resource Environment Group (BSG)" on page 148
- "Include Object (IOB)" on page 180
- "Include Page Overlay (IPO)" on page 194
- "Include Page Segment (IPS)" on page 197
- "Index Element (IEL)" on page 176
- "Invoke Medium Map (IMM)" on page 178
- "Link Logical Element (LLE)" on page 199
- "Map Coded Font (MCF) Format 2" on page 213
- "Map Color Attribute Table (MCA)" on page 207
- "Map Data Resource (MDR)" on page 223
- "Map Media Type (MMT)" on page 260
- "Map Page Overlay (MPO)" on page 265
- "Preprocess Presentation Object (PPO)" on page 299
- "Tag Logical Element (TLE)" on page 310

Fully Qualified Name Triplet X'02'

The Fully Qualified Name triplet enables the identification and referencing of objects using Global Identifiers (GIDs). A GID can be one of the following:

- A Coded Graphic Character Set Global Identifier (CGCSGID)
- A Code Page Global ID (CPGID)
- A Font Typeface Global Identifier (FGID)
- A Graphic Character Set Global Identifier (GCSGID)
- A Global Resource Identifier (GRID)
- An ASN.1 object identifier (OID), as defined in ISO/IEC 8824:1990(E)
- An encoded graphic character string that, when qualified by the associated CGCSGID, specifies a reference name
- An identifier used by a data object to reference a resource.
- A Uniform Resource Locator (URL), as defined in RFC 1738, Internet Engineering Task Force (IETF), December, 1994.

Architecture Note: Use of this GID is limited to the LLE structured field. See "Link Logical Element (LLE)" on page 199.

Implementation Note: Most MO:DCA character strings are carried in Fully Qualified Name (FQN) triplets. This triplet limits the length of the data to 250 bytes. When such a character string is converted from one character encoding (such as single-byte EBCDIC) to another character encoding (such as double-byte UTF-16), the string may increase in length. When the new length exceeds the 250 byte triplet limit, AFP presentation servers generate an exception. Such encoding conversions are commonly used to compare object names that are specified in different encodings, therefore it is strongly recommended that object names that are specified using a single-byte encoding are limited to 125 characters or fewer.

Triplet X'02' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	5–254	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'02'	Identifies the Fully Qualified Name triplet	M	X'00'

Offset	Type	Name	Range	Meaning	M/O	Exc
2	CODE	FQNType	X'01', X'07'–X'0D', X'11', X'6E', X'7E', X'83'– X'87', X'8D'–X'8E', X'98', X'B0', X'BE', X'CA', X'CE', X'DE'	Specifies how the GID will be used: **X'01'** Replace First GID name **X'07'** Font Family Name **X'08'** Font Typeface Name **X'09'** MO:DCA Resource Hierarchy Reference **X'0A'** Begin Resource Group Reference **X'0B'** Attribute GID **X'OC'** Process Element GID **X'0D'** Begin Page Group Reference **X'11'** Media Type Reference **X'6E'** Data-object Font Base Font Identifier **X'7E'** Data-object Font Linked Font Identifier **X'83'** Begin Document Reference **X'84'** Resource Object Reference **X'85'** Code Page Name Reference **X'86'** Font Character Set Name Reference **X'87'** Begin Page Reference **X'8D'** Begin Medium Map Reference **X'8E'** Coded Font Name Reference **X'98'** Begin Document Index Reference **X'B0'** Begin Overlay Reference **X'BE'** Data Object Internal Resource Reference **X'CA'** Index Element GID **X'CE'** Other Object Data Reference **X'DE'** Data Object External Resource Reference	M	X'06'
3	CODE	FQNFmt	X'00', X'10', X'20'	Specifies the GID format: **X'00'** Character string **X'10'** OID **X'20'** URL	M	X'06'
4–*n*		FQName		GID of the MO:DCA construct. Can be up to 250 bytes in length. The data type is format-dependent. See the semantic description of the *FQNFmt* parameter.	M	X'04'

Triplet X'02' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Fully Qualified Name triplet.

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

FQNType **Description**

X'01' This GID replaces the first parameter in the structured field that contains a GID name.

Note: Global Identifiers that override eight-byte positional GID names have the same semantics as the eight-byte name parameter.

X'07' The triplet contains the name of the font family. This identifier corresponds to the family name of the font design. For example, `Times New Roman` is the family name for the Monotype Times New Roman Expanded font design. The family name is a character string that normally also appears as a substring in the typeface name as specified in the Fully Qualified Name type X'08'. Font Typeface Name triplet.

Application Note: Font family names are not consistently identified in the industry, therefore it may be necessary for implementations to define a synonym table for mapping names. For example, the name `TimesNewRoman` may need to be mapped to `Times New Roman`.

X'08' This triplet contains the name of the font typeface. This identifier corresponds to the full name of the typeface as specified by the font supplier. This is the user interface name which, for example, may be used for specification or selection of the font design. It is possible that it does not correspond exactly to the font resource name, character content or supported sizes, such as in the case of ITC Italic Bold Garamond or Monotype Times New Roman Expanded.

X'09' The triplet specifies a reference to the MO:DCA resource hierarchy. The normal MO:DCA resource search order should be used for resolving a resource object reference when this triplet is specified. See “Resource Groups” on page 78.

X'0A' The triplet contains a GID reference to a Begin Resource Group structured field.

X'0B' The triplet contains the GID of a document attribute.

X'0C' The triplet contains the GID of a process element.

X'0D' The triplet contains a GID reference to a Begin Named Page Group structured field.

X'11' The triplet contains a GID reference to a media type.

X'6E' The triplet contains a GID reference to a data-object font file that defines a base font. In font linking, the base font is the font that is referenced in the data stream and that is processed first. The GID is a full font name that has been assigned to the font.

Architecture Note: This triplet is used on a TrueType Collection (TTC) container in a print-file-level resource group to specify a base TrueType/OpenType font (TTF/OTF) that is contained in the collection. Although the triplet may be specified on both the Begin Resource (BRS) and the Begin Object Container (BOC) structured fields of the collection container, AFP presentation servers always search for the triplet on the BRS.

X'7E' The triplet contains a GID reference to a data-object font file that defines a linked font. In font linking, a linked font is not referenced in the data stream and is processed in the order in which it is linked to the base font. The GID is a full font name that has been assigned to the font.

Architecture Note: This triplet is used on a TrueType/OpenType font (TTF/OTF) container or a TrueType Collection (TTC) container in a print-file level resource group to specify a linked font that is to be associated with a base font in the container. Although the triplet may be specified on both the Begin Resource (BRS) and the Begin Object Container (BOC) structured fields of the container, AFP presentation servers always use the triplet on the BRS, as follows:

- If the BRS envelopes a TTF/OTF container, the FQN type X'7E' triplet specifies a linked TTF/OTF for the font in the container.
- If the BRS envelopes a TTC container, the FQN type X'7E' triplet specifies a linked

TTF/OTF for the base font that is defined by the immediately preceding FQN type X'6E' triplet.

X'83' The triplet contains a GID reference to a Begin Document structured field.

X'84' The triplet contains a GID name reference to a begin structured field associated with a resource; or it contains a GID reference to a coded font. In MO:DCA-L data streams, the reference to a coded font is specified in the form of a coded font name. In MO:DCA-P data streams, the FQN format X'00' reference to a coded font is specified in the form of a global resource identifier (GRID). For a description of the GRID, see "Global Resource Identifier (GRID) Definition" on page 329.

X'85' The triplet contains a GID name reference to a code page that specifies the code points and graphic character names for a coded font.

Application Note: In AFP environments, the name consists of 8 characters and follows the naming conventions for AFP code pages. For a definition of these naming conventions, see the font publications referenced in "Related Publications" on page vi. An example of a code page name is T1V10500. The name is encoded in EBCDIC using code page 500 and a character set that includes the characters allowed for the name, such as CS 961 or CS 697. The allowed characters are A–Z, 0–9, $, #, @. For more information on the AFP naming conventions, see "External Resource Naming Conventions" on page 80.

X'86' The triplet contains a GID name reference to a font character set that specifies a set of graphic characters.

Application Note: In AFP environments, the name consists of 8 characters and follows the naming conventions for AFP font character sets. For a definition of these naming conventions, see the font publications referenced in "Related Publications" on page vi. An example of a font character set name is C0H40080.

The name is encoded in EBCDIC using code page 500 and a character set that includes the characters allowed for the name, such as CS 961 or CS 697. The allowed characters are A–Z, 0–9, $, #, @. For more information on the AFP naming conventions, see “External Resource Naming Conventions” on page 80.

X'87' The triplet contains a GID reference to a Begin Page structured field.

X'8D' The triplet contains a GID reference to a Begin Medium Map structured field.

X'8E' The triplet contains a GID name reference to a coded font, which identifies a specific code page and a specific font character set.

Application Note: In AFP environments, the name consists of 8 characters and follows the naming conventions for AFP coded fonts. For a definition of these naming conventions, see the font publications referenced in “Related Publications” on page vi. An example of a coded font name is X0H4108C, which identifies a Helvetica Roman Bold 8 point typeface for the Latin 1 language group. The code page is T1V10500, and the font character set is C0H40080. The name is encoded in EBCDIC using code page 500 and a character set that includes the characters allowed for the name, such as CS 961 or CS 697. The allowed characters are A–Z, 0–9, $, #, @. For more information on the AFP naming conventions, see “External Resource Naming Conventions” on page 80.

Architecture Note: A coded font name reference is also supported by the FQN type X'84' triplet in MO:DCA-L data streams, where the coded font name is the 8-character name supplied by the CPI call.

X'98' The triplet contains a GID reference to a Begin Document Index structured field.

X'B0' The triplet contains a GID reference to a Begin Overlay structured field.

X'BE' The triplet contains a GID reference to a resource used by a data object. The GID is the identifier that is used internally by the data object to reference the resource, therefore it is called an *internal* resource reference. The data type of the identifier is defined by the specific data object. Therefore, it is undefined (UNDF) at the MO:DCA data stream level. The data object that uses this resource may or may not be defined by an IBM presentation architecture.

Note: If the data object that requires this resource is also processed as a resource, the term *secondary resource* is applied to the resource used by the data object.

Architecture Note: The identifier specified by the FQN type X'BE' triplet is the identifier used within the data object to reference the resource object. It is analogous to the local ID that is used, for example, within PTOCA and GOCA objects to reference a font.

X'CA' This triplet contains the GID of an Index Element structured field.

X'CE' The triplet contains a GID reference to other object data, which may or may not be defined by an IBM presentation architecture. The GID may be a file name or any other identifier associated with the object data.

X'DE' The triplet contains a GID reference to a resource used by a data object. The GID may be a file name or any other identifier associated with the resource and is used to locate the resource object in the resource hierarchy. The data object that uses this resource may or may not be defined by an IBM presentation architecture.

Note: If the data object that requires this resource is also processed as a resource, the term *secondary resource* is applied to the resource used by the data object.

Architecture Note: he GID specified by the FQN type X'DE' triplet is the identifier used to find the resource object in the presentation system. In that sense, it is analogous, for example, to the name of a coded font that is used to find the font in a font library, or the GRID used to find a resident printer font.

All others Reserved

FQNFmt Specifies the format of the Global Identifier:

FQNFmt	Description
X'00'	The GID is either a character-encoded name, in which case the data type is CHAR, or a binary identifier, in which case the data type is CODE. The GID is a binary identifier when the FQN type X'84' specifies a GRID reference to a coded font. See “Global Resource Identifier (GRID) Definition” on page 329. In the case of FQN type X'BE'—Other Object Internal Resource Reference, the data type of the GID reference is undefined (UNDF) at the MO:DCA data stream level; it is not character (CHAR) data. In that case the data type is defined internally by the data object that generates the reference.
X'10'	The GID is an ASN.1 Object Identifier (OID), defined in ISO/IEC 8824:1990(E). The data type is CODE. The OID is encoded using the Basic Encoding Rules for ASN.1 specified in ISO/IEC 8825:1990(E). The encoding is in the definite short form and has the following syntax:

Byte	Description
0	Identifier byte, set to X'06' to indicate an OID encoding.
1	Length of content bytes that follow. Bit 0 of the length byte must be set to zero, which limits the number of content bytes to X'7F' = 127.
2–*n*	Content bytes that encode the OID component identifiers

See “Constructing Object Identifiers (OIDs)” on page 328.

X'20' The GID is a Uniform Resource Locator (URL), defined in RFC 1738, Internet Engineering Task Force (IETF), December, 1994. The data type is CHAR. The URL is encoded using the US-ASCII coded character set, which is defined in *Coded Character Set—7-bit American Standard Code for Information Interchange, ANSI X3.4* (1986).

Architecture Note: Use of this GID is limited to the LLE structured field. See “Link Logical Element (LLE)” on page 199.

All others Reserved

FQName Contains the Global Identifier (GID) of a MO:DCA construct or the GID reference to a MO:DCA construct. The format and data type of the identifier is defined by the FQNFmt parameter.

Constructing Object Identifiers (OIDs)

The construction of OIDs is shown in the following examples. Given an OID consisting of a sequence of component Identifiers, for example the OID {2.100.3} consisting of component identifiers {2, 100, 3}, the content bytes for the encoding are generated as follows.

- Each component identifier, except for the first two which are treated as a special case, is represented as a series of one or more bytes. Bit 0 of each byte is reserved to indicate whether the byte is the last in the series:

  Bit 0 = 1 The byte is not the last byte.
  Bit 0 = 0 The byte is the last byte.

  Bits 1–7 of each byte in the series are concatenated to carry the encoding of the component identifier as an unsigned binary number. The component identifier is encoded in the fewest possible bytes, that is, the leading byte of the encoding cannot have the value X'80'. Encoding starts by placing the least significant bit of the component identifier into the least significant bit of the encoded bytes.

  Example 1:

  ```
  component identifier = 200
                       = X'C8'
                       = B'1100 1000'
  ```

  Because this number has 8 significant bits, two bytes are needed to encode it: `B'1 000 0001 0 100 1000' =` X'8148'.

  Example 2:

  ```
  component identifier = 3
                       = X'03'
                       = B'0000 0011'
  ```

  Because this number has 2 significant bits, only one byte is needed to encode it: `X'0 000 0011' = X'03'`.

- The first two component identifiers, represented by x and y in the OID (x.y.z.....), are combined into a single number using the equation

  ```
  (x•40) + y
  ```

  The resulting number is then encoded into the first series of content bytes using the previously defined algorithm. Therefore, the *n*th component identifier in the OID (*n*>2) is represented by the (*n*–1)'th series of bytes in the content.

  Example 3:

  ```
  OID {2.100.3}
  Encoded OID = X'06 03 813403'
  ```

  Example 4:

  ```
  OID {1.3.18.0.4.1.1.14}
  Encoded OID = X'06 07 2B12000401010E'
  ```

Application Note: The purpose of supporting ISO object identifiers in the FQN triplet is to provide a means for generating MO:DCA object identifiers that are *guaranteed* to be unique across all environments that generate these identifiers in accordance with the ISO standard. When OIDs are used in a MO:DCA data stream to identify and reference objects, the presentation system assumes that the OIDs have been generated properly and have been uniquely assigned to objects. That is, the MO:DCA presentation system assumes that:

- If an object is assigned an OID, no other object can be assigned the same OID
- If the object definition is changed, the object must be assigned a new and different OID

This allows the presentation system to manage objects by their OIDs in a manner that is independent of time, location, and platform. Any violation of these rules will result in unpredictable and incorrect presentation.

Global Resource Identifier (GRID) Definition

The global resource identifier (GRID) is an eight-byte binary identifier used to reference a coded font. It consists of a concatenation of the following four binary items:

Byte **Content**

0–1 The two-byte binary Graphic Character Set Global Identifier (GCSGID). The character set defined by the GCSGID is associated with the coded font and identifies a minimum set of coded font graphic characters required for presentation. It may be a character set that is associated with the code page, or with the font character set, or with both. Valid values are 1–65534. A value of 65535 (X'FFFF') indicates that a character set consisting of all characters that have assigned code points in the associated code page is to be used.

2–3 The two-byte binary Code Page Global Identifier (CPGID) assigned to the code page. Valid values are 1–65534.

4–5 The two-byte binary Font Typeface Global ID (FGID) assigned to the font design. Valid values are 1–65534.

6–7 A two-byte binary number that represents the font width (specified horizontal font size) in 1440ths of an inch (see the *Font Object Content Architecture Reference* for a description of the horizontal font size parameter). Valid values are 1–32767. A value of 0 indicates that the font width is not specified. The value X'FFFF' is retired; see "Retired Parameters" on page 515.

The font width may be used to generate the specified vertical font size, which is used to scale outline technology fonts to the desired point size, as follows:

- For typographic, proportionally-spaced fonts, the vertical font size is three times the font width.
- For fixed-pitch, uniform character increment fonts, including Proportional Spacing Machine (PSM) fonts, the vertical font size is calculated as follows:

$$\text{vertical font size} = \frac{1000 \cdot \text{font width}}{\begin{array}{c}\text{space character increment}\\ \text{(in relative units)}\end{array}}$$

Application Note: For IBM Core Interchange Courier fonts, and for IBM Expanded Core fonts with FGID values less than 750 and with FGID values between 3840 and 4095 inclusive (fixed pitch, uniform character increment, and PSM fonts), a value of 600 relative units can be used for the space character increment.

If the generated vertical font size conflicts with the nominal vertical font size in the font object, the generated vertical font size overrides.

Architecture Note: Code page objects and font character set objects may each be associated with multiple character sets. Because the GRID only specifies a single character set, the presentation server that resolves the GRID reference must understand subset/superset relationships between the character set specified in the GRID and the character sets associated with the referenced code page and font character set. All graphic characters in the specified character set must also belong to a character set associated with the code page and a character set associated with the font character set. To optimize coded font selection, generators of the GRID should specify the smallest character set that is a subset of both a character set associated with the code page and a character set associated with the font character set.

Structured Fields Using Triplet X'02'

- “Begin Bar Code Object (BBC)” on page 105
- “Begin Color Attribute Table (BCA)” on page 107
- “Begin Document (BDT)” on page 114
- “Begin Document Index (BDI)” on page 112
- “Begin Graphics Object (BGR)” on page 118
- “Begin Image Object (BIM)” on page 120
- “Begin Overlay (BMO)” on page 124
- “Begin Named Page Group (BNG)” on page 126
- “Begin Page (BPG)” on page 134
- “Begin Object Container (BOC)” on page 129
- “Begin Presentation Text Object (BPT)” on page 139
- “Begin Resource Group (BRG)” on page 141
- | “Begin Resource (BRS)” on page 143
- “End Bar Code Object (EBC)” on page 152
- “End Color Attribute Table (ECA)” on page 153
- “End Document (EDT)” on page 157
- “End Document Index (EDI)” on page 156
- “End Graphics Object (EGR)” on page 159
- “End Image Object (EIM)” on page 160
- “End Overlay (EMO)” on page 162
- “End Object Container (EOC)” on page 164
- “End Page (EPG)” on page 166
- “End Named Page Group (ENG)” on page 163
- “End Presentation Text Object (EPT)” on page 168
- “End Resource Group (ERG)” on page 170
- “Index Element (IEL)” on page 176
- “Include Object (IOB)” on page 180
- “Include Page Overlay (IPO)” on page 194
- “Link Logical Element (LLE)” on page 199
- “Map Coded Font (MCF) Format 2” on page 213
- “Map Color Attribute Table (MCA)” on page 207
- “Map Data Resource (MDR)” on page 223
- “Map Media Type (MMT)” on page 260
- “Map Page Overlay (MPO)” on page 265

- “Preprocess Presentation Object (PPO)” on page 299
- “Tag Logical Element (TLE)” on page 310

Mapping Option Triplet X'04'

The Mapping Option is used to specify the mapping of a data object presentation space to an object area.

Triplet X'04' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'04'	Identifies the Mapping Option triplet	M	X'00'
2	CODE	MapValue	X'00', X'10', X'20', X'30', X'41', X'42', X'50', X'60'	Data object mapping option: **X'00'** Position **X'10'** Position and trim **X'20'** Scale to fit **X'30'** Center and trim **X'41'** Migration mapping **X'42'** Migration mapping **X'50'** Migration mapping **X'60'** Scale to fill	M	X'06'

Triplet X’04’ Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Mapping Option triplet.

MapValue Specifies the mapping option to be used for the data object referenced by the structured field.

Note: Not all mapping options are supported for all data objects; see the Map structured field for each data object to see which options are supported.

Value	Description
X'00'	Position. The upper left corner of the data object’s presentation space or window is positioned coincident with the data object’s content origin specified in the XocaOset and YocaOset parameters in the Object Area Position structured field. All data must be presented within the object area extents, or a X’01’ exception condition exists.
X'10'	Position and trim. The upper left corner of the data object’s presentation space or window is positioned coincident with the data object’s content origin specified in the XocaOset and YocaOset parameters in the Object Area Position structured field. All data that falls within the object area extents is presented, but data that falls outside of the object area is not presented.
X'20'	Scale to fit. The center of the data object’s presentation space or window is mapped to the center of the object area defined by the associated Object Area Descriptor structured field. The data

object is symmetrically scaled up or down while preserving the aspect ratio so that, at its maximum data size, it is totally contained in the object area.

When this option is specified, the data object's content origin specified in the XocaOset and YocaOset parameters in the Object Area Position structured field is ignored.

Note: For static presentation objects, a presentation space size is required for a scale-to-fit mapping of the object presentation space to the object area. If the size of the presentation space is not specified by the object data descriptor, the object data itself may specify the size. See "Object Type Identifiers" on page 535 for information on how the presentation space size is specified by various objects. If the presentation space size is not specified in the data descriptor, and if it is also not specified by the object, the architected default is the presentation space size of the including page or overlay.

X'30' Center and trim. The center of the data object's presentation space or window is mapped to the center of the object area defined by the associated Object Area Descriptor structured field. All data that falls within the object area is presented, but data that falls outside of the object area is not presented.

When this option is specified, the data object's content origin specified in the XocaOset and YocaOset parameters in the Object Area Position structured field is ignored.

X'41' Migration mapping. See "Coexistence Parameters" on page 533 for a description.

X'42' Migration mapping. See "Coexistence Parameters" on page 533 for a description.

X'50' Migration mapping. See "Coexistence Parameters" on page 533 for a description.

X'60' Scale to fill. The center of the data object's presentation space or window is mapped to the center of the object area defined by the associated Object Area Descriptor structured field. The data object is scaled up or down so that it totally fills the object area in both the X and Y directions. This may require that the object presentation space be asymmetrically scaled by different scale factors in the X and Y directions. Therefore, this mapping does not, in general, preserve the the aspect ratio of the data object.

When this option is specified, the data object's content origin specified in the XocaOset and YocaOset parameters in the Object Area Position structured field is ignored.

Note: For static presentation objects, a presentation space size is required for a scale-to-fill mapping of the object presentation space to the object area. If the size of the presentation space is not specified by the object data descriptor, the object data itself may specify the size. See "Object Type Identifiers" on page 535 for information on how the presentation space size is specified by various objects. If the presentation space size is not specified in the data descriptor, and if it is also not specified by the object, the architected default is the presentation space size of the including page or overlay.

All others
Reserved

Structured Fields Using Triplet X'04'

- "Include Object (IOB)" on page 180
- "Map Bar Code Object (MBC)" on page 206
- "Map Container Data (MCD)" on page 211
- "Map Graphics Object (MGO)" on page 245
- "Map Image Object (MIO)" on page 246
- "Preprocess Presentation Object (PPO)" on page 299

Object Classification Triplet X'10'

The Object Classification is used to classify and identify object data. The object data may or may not be defined by an IBM presentation architecture.

Triplet X'10' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	24–96	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'10'	Identifies the Object Classification triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3	CODE	ObjClass	X'01', X'10', X'20', X'30', X'40', X'41'	Specifies the object class: **X'01'** Time-invariant paginated presentation object **X'10'** Time-variant presentation object **X'20'** Executable program (non-presentation object) **X'30'** Set-up file (non-presentation object); document level **X'40'** Secondary Resource **X'41'** Data-object font	M	X'06'
4–5				Reserved; must be zero	M	X'06'
6–7	BITS	StrucFlgs		Provides information on the structure of the object container. See “Triplet X'10' Semantics” for StrucFlgs bit definitions.	M	X'06'
8–23	CODE	RegObjId		MO:DCA-registered ASN.1 object identifier (OID) for object type.	M	X'06'
24–55	CHAR	ObjTpName		Name of the object type	O	X'00'
56–63	CHAR	ObjLev		Release level or version number of the object type	O	X'00'
64–95	CHAR	CompName		Name of company or organization that owns object definition	O	X'00'

Triplet X'10' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Classification triplet.

ObjClass Specifies the object class based on differentiators such as temporal characteristics and presentation form.

Value Description

X'01' Time-invariant paginated presentation object. If included for presentation, the scope of the object is the including page or overlay.

X'10' Time-variant presentation object. The scope of the object is not defined.

X'20' Executable program such as an object handler. This is not a presentation object, that is, it is not a specification of final-form paginated object data. The scope of the object is not defined.

X'30' Set-up information file, document level. This is not a presentation object, that is, it is not a specification of final-form paginated object data. The scope of the object is the document or documents for which the set-up file is invoked.

X'40' Secondary resource. This is a resource used by a presentation object that may itself be a resource object. The resource itself is not a stand-alone page-level presentation object. The scope of the resource is the object that uses the resource.

X'41' Data-object font. This is a non-FOCA font resource used to present text in a data object. Examples of data-object fonts are TrueType fonts and OpenType fonts. This object class includes collections of data-object fonts, such as TrueType Collections (TTCs). The resource itself is not a stand-alone page-level presentation object. The scope of the resource is the data object that uses the resource. If the data object that uses this font is also a resource, the font resource becomes a secondary resource.

All others
Reserved

StrucFlgs Flags that characterize the structure of the object data. StrucFlgs bits have the following definitions:

Bits **Description**

0–1 Object Container (BOC/EOC)

B'00' Reserved

B'01' The object data is not carried in a MO:DCA object container.

B'10' The container structure of the object data is unknown.

B'11' The object data is carried in a MO:DCA object container.

Notes:

1. These bits must be set to B'11' when the triplet appears on a Begin Object Container (BOC) structured field.
2. When bits 0–1 are set to B'11', bits 4–5 must also be set to B'11'.
3. It is not advisable to set the bits to B'11' when the triplet appears on a structured field that references the

object such as an Include Object (IOB), since the reference would become invalid if the object data is eventually carried in a MO:DCA object container.

2–3 Object environment group (OEG)

B'00' Reserved

B'01' Object container does not include an OEG.

B'10' It is not known whether the object structure includes an OEG.

B'11' Object container includes an OEG for the object data.

Notes:

1. When bits 2–3 are set to B'11', bits 0–1 must be set to B'11', and bits 4–5 must be set to B'11'.
2. It is not advisable to set the bits to B'01' when the triplet appears on a structured field that references the object such as an Include Object (IOB), since the reference would become invalid if an OEG is eventually added.

4–5 Object Container Data (OCD) structured fields

B'00' Reserved

B'01' Object data is not carried in OCD structured fields.

B'10' It is not known whether the object data is carried in OCD structured fields.

B'11' Object data is carried in OCD structured fields.

Notes:

1. When bits 4–5 are set to B'11', bits 0–1 must also be set to B'11'. Conversely, when bits 0–1 are set to B'11', bits 4–5 must also be set to B'11'.
2. It is not advisable to set the bits to B'01' when the triplet appears on a structured field that references the object such as an Include Object (IOB), since the reference would become invalid if the object data is eventually carried in OCD structured fields.

6–15 Reserved; all bits must be B'0'.

RegObjId Specifies a unique numeric identifier for the object type carried in the object container. The numeric identifier is an ASN.1 Object Identifier (OID), defined in ISO/IEC 8824:1990(E), whose last component identifier is registered in the MO:DCA architecture. The complete OID is encoded using the Basic Encoding Rules for ASN.1 specified in ISO/IEC 8825:1990(E). A table of the registered component identifiers and the encoded OIDs is provided in “Object Type Identifiers” on page 535. The OID is left justified and padded with zeros. This identifier is mandatory.

ObjTpName Specifies the generic name used to refer to the object type. The name is left-justified and padded with blanks. A value of all blanks, encoded using the active code page and character set, indicates that the name is not specified.

ObjLev Specifies the release level or version number of the object type. The level is left-justified and padded with blanks. A value of all blanks, encoded using the active code page and character set, indicates that the level is not specified.

CompName Specifies the name of the company or organization that owns the syntactic and semantic definition of the object type. The name is left-justified and padded with blanks. If the object type is defined by a standards organization, specifies the name of that standards organization. A value of all blanks, encoded using the active code page and character set, indicates that the name is not specified.

Note: If an optional positional parameter is included on this triplet, all preceeding optional positional parameters become mandatory.

Application Note: The following illustrates how the parameters in this triplet can be used to identify and classify non-OCA object data:

- Encapsulated PostScript object that is carried in a MO:DCA object container:

Parameter	**Value**
ObjClass	X'01'
StrucFlgs	X'EC00'
ObjId	X'06072B12000401010D'
ObjTpName	Encapsulated PostScript
ObjLev	2.0
CompName	Adobe

- TIFF single-page image object whose container structure is not known:

Parameter	**Value**
ObjClass	X'01'
StrucFlgs	X'A800'
ObjId	X'06072B12000401010E'
ObjTpName	TIFF
ObjLev	6.0
CompName	Aldus

Structured Fields Using Triplet X'10'

- “Begin Object Container (BOC)” on page 129
- “Include Object (IOB)” on page 180
- “Link Logical Element (LLE)” on page 199
- “Map Data Resource (MDR)” on page 223
- “Preprocess Presentation Object (PPO)” on page 299

MO:DCA Interchange Set Triplet X'18'

The MO:DCA Interchange Set triplet identifies the interchange set and the data stream type.

Triplet X'18' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	5	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'18'	Identifies the MO:DCA Interchange Set triplet	M	X'00'
2	CODE	IStype	X'01', X'03'	Specifies the type of interchange set: **X'01'** Presentation **X'03'** Resource	M	X'06'
3–4	CODE	ISid	X'0900', X'0C00'	Interchange set identifier: **X'0900'** MO:DCA-P IS/1 **X'0C00'** MO:DCA-P IS/2 or MO:DCA-L	M	X'06'

Triplet X'18' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the MO:DCA Interchange Set triplet.

ISType Specifies the interchange set type. The valid interchange set type codes are:

Value	Description
X'01'	Presentation Document
X'03'	Resource Document
All others	Reserved

ISid Specifies the interchange set identifier.

The code assignments for a presentation document interchange set, type X'01', are:

Value	Description
X'0900'	MO:DCA-P IS/1. See “MO:DCA Presentation Interchange Set 1” on page 429
X'0C00'	MO:DCA-P IS/2. See “MO:DCA Presentation Interchange Set 2” on page 444 for details.
All others	Reserved

The code assignments for a resource document interchange set, type X'03', are:

Value	Description
X'0C00'	MO:DCA-L. See “MO:DCA Resource Interchange Set” on page 463 for details.
All others	Reserved

Note: Data streams that do not comply completely with an interchange set, such as those intended for private use or exchange purposes, must ensure that this triplet is *not* specified on the Begin Document structured field.

Structured Fields Using Triplet X'18'

- “Begin Document (BDT)” on page 114

Font Descriptor Specification Triplet X'1F'

The Font Descriptor Specification triplet specifies the attributes of the desired font in a coded font reference.

Triplet X'1F' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	9–20	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'1F'	Identifies the Font Descriptor Specification triplet	M	X'00'
2	CODE	FtWtClass	X'00'–X'09'	Specifies character stroke thickness: **X'00'** Not specified **X'01'** Ultra-light **X'02'** Extra-light **X'03'** Light **X'04'** Semi-light **X'05'** Medium (normal) **X'06'** Semi-bold **X'07'** Bold **X'08'** Extra-bold **X'09'** Ultra-bold	M	X'06'
3	CODE	FtWdClass	X'00'–X'09'	Specifies character width-to-height ratio: **X'00'** Not specified **X'01'** Ultra-condensed **X'02'** Extra-condensed **X'03'** Condensed **X'04'** Semi-condensed **X'05'** Medium (normal) **X'06'** Semi-expanded **X'07'** Expanded **X'08'** Extra-expanded **X'09'** Ultra-expanded	M	X'06'
4–5	UBIN	FtHeight	0–32767	Specifies vertical font size in 1440ths of an inch or in world coordinate values	M	X'06'
6–7	UBIN	FtWidth	0–32767	Specifies horizontal font size in 1440ths of an inch or in world coordinate values.	M	X'06'
8	BITS	FtDsFlags		Qualifies the type of font characters. See “Triplet X'1F' Semantics” for FtDsFlags bit definitions.	M	X'06'
9–18				Reserved; not checked	O	X'00'
19	BITS	FtUsFlags		Describes the font environment. See “Triplet X'1F' Semantics” for FtUsFlags bit definitions.	O	X'02'

Triplet X'1F' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Font Descriptor Specification triplet.

FtWtClass Is a code that describes the thickness of strokes of the characters as one of the following values:

Value	Description
X'00'	Not specified
X'01'	Ultra-light
X'02'	Extra-light
X'03'	Light
X'04'	Semi-light
X'05'	Medium (normal)
X'06'	Semi-bold
X'07'	Bold
X'08'	Extra-bold
X'09'	Ultra-bold
All others	Reserved

FtWdClass Is a code that describes the relative width-to-height ratio of the characters as one of the following values:

Value	Description
X'00'	Not specified
X'01'	Ultra-condensed
X'02'	Extra-condensed
X'03'	Condensed
X'04'	Semi-condensed
X'05'	Medium (normal)
X'06'	Semi-expanded
X'07'	Expanded
X'08'	Extra-expanded
X'09'	Ultra-expanded
All others	Reserved

FtHeight Specifies the vertical size of the font character set in 1440ths of an inch or in world coordinate values. (See the *Font Object Content Architecture Reference* for a description of the Vertical Font Size parameter).

Architecture Note: The use of world coordinate values is limited to MO:DCA-L data streams. This measure is not used in MO:DCA-P data streams.

The specified vertical font size is used to select a raster font or to scale an outline technology font to the desired point size. A value of zero indicates that the vertical font size is not specified. If the specified vertical font size conflicts with the nominal vertical font size in the font object, the specified vertical font size overrides.

FtWidth Specifies the horizontal size of the font character set in 1440ths of an inch or in world coordinate values. (See the *Font Object Content Architecture Reference* for a description of the Horizontal Font Size parameter).

Architecture Note: The use of world coordinate values is limited to MO:DCA-L data streams. This measure is not used in MO:DCA-P data streams.

A value of zero indicates that the horizontal font size is not specified.

Note: The specified horizontal font size may be used to generate the vertical font size, which is used to select a raster font or to scale an outline technology font to the desired point size, as follows:

- For typographic, proportionally-spaced fonts, the vertical font size is three times the horizontal font size.
- For fixed-pitch, uniform character increment fonts, including Proportional Spacing Machine (PSM) fonts, the vertical font size is calculated as follows:

```
                              1000 · font width
vertical font size =  -------------------------
                          space character increment
                             (in relative units)
```

If the generated vertical font size conflicts with the specified vertical font size, the specified vertical font size takes precedence.

Application Note: For IBM Core Interchange Courier fonts, and for IBM Expanded Core fonts with FGID values less than 750 and with FGID values between 3840 and 4095 inclusive (fixed pitch, uniform character increment, and PSM fonts), a value of 600 relative units can be used for the space character increment.

FtDsFlags Qualify the type of font characters. Flag bit 7 defines the meaning of this parameter when all other flag bits have the value B'0'. FtDsFlags bits have the following descriptions:

Bit **Description**

0 Italic characters:
B'0' Font contains no italic characters.
B'1' Font contains italic characters.

1 Underscored characters:
B'0' Font contains no underscored characters.
B'1' Font contains underscored characters.

2 Reserved; must be B'0'

3 Hollow characters:
B'0' Font contains no hollow characters.
B'1' Font contains hollow characters.

4 Overstruck characters:
B'0' Font contains no overstruck characters.
B'1' Font contains overstruck characters.

5 Proportionally spaced characters:
B'0' Font contains uniformly spaced characters.
B'1' Font contains proportionally spaced characters.

6 Pairwise kerned characters:

B'0' Font contains no pairwise kerned characters.
B'1' Font contains pairwise kerned characters.

7 Definition of FtDsFlags parameter when bits 0–6 = B'0000000':
B'0' Parameter is not specified.
B'1' Parameter is specified; each flag bit carries its assigned meaning.

FtUsFlags Describe the font environment.

Bit	Description
0	Reserved; must be B'0'
1	Font type: **B'0'** Bitmapped font **B'1'** Outline or vector font
2	Transform font: **B'0'** Font will not be transformed. **B'1'** Font may be transformed, that is, scaled, rotated, or sheared.
3–7	Reserved; all bits must be B'0'.

Structured Fields Using Triplet X'1F'

- "Map Coded Font (MCF) Format 2" on page 213

Font Coded Graphic Character Set Global Identifier Triplet X'20'

The Font Coded Graphic Character Set Global Identifier triplet is used to specify the code page and character set for a coded font.

Triplet X'20' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'20'	Identifies the Font Coded Graphic Character Set Global Identifier triplet	M	X'00'
2–3	CODE	GCSGID	X'0001'–X'FFFE'	Specifies the Graphic Character Set Global Identifier	M	X'06'
			X'FFFF'	Specifies the character set consisting of all characters in the code page		
4–5	CODE	CPGID	X'0001'–X'FFFE'	Specifies the Code Page Global Identifier	M	X'06'

Triplet X'20' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Font Coded Graphic Character Set Global Identifier triplet.

GCSGID Specifies the two-byte binary Graphic Character Set Global Identifier (GCSGID). The character set defined by the GCSGID is associated with the coded font and identifies a minimum set of coded font graphic characters required for presentation. It may be a character set that is associated with the code page, or with the font character set, or with both. Valid values for Graphic Character Set Global Identifiers are 1 through 65534. A value of 65535 (X'FFFF') indicates that a character set consisting of all characters that have assigned code points in the associated code page is to be used.

CPGID Specifies the two-byte binary Code Page Global Identifier (CPGID) assigned to the code page associated with the coded font. Valid values for Code Page Global Identifiers are 1 through 65534.

Note: The concatenation of the GCSGID and CPGID is currently referred to as the Coded Graphic Character Set Global Identifier (CGCSGID). In the past, it was also known as the Global Character Set Identifier (GCID).

Structured Fields Using Triplet X'20'

- “Map Coded Font (MCF) Format 2” on page 213
- “Map Data Resource (MDR)” on page 223

Object Function Set Specification Triplet X'21'

The Object Function Set Specification triplet is used to specify the Object Content Architecture (OCA) level for an object in a MO:DCA-P IS/1 data stream.

Triplet X'21' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	8–254	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'21'	Identifies the Object Function Set Specification triplet	M	X'00'
2	CODE	ObjType	X'02'–X'03', X'05'–X'06'	Specifies the OCA: **X'02'** Presentation Text **X'03'** Graphics **X'05'** Retired value **X'06'** Image	M	X'06'
3	CODE	ArchVrsn	X'00'	Specifies the architecture level of the OCA	M	X'06'
4–5	CODE	DCAFnSet	X'8000'	Specifies the MO:DCA function set identifier	M	X'06'
6–7	CODE	OCAFnSet	X'0000', X'4000', X'8000'	Identifies the OCA function set: **X'0000'** PTOCA PT1 **X'4000'** GOCA DR/2V0 **X'8000'** IOCA FS10	M	X'06'
8–*n*				Reserved; not checked	O	X'00'

Triplet X'21' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Function Set Specification triplet.

ObjType Specifies the object for which a function set is being defined. The codes for the objects are as follows:

Value	Description
X'02'	Presentation Text
X'03'	Graphics
X'05'	Retired object type. See “Retired Parameters” on page 515.
X'06'	Image
All others	Reserved

ArchVrsn Specifies the architecture level of the OCA.

DCAFnSet Defines the function set for the group of MO:DCA constructs identified by the ObjType parameter.

OCAFnSet Specifies the function set of the OCA defined by the ObjType parameter. The presence of this parameter containing the value X'0000' indicates that at least one object from the base function set is present in the data stream. OCAFnSet values have the following meanings:

Value **Description**

X'0000'	Presentation text data, PT1 level
X'4000'	Graphics data, DR/2V0 level
X'8000'	Image data, IOCA FS10 level
All others	Reserved

Architecture Note:

1. The OCAFnSet parameter value of X'0000' is used with retired object type X'05'. See "Retired Parameters" on page 515.
2. The OCAFnSet parameter value of X'4000' is retired for object type X'02' to indicate PTOCA PT2. See "Retired Parameters" on page 515.

Structured Fields Using Triplet X'21'

- "Begin Document (BDT)" on page 114

Extended Resource Local Identifier Triplet X'22'

The Extended Resource Local Identifier triplet specifies a resource type and a four-byte local identifier or LID. The LID usually is associated with a specific resource name by a map structured field, such as a Map Data Resource structured field, or a Map Media Type structured field.

Triplet X'22' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'22'	Identifies the Extended Resource Local Identifier triplet	M	X'00'
2	CODE	ResType	X'10', X'30', X'40'	Specifies the resource type: **X'10'** Image resource **X'30'** Retired value **X'40'** Media Type resource	M	X'06'
3–6	CODE	ResLID	X'00000000'–X'FFFFFFFF'	Specifies the extended resource local ID: **X'00000000'–X'0000FFFF'** Resource type X'40' **X'00000000'–X'FFFFFFFF'** Resource types other than X'40'	M	X'06'

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'10'	Image Resource
X'30'	Retired for private use. See “Retired Parameters” on page 515. **Architecture Note:** This value is used in AFP line-data environments in a Page Definition object to denote an IOB Reference. It matches an Include Object (IOB) structured field to a Descriptor. For more information see *Advanced Function Presentation: Programming Guide and Line Data Reference*.
X'40'	Media type resource
All others	Reserved

ResLID Specifies a unique resource object Local ID. It may be in the range of X'00000000' to X'FFFFFFFF' for all resource types other than X'40'. For resource type X'40' (media type), the range is restricted to X'00000000' to X'0000FFFF'.

Architecture Note: The local IDs used with resource type X'40' are specified with a X'EB*nn*' + X'E9*nn*' keyword pair on the MMC that can only carry a 2-byte ID. Therefore, the range for this resource type is restricted to 2-byte values.

Structured Fields Using Triplet X'22'

- "Map Data Resource (MDR)" on page 223
- "Map Media Type (MMT)" on page 260

Resource Local Identifier Triplet X'24'

The Resource Local Identifier triplet may be used to specify a resource type and a one-byte local identifier or LID. The LID usually is associated with a specific resource name by a map structured field, such as a Map Coded Font structured field.

Triplet X'24' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'24'	Identifies the Resource Local Identifier triplet	M	X'00'
2	CODE	ResType	X'00', X'02', X'05', X'07'	Specifies the resource type: **X'00'** Usage-dependent **X'02'** Page Overlay **X'05'** Coded Font **X'07'** Color Attribute Table	M	X'06'
3	CODE	ResLID	X'00'–X'FE'	Specifies the resource local ID	M	X'06'

Triplet X'24' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Resource Local Identifier triplet.

ResType Specifies the resource type associated with the local ID.

Value	Description
X'00'	Usage-dependent. The resource type is implied by the context of the structured field in which this triplet parameter occurs. A X’01’ exception condition exists if more than one resource local ID occurs within a given structured field and this value is specified.
X'02'	Page Overlay resource
X'05'	Coded Font resource
X'07'	Color Attribute Table resource
All others	Reserved

ResLID Specifies a unique resource object local ID. It may be in the range of X'00' to X'FE'.

Structured Fields Using Triplet X'24'

- “Map Coded Font (MCF) Format 2” on page 213
- “Map Color Attribute Table (MCA)” on page 207
- “Map Page Overlay (MPO)” on page 265

Resource Section Number Triplet X'25'

The Resource Section Number triplet specifies a coded font section number. It may be used to select a single section of a double-byte coded font if less than the entire double-byte coded font is required for processing.

Triplet X'25' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'25'	Identifies the Resource Section Number triplet	M	X'00'
2	CODE	ResSNum	X'00'–X'FF'	Specifies the resource section number	M	X'06'

Triplet X'25' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Resource Section Number triplet.

ResSNum Specifies the resource section number. The valid resource section number values are determined by the encoding scheme used for the font. For fonts encoded using the EBCDIC Presentation double-byte encoding scheme (encoding scheme ID X'62*nn*') or the EBCDIC Presentation single-byte encoding scheme (encoding scheme ID X'61*nn*'), the valid resource section numbers are:

Value	Comments
X'00'	Must be used when this triplet references a single-byte coded font. Specifies all sections when this triplet references a double-byte coded font.
X'41'–X'FE'	Used only for double-byte coded fonts to select a specific font section
All others	Reserved

Notes:

1. If this triplet is omitted, the architected default value for the resource section number is X'00'.
2. The encoding scheme is specified by the Encoding Scheme ID triplet; see "Encoding Scheme ID Triplet X'50'" on page 371.

Structured Fields Using Triplet X'25'

- "Map Coded Font (MCF) Format 2" on page 213

Character Rotation Triplet X'26'

The Character Rotation triplet is used to specify character rotation relative to the character coordinate system. See the *Font Object Content Architecture Reference* for further information.

Triplet X'26' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'26'	Identifies the Character Rotation triplet	M	X'00'
2–3	CODE	CharRot	X'0000', X'2D00', X'5A00', X'8700'	Specifies the clockwise character rotation: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	M	X'06'

Triplet X'26' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Character Rotation triplet.

CharRot Specifies the clockwise character rotation relative to the character coordinate system. Valid values are the following:

Value	Character Rotation
X'0000'	0 degrees
X'2D00'	90 degrees
X'5A00'	180 degrees
X'8700'	270 degrees
All others	Reserved

Note: If this triplet is omitted, the architected default value for the character rotation is X'0000', zero degrees.

Structured Fields Using Triplet X'26'

- "Map Coded Font (MCF) Format 2" on page 213

Object Byte Offset Triplet X'2D'

The Object Byte Offset triplet is used to specify the byte offset of an indexed object within a document.

Triplet X'2D' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6, 10	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'2D'	Identifies the Object Byte Offset triplet	M	X'00'
2–5	UBIN	DirByOff	X'00000000'–X'FFFFFFFE'	Byte offset	M	X'06'
			X'FFFFFFFF'	If bytes 6–9 are not specified, object is outside document		
6–9	UBIN	DirByHi	X'00000000'–X'FFFFFFFF'	Byte offset, high-order bytes	O	X'00'

Triplet X'2D' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Byte Offset triplet.

DirByOff Specifies the offset, in bytes, of an indexed object from the beginning of the document. The Begin Document (BDT) structured field begins the document object and has an offset of 0. The first byte in the BDT is counted as byte 1 of the offset to objects that follow, so that if the BDT consists of *n* bytes, the offset to a Begin Object structured field that immediately follows the BDT is *n*. The byte offset has a range of X'00000000' to X'FFFFFFFE'. A value of X'FFFFFFFF' signifies that the indexed object is outside the document.

DirByHi If specified, indicates that this triplet specifies the byte offset as an 8-byte parameter, where DirByOff specifies the low-order 4 bytes and DirByHi specifies the high-order 4 bytes. In that case, the value DirByOff = X'FFFFFFFF' is a real offset value and does *not* signify that the indexed object is outside the document.

Structured Fields Using Triplet X'2D'

- “Index Element (IEL)” on page 176

Attribute Value Triplet X'36'

The Attribute Value triplet is used to specify a value for a document attribute.

Triplet X'36' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4–254	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'36'	Identifies the Attribute Value triplet	M	X'00'
2–3				Reserved; must be zero	M	X'06'
4–*n*	CHAR	AttVal		Attribute Value	O	X'00'

Triplet X'36' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Attribute Value triplet.

AttVal Is a character string which specifies the value of a document attribute. If this parameter is omitted, the value of the document attribute is specified to be null, that is, no value is assigned to the attribute.

Structured Fields Using Triplet X'36'

- “Tag Logical Element (TLE)” on page 310

Descriptor Position Triplet X'43'

The Descriptor Position triplet is used to associate an Object Area Position structured field with an Object Area Descriptor structured field.

Triplet X'43' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'43'	Identifies the Descriptor Position triplet	M	X'00'
2	CODE	DesPosID	X'01'–X'7F'	Specifies the associated Object Area Position structured field	M	X'06'

Triplet X'43' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Descriptor Position triplet.

DesPosID Specifies the identifier of the Object Area Position structured field that is associated with the descriptor for this object area.

Structured Fields Using Triplet X'43'

- “Object Area Descriptor (OBD)” on page 270

Media Eject Control Triplet X'45'

The Media Eject Control triplet is used to specify the type of media eject that is performed and the type of controls that are activated when a new medium map is invoked and N-up partitioning is specified.

Triplet X'45' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'45'	Identifies the Media Eject Control triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3	CODE	EjCtrl	X'01'–X'04'	Media eject controls: **X'01'** Eject to new sheet **X'02'** Conditional eject to next partition **X'03'** Conditional eject to next front-side partition **X'04'** Conditional eject to next back-side partition	M	X'06'

Triplet X'45' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Media Eject Control triplet.

EjCtrl Is a code that identifies the type of media eject that should be performed and the type of controls that should be activated when the medium map containing this triplet is invoked and N-up partitioning is specified. This triplet is ignored when it occurs on the medium map that is activated at the beginning of a document regardless of whether this medium map is explicitly invoked or implicitly invoked as the default. The following types of media eject can be specified:

- Eject to new sheet
- Conditional eject to next partition
- Conditional eject to next front-side partition
- Conditional eject to next back-side partition

The two types of controls that may be activated are medium-level controls and page-level controls. Media-level controls are controls that affect the medium, such as the specification of medium overlays, medium size, medium orientation, medium copies, N-up, simplex or duplex, medium finishing, media type, and media source and destination selection. They are defined by the Map Medium Overlay (MMO), Medium Descriptor (MDD), Medium Copy Count (MCC), Medium Finishing Control (MFC), Map Media Type (MMT), and Medium Modification Control (MMC) structured fields. Page-level controls are controls that affect the pages that are placed on the medium, such as the specification of page modifications, page position, and page orientation. They are defined by the Map Page Overlay (MPO), Page Position (PGP), and Page Modification Control (PMC) structured fields.

In the following descriptions, the term “existing PGP” refers to the Page Position (PGP) structured field that was active with the existing medium map, and the term “new PGP” refers to the PGP that is activated with the new medium map. The media-level controls in the new and existing medium maps are considered to be *identical* if and only if all of the following conditions are met:

- Any MMO, MDD, MCC, MFC with MFCScpe = X'04' (medium-map-level MFC, each sheet), MMT, and MMC structured field that appears in the existing medium map must also appear in the new medium map.
- The MMO, MDD, MCC, MFC with MFCScpe = X'04' (medium-map-level MFC, each sheet), MMT, and MMC structured fields that appear in both the new and existing medium maps must not only have the same functional content but also must have the same form. For example, if both medium maps contain an MMO structured field, the MMO repeating groups must map the same overlay names to the same local IDs, and the repeating groups must appear in the same order. Similarly, if both medium maps contain an MMC structured field, the MMC keywords must be the same, must specify the same values, and must appear in the same order.

Note that MFCs that start and continue medium-map-level sheet collections for finishing (MFCScpe = X'05') are excluded from the media-level-controls compare. These structured fields are processed and may cause a sheet eject based on their own processing rules. If processing such MFCs does not cause a sheet eject, the media-level-control compare determines whether or not a sheet eject is performed. Note also that a sheet eject is always generated after a finishing operation is applied to a collection of media or sheets.

The following values are supported for the EjCtrl parameter:

Value **Description**

X'01' Eject to new sheet. The new medium map is a complete replacement for the existing medium map and specifies the medium-level controls and page-level controls to be used to process the new sheet.

X'02' Conditional eject to next partition. This control is used with N-up partitioning. If N-up is not specified, or if the presentation device does not support N-up, this control is processed as X'01' (eject to new sheet). If the medium-level controls in the new medium map are not *identical* to the medium-level controls in the existing medium map, or if the page-level controls in the new medium map specify a different page placement than the page-level controls in the existing medium map, this control is processed as X'01' (eject to new sheet). If the medium-level controls in the new medium map are *identical* to the medium-level controls in the existing medium map, and if both medium maps specify default page placement or both specify explicit page placement, the page-level controls in the new medium map are activated and an eject to the next partition is performed. The location of the next partition is determined as follows:

- *Default page placement*: The next partition is the next sequential partition on the current sheet-side. If all partitions on the current sheet-side have been used, it is the first partition on the next sheet-side, which for simplex printing is always the front side of the next sheet, and for duplex printing is either the back side of the current sheet (if currently on a front side) or the front side of the next sheet (if currently on a back side).
- *Explicit page placement*: The next partition is defined by the repeating group in the new PGP that corresponds to the next repeating group that was to be processed in the existing PGP. If all PGP repeating groups have been processed, an implicit sheet eject is performed and processing continues with the first repeating group in the new PGP. For example, if the first repeating group in the existing PGP was last used to place a page, processing continues with the second repeating group in the new PGP.

  Note: The new PGP should place pages into the same partitions as the existing PGP. Otherwise, previously placed pages may be overwritten.

X'03' Conditional eject to next front-side partition. This control is used with N-up partitioning. If N-up is not specified, or if the presentation device does not support N-up, this control is processed as X'01' (eject to new sheet). If the medium-level controls in the new medium map are not *identical* to the medium-level controls in the existing medium map, or if the page-level controls in the new medium map specify a different page placement than the page-level controls in the existing medium map, this control is processed as X'01' (eject to new sheet). If the medium-level controls in the new medium map are *identical* to the medium-level controls in the existing medium map, and if both medium maps specify default page placement or both specify explicit page placement, the page-level controls in the new medium map are activated and an eject to the next front-side partition is performed. The location of the next front-side partition is determined as follows:

- *Default page placement:* If currently placing pages on the front sheet side, the next front-side partition is the next sequential partition. If all partitions on the front sheet-side have been used, an implicit sheet eject is performed and processing continues with the first partition on the front side of the next sheet. If currently placing pages on the back sheet side, an implicit sheet eject is performed and processing continues with the first partition on the front side of the next sheet.

- *Explicit page placement:* The next front-side partition is defined by the repeating group in the new PGP that corresponds to the next repeating group specifying front sheet-side that was to be processed in the existing PGP. If all PGP repeating groups that specify front sheet-side have been processed, an implicit sheet eject is performed and processing continues with the first repeating group in the new PGP that specifies front sheet-side. For example, if the first repeating group in the existing PGP was last used to place a page, and if the second repeating group specifies a back-side partition and the third repeating group specifies a front-side partition, processing continues with the third repeating group in the new PGP.

  Note: The new PGP should place pages into the same partitions as the existing PGP, otherwise previously-placed pages may be overwritten.

X'04' Conditional eject to next back-side partition. This control is used with N-up partitioning. If N-up is not specified, or if the presentation device does not support N-up, this control is processed as X'01' (eject to new sheet). If the medium-level controls in the new medium map are not *identical* to the medium-level controls in the existing medium map, or if the page-level controls in the new medium map specify a different page placement than the page-level controls in the existing medium map, this control is processed as X'01' (eject to new sheet). If the medium-level controls in the new medium map are *identical* to the medium-level controls in the existing medium map, and if both medium maps specify default page placement or both specify explicit page placement, the page-level controls in the new medium map are activated and an eject to the next back-side partition is performed. The location of the next back-side partition is determined as follows:

- *Default page placement:* If currently placing pages on the back sheet side, the next back-side partition is the next sequential partition. If all partitions on the back sheet-side have been used, an implicit sheet eject is performed and processing continues with the first partition on the back side of the next sheet. If currently placing pages on the front sheet-side, processing continues with the first partition on the back sheet-side.
- *Explicit page placement:* The next back-side partition is defined by the repeating group in the new PGP that corresponds to the next repeating group specifying back sheet-side that was to be processed in the existing PGP. If all PGP repeating groups that specify back sheet-side have been processed, an implicit sheet eject is performed and processing continues with the first repeating group in the new PGP that specifies back sheet-side. For example, if the first repeating group in the existing PGP was last used to place a page, and if the second and third repeating groups specify front-side partitions and

the fourth repeating group specifies a back-side partition, processing continues with the fourth repeating group in the new PGP.

Note: The new PGP should place pages into the same partitions as the existing PGP, otherwise previously-placed pages may be overwritten.

All others
Reserved

Note: If this triplet is not specified, the architected default for the EjCtrl parameter is X'01', that is perform a sheet eject and activate all controls specified by the invoked medium map.

Structured Fields Using Triplet X'45'

- "Begin Medium Map (BMM)" on page 122

Page Overlay Conditional Processing Triplet X'46'

The Page Overlay Conditional Processing triplet is used to identify the intended utilization of a page overlay as produced by a generator. This triplet can also be used to define an overlay level that determines whether the overlay is to be processed.

Triplet X'46' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3–4	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'46'	Identifies the Page Overlay Conditional Processing triplet	M	X'00'
2	CODE	PgOvType	X'01'–X'03'	Specifies the page overlay type: **X'00'** Type 0: Normal **X'01'** Type 1: Annotation **X'02'** Type 2: Redaction **X'03'** Type 3: Highlight	M	X'06'
3	CODE	Level	X'01'–X'FE'	The level of the overlay	O	X'02'

Triplet X'46' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Page Overlay Conditional Processing triplet.

PgOvType Specifies the intended use of the overlay. If this parameter contains a value that is not supported by the receiver, the overlay is not processed.

The page overlay types are defined as follows:

Type	Description
Type 0	Normal page overlay.
Type 1	Annotation overlay. Type 1 indicates that the page overlay is an annotation overlay used to indicate changes or annotations to the contents of the page to which it applies.
Type 2	Redaction overlay. Type 2 indicates that the page overlay is a redaction overlay used to mask or hide all or a portion of the page to which it applies.
Type 3	Highlight overlay. Type 3 indicates that the page overlay is a highlight overlay used to highlight all or a portion of the page to which it applies.

Level Specifies the processing level of the overlay. An overlay level is used to determine whether the overlay is to be processed by a particular application.

Value	Description
X'01'–X'FE'	Level

All others Reserved

Note: Should the optional *Level* value be omitted, the architected default is X'01'.

Overlay Type Conditional Processing

Conditional processing is applied to the overlay types as follows:

Type **Conditional Processing Description**

Type 0 No conditional processing is applied. If a level value was specified, it is ignored, and the page overlay is processed normally.

Type 1 The overlay level is matched against one contained within the application, and if it is equal to or lower than the application's level it is processed. Should the level be higher than the level contained in the application, or if the application does not contain a level, overlay processing is not performed.

Type 2 The overlay level is matched against one contained within the application, and if it is higher than the application's level, or if the application does not contain a level, it is processed. If the level be equal to or lower than the level contained in the application, overlay processing is not performed.

Type 3 If the receiver is enabled to present highlighted areas, the overlay is processed. If the receiver is not enabled to present highlighted areas, the overlay is not processed. The enablement is achieved external to the data stream. The overlay level is not used with highlight overlays. If a level is specified, it is ignored.

Architecture Note: In general, the highlighting effect is achieved by including a colored highlight overlay on a page using a specified set of mixing rules. When a presentation device does not support the functions necessary to present the specified highlighting, as in the case of a bi-level device, it may choose to default to a highlighting implementation where the area defined by the highlight overlay is presented in reverse video.

Note: If this triplet is omitted, the architected default value for *PgOvType* is X'00', *Type 0*, which indicates that the page overlay is always processed.

Structured Fields Using Triplet X'46'

- "Include Page Overlay (IPO)" on page 194
- "Map Page Overlay (MPO)" on page 265

Resource Usage Attribute Triplet X'47'

The Resource Usage Attribute triplet can be used for resource management. It is used with the Include Page Overlay and Map Page Overlay structured fields to identify the approximate frequency with which an associated page overlay is processed. This is indicated by assigning either a *low* or *high* value to this triplet. The Resource Usage Attribute triplet has no processing semantics associated with it.

Triplet X'47' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'47'	Identifies the Resource Usage Attribute triplet	M	X'00'
2	CODE	Frequency	X'00', X'FF'	Frequency of use: **X'00'** Low **X'FF'** High	M	X'06'

Triplet X'47' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Resource Usage Attribute triplet.

Frequency Specifies the processing frequency of the associated page overlay. The valid values are:

Value	Description
X'00'	Low
X'FF'	High
All others	Reserved

Structured Fields Using Triplet X'47'

- “Include Page Overlay (IPO)” on page 194
- “Map Page Overlay (MPO)” on page 265

Measurement Units Triplet X'4B'

The Measurement Units triplet is used to specify the units of measure for a presentation space.

Triplet X'4B' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	8	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'4B'	Identifies the Measurement Units triplet	M	X'00'
2	CODE	XoaBase	X'00'–X'01'	Presentation space unit base for the X axis: **X'00'** 10 inches **X'01'** 10 centimeters	M	X'06'
3	CODE	YoaBase	X'00'–X'01'	Presentation space unit base for the Y axis: **X'00'** 10 inches **X'01'** 10 centimeters	M	X'06'
4–5	UBIN	XoaUnits	1–32767	Presentation space units per unit base for the X axis	M	X'06'
6–7	UBIN	YoaUnits	1–32767	Presentation space units per unit base for the Y axis	M	X'06'

Triplet X'4B' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Measurement Units triplet.

XoaBase Specifies the unit base for the X axis of the presentation space coordinate system.

YoaBase Specifies the unit base for the Y axis of the presentation space coordinate system.

Note: A X’01’ exception condition exists if the XoaBase and YoaBase values are not identical.

XoaUnits Specifies the number of units per unit base for the X axis of the presentation space coordinate system.

YoaUnits Specifies the number of units per unit base for the Y axis of the presentation space coordinate system.

Structured Fields Using Triplet X'4B'

- “Include Object (IOB)” on page 180
- “Link Logical Element (LLE)” on page 199
- “Object Area Descriptor (OBD)” on page 270
- “Page Modification Control (PMC)” on page 297
- “Preprocess Presentation Object (PPO)” on page 299

Object Area Size Triplet X'4C'

The Object Area Size triplet is used to specify the extent of an object area in the X and Y directions.

Triplet X'4C' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	9	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'4C'	Identifies the Object Area Size triplet	M	X'00'
2	CODE	SizeType	X'02'	Specifies the actual object area size to be used	M	X'06'
3–5	UBIN	XoaSize	1–32767	Object area extent for the X axis	M	X'06'
6–8	UBIN	YoaSize	1–32767	Object area extent for the Y axis	M	X'06'

Triplet X'4C' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Area Size triplet.

SizeType Specifies the object area size type.

Value	Description
X'02'	Object Area Size
All others	Reserved

XoaSize Specifies the extent of the X axis of the object area coordinate system. This is also known as the object area’s X axis size.

YoaSize Specifies the extent of the Y axis of the object area coordinate system. This is also known as the object area’s Y axis size.

Structured Fields Using Triplet X'4C'

- “Include Object (IOB)” on page 180
- “Object Area Descriptor (OBD)” on page 270
- “Preprocess Presentation Object (PPO)” on page 299

Area Definition Triplet X'4D'

The Area Definition triplet is used to define the position and size of a rectangular area on a document component presentation space. The document component may be a page or overlay, in which case the area is defined on the page or overlay presentation space, or it may be a data object, in which case the area is defined on the object area presentation space.

Triplet X'4D' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	15	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'4D'	Identifies the Area Definition triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3–5	SBIN	XarOset	0–32767	X-axis origin of the area	M	X'06'
6–8	SBIN	YarOset	0–32767	Y-axis origin of the area	M	X'06'
9–11	UBIN	XarSize	1–32767	Area extent for the X axis	M	X'06'
12–14	UBIN	YarSize	1–32767	Area extent for the Y axis	M	X'06'

Triplet X'4D' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Area Definition triplet

XarOset Specifies the offset along the X axis of the presentation space coordinate system to the origin of the area.

YarOset Specifies the offset along the Y axis of the presentation space coordinate system to the origin of the area.

XarSize Specifies the extent of the area along the X axis of the presentation space coordinate system.

YarSize Specifies the extent of the area along the Y axis of the presentation space coordinate system.

Structured Fields Using Triplet X'4D'

- “Link Logical Element (LLE)” on page 199

Color Specification Triplet X'4E'

The Color Specification triplet is used to specify a color value and defines the color space and encoding for that value.

Triplet X'4E' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	14–16	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'4E'	Identifies the Color Specification triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3	CODE	ColSpce	X'01', X'04', X'06', X'08', X'40'	Color space: **X'01'** RGB **X'04'** CMYK **X'06'** Highlight color space **X'08'** CIELAB **X'40'** Standard OCA color space	M	X'06'
4–7				Reserved; must be zero	M	X'06'
8	UBIN	ColSize1	X'01'–X'08', X'10'	Number of bits in component 1; see color space definitions	M	X'06'
9	UBIN	ColSize2	X'00'–X'08'	Number of bits in component 2; see color space definitions	M	X'06'
10	UBIN	ColSize3	X'00'–X'08'	Number of bits in component 3; see color space definitions	M	X'06'
11	UBIN	ColSize4	X'00'–X'08'	Number of bits in component 4; see color space definitions	M	X'06'
12–*n*		Color		Color specification; see “Triplet X'4E' Semantics” for details	M	X'06'

Triplet X'4E' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Color Specification triplet.

ColSpce Is a code that defines the color space and the encoding for the color specification.

Value **Description**

X'01' RGB color space. The color value is specified with three components. Components 1, 2, and 3 are unsigned binary numbers that specify the red, green, and blue intensity values, in that order. ColSize1, ColSize2, and ColSize3 are non-zero and define the number of bits used to specify each component. ColSize4 is reserved and should be set to zero. The intensity range for the R,G,B components is 0 to 1, which is mapped to the binary value range 0 to (2^{ColSizeN} – 1), where N=1,2,3.

Architecture Note: The reference white point and the chromaticity coordinates for RGB are defined in SMPTE RP 145-1987, entitled *Color Monitor Colorimetry*, and in RP 37-1969, entitled *Color Temperature for Color Television Studio Monitors*, respectively. The reference white point is commonly known as *Illuminant* D_{6500} or simply *D65*. The R,G,B components are assumed to be gamma-corrected (non-linear) with a gamma of 2.2.

X'04' CMYK color space. The color value is specified with four components. Components 1, 2, 3, and 4 are unsigned binary numbers that specify the cyan, magenta, yellow, and black intensity values, in that order. ColSize1, ColSize2, ColSize3, and ColSize4 are non-zero and define the number of bits used to specify each component. The intensity range for the C,M,Y,K components is 0 to 1, which is mapped to the binary value range 0 to ($2^{\text{ColSize}N}$ – 1), where *N*=1,2,3,4. This is a device-dependent color space.

X'06' Highlight color space. This color space defines a request for the presentation device to generate a highlight color. The color value is specified with one to three components.

Component 1 is a two-byte unsigned binary number that specifies the highlight color number. The first highlight color is assigned X'0001', the second highlight color is assigned X'0002', and so on. The value X'0000'. specifies the presentation device default color. ColSize1 = X'10' and defines the number of bits used to specify component 1.

Component 2 is an optional one-byte unsigned binary number that specifies a percent coverage for the specified color. Percent coverage can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If the coverage is less than 100%, the remaining coverage is achieved with color of medium. ColSize2 = X'00' or X'08' and defines the number of bits used to specify component 2. A value of X'00' indicates that component 2 is not specified in the color value, in which case the architected default for percent coverage is 100%. A value of X'08' indicates that component 2 is specified in the color value.

Component 3 is an optional one-byte unsigned binary number that specifies a percent shading, which is a percentage of black that is to be added to the specified color. Percent shading can be any value from 0% to 100% (X'00'–X'64'). The number of distinct values supported is presentation-device dependent. If percent coverage and percent shading are specified, the effective range for percent shading is 0% to (100-coverage)%. If the sum of percent coverage plus percent shading is less than 100%, the remaining coverage is achieved with color of medium. ColSize3 = X'00' or X'08' and defines the number of bits used to specify component 3. A value of X'00' indicates that component 3 is not specified in the color value, in which case the architected default for percent shading is 0%. A value of X'08' indicates that component 3 is specified in the color value.

Implementation Note: The percent shading parameter is currently not supported in AFP environments.

ColSize4 is reserved and should be set to zero. This is a device-dependent color space.

Architecture Notes:

1. The color that is rendered when a highlight color is specified is device-dependent. For presentation devices that support colors other than black, highlight color values in the range X'0001' to X'FFFF' may be mapped to any color. For bi-level devices, the color may be simulated with a graphic pattern.
2. If the specified highlight color is “presentation device default”, devices whose default color is black use the percent coverage parameter, which is specified in component 2, to render a percent shading.
3. On printing devices, the color of medium is normally white, in which case a coverage of *n*% results in adding (100−*n*)% white to the specified color, or *tinting* the color with (100−*n*)% white. Display devices may assume the color of medium to always be white and use this algorithm to render the specified coverage.
4. The highlight color space can also specify indexed colors when used in conjunction with a Color Mapping Table (CMT). In that case, component 1 specifies a two-byte value that is an index into the CMT, and components 2 and 3 are not specified in the color value. For a description of the CMT, see “The Color Mapping Table Resource” on page 476.

X'08' CIELAB color space. The color value is specified with three components. Components 1, 2, and 3 are binary numbers that specify the L, a, b values, in that order, where L is the luminance and a and b are the chrominance differences. Component 1 specifies the L value as an unsigned binary number; components 2 and 3 specify the a and b values as signed binary numbers. ColSize1, ColSize2, and ColSize3 are non-zero and define the number of bits used to specify each component. ColSize4 is reserved and should be set to zero. The range for the L component is 0 to 100, which is mapped to the binary value range 0 to (2^{ColSize1} − 1). The range for the a and b components is −127 to +127, which is mapped to the binary range −($2^{\text{ColSize}N-}$ − 1) to +($2^{\text{ColSize}N-}$ − 1).

For color fidelity, 8-bit encoding should be used for each component, that is, ColSize1, ColSize2, and ColSize3 are set to X'08'. When the recommended 8-bit encoding is used for the a and b components, the range is extended to include

−128, which is mapped to the value X'80'. If the encoding is less than 8 bits, treatment of the most negative binary endpoint for the a and b components is device-dependent, and tends to be insignificant because of the quantization error.

Architecture Note: The reference white point for CIELAB is known as *D50* and is defined in CIE publication 15-2 entitled *Colorimetry*.

X'40' Standard OCA color space. The color value is specified with one component. Component 1 is an unsigned binary number that specifies a named color using a two-byte value from the Standard OCA Color Value Table. For a complete description of the Standard OCA Color Value Table, see “Standard OCA Color Value Table” on page 473. ColSize1 = X'10' and defines the number of bits used to specify component 1. ColSize2, ColSize3, ColSize4 are reserved and should be set to zero. This is a device-dependent color space.

All others
Reserved

ColSize1 Defines the number of bits used to specify the first color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary. For example, if ColSize1 = X'06', the first color component has two padding bits.

ColSize2 Defines the number of bits used to specify the second color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

ColSize3 Defines the number of bits used to specify the third color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

ColSize4 Defines the number of bits used to specify the fourth color component. The color component is right-aligned and padded with zeros on the left to the nearest byte boundary.

Color Specifies the color value in the defined format and encoding. Note that the number of bytes specified for this parameter depends on the color space. For example, when using 8 bits per component, an RGB color value is specified with 3 bytes, while a CMYK color value is specified with 4 bytes. If extra bytes are specified, they are ignored as long as the triplet length is valid.

Architecture Note: For a description of color spaces and their relationships, see R. Hunt, *The Reproduction of Colour in Photography, Printing, and Television* (Fifth Edition, Fountain Press, 1995).

Structured Fields Using Triplet X'4E'

- “Include Object (IOB)” on page 180
- “Object Area Descriptor (OBD)” on page 270
- “Page Descriptor (PGD)” on page 279

Encoding Scheme ID Triplet X'50'

The Encoding Scheme ID triplet is used to specify the encoding scheme associated with a code page. It may optionally also specify the encoding scheme for the user data.

Triplet X'50' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4, 6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'50'	Identifies the Encoding Scheme ID triplet	M	X'00'
2–3	CODE	ESidCP	See “Triplet X'50' Semantics”	Encoding Scheme Identifier for Code Page	M	X'06'
4–5	CODE	ESidUD	See “Triplet X'50' Semantics”	Encoding Scheme Identifier for User Data	O	X'00'

Triplet X'50' Semantics

Architecture Note: The encoding scheme defined in this triplet is based on the encoding scheme identifier defined by the IBM Character Data Representation Architecture (CDRA). However, only those values applicable to MO:DCA environments are exposed. The remainder of the values are reserved at this time. Note also that the bit definitions for the ESidCP and ESidUD parameters are informational; the codes defined in Table 22 on page 373, Table 23 on page 373, and Table 24 on page 373 should be used as the valid parameter values. See the *Character Data Representation Architecture Reference and Registry*, SC09-2190, for detailed information on the encoding scheme identifier.

Tlength Contains the length of the triplet.

Tid Identifies the Encoding Scheme ID triplet.

ESidCP Specifies the encoding scheme used for a code page.

Note: See the appropriate structured field descriptions for definitions of the default code page encoding if this triplet is omitted.

Bit **Description**

0–3 Basic Encoding Structure

X'0' Encoding structure not specified. Defaults to presentation environment encoding structure.

X'2' IBM-PC Data; an extension of the ISO 646 (ASCII-based) 7-bit encoding to an 8-bit encoding.

X'3' IBM-PC Display; an extension of the ISO 646 (ASCII-based) 7-bit encoding to an 8-bit encoding.

Implementation Note: The IBM-PC Display encoding scheme is not used in AFP FOCA fonts.

X'6' EBCDIC Presentation; all code points assigned to graphic characters.

X'7' UTF-16, including surrogates.

Architecture Note: The UTF-16 character encoding is defined in the Unicode Standard, which is available from the Unicode Consortium at *http://www.unicode.org*.

X'8' Unicode Presentation; a subset of UTF-16 that contains only 2-byte code points that can be directly mapped to a single glyph. The byte order is big endian.

Implementation Note: The Unicode Presentation encoding scheme is only used in the AFP FOCA Unicode Migration fonts.

All others
Reserved

4–7 Number of Bytes per Code Point

X'0'	Reserved for use with zero value for the basic encoding structure
X'1'	Fixed single-byte
X'2'	Fixed double-byte
All others	Reserved

8–15 Code Extension Method
X'00' No extensions are specified

ESidUD Specifies the encoding scheme for the user data that is to be rendered with the referenced font.

Note: See the appropriate structured field descriptions for definitions of the default user data encoding if this parameter in the X'50' triplet is omitted or if the complete X'50' triplet is omitted.

Bit **Description**

0–3 Basic Encoding Structure

X'7'	UTF-16, including surrogates. The byte order is big endian (UTF-16BE).
All others	Reserved

4–7 Number of Bytes per Code Point

X'2'	Fixed double-byte
X'8'	UTF-n variable number of bytes, self describing
All others	Reserved

8–15 Code Extension Method

X'00'	No extensions are specified
X'07'	UTF-8 Universal Transformation Format
All others	Reserved

Architecture Note: The UTF-8 character encoding is defined in the Unicode Standard Version 3.2, which is available from the Unicode Consortium at

http://www.unicode.org

.

Table 22 and Table 23 list the complete ESidCP and ESidUD values that are supported.

Table 22. Supported ESidCP Values

ESidCP	Definition
X'0000'	ESidCP not specified; use presentation environment default encoding
X'0100'	Presentation environment default SBCS encoding
X'0200'	Presentation environment default DBCS encoding
X'2100'	PC-Data SBCS (ASCII-based)
X'3100'	PC-Display SBCS (ASCII-based)
X'6100'	EBCDIC Presentation SBCS
X'6200'	EBCDIC Presentation DBCS
X'7200'	UTF-16, including surrogates
X'8200'	Unicode Presentation; byte order is big endian

Table 23. Supported ESidUD Values

ESidUD	Definition
X'7200'	UTF-16, including surrogates; byte order is big endian (UTF-16BE)
X'7807'	UTF-8

Application Note: When ESidUD does not match ESidCP, the presentation system may need to transform the user data to match the encoding in the code page. Not all presentation systems support such transforms. To see which transforms are supported, consult your product documentation.

Architecture Note: The following additional ESidUD values are allowed in AFP Line Data when the X'50' triplet is specified on the Begin Data Map (BDM) structured field in a Page Definition.

Table 24. Additional ESidUD Values in AFP Line Data

ESidUD	Definition
X'2100'	PC-Data SBCS (ASCII-based)
X'6100'	EBCDIC Presentation SBCS

Structured Fields Using Triplet X'50'

- “Map Coded Font (MCF) Format 2” on page 213
- “Map Data Resource (MDR)” on page 223

Medium Map Page Number Triplet X'56'

The Medium Map Page Number triplet is used to specify the sequence number of the page in the set of sequential pages whose presentation is controlled by the most recently activated medium map.

Triplet X'56' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'56'	Identifies the Medium Map Page Number triplet	M	X'00'
2–5	UBIN	PageNum	X'00000001'–X'7FFFFFFF'	Sequence Number of Page	M	X'06'

Triplet X'56' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Medium Map Page Number triplet.

PageNum Specifies the sequence number of the page in the set of sequential pages whose presentation is controlled by the active medium map. The first page in this set has sequence number 1.

Structured Fields Using Triplet X'56'

- "Begin Named Page Group (BNG)" on page 126
- "Begin Page (BPG)" on page 134
- "Index Element (IEL)" on page 176

Object Byte Extent Triplet X'57'

The Object Byte Extent triplet is used to specify the number of bytes contained in an object.

Triplet X'57' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6, 10	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'57'	Identifies the Object Byte Extent triplet	M	X'00'
2–5	UBIN	ByteExt	X'00000000'–X'FFFFFFFF'	Byte Extent of Object	M	X'06'
6–9	UBIN	BytExtHi	X'00000000'–X'FFFFFFFF'	Byte extent of object, high-order bytes	M	X'06'

Triplet X'57' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Byte Extent triplet.

ByteExt Specifies the number of bytes contained in the object. The first byte of the Begin Object structured field is counted as the first byte in the object, and the last byte in the End Object structured field is counted as the last byte of the object. Objects that are bounded by Begin/End structured fields have a minimum byte extent of X'00000010'. When this triplet is used to specify the byte extent of object data that is not bounded by Begin/End structured fields, the minimum byte extent is X'00000000'.

BytExtHi If specified, indicates that this triplet specifies the byte extent as an 8-byte parameter, where ByteExt specifies the low-order 4 bytes and BytExtHi specifies the high-order 4 bytes.

Structured Fields Using Triplet X'57'

- “Begin Object Container (BOC)” on page 129
- “Index Element (IEL)” on page 176

Object Structured Field Offset Triplet X'58'

The Object Structured Field Offset triplet is used to specify the structured field offset of an indexed object from the beginning of the document.

Triplet X'58' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6, 10	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'58'	Identifies the Object Structured Field Offset triplet	M	X'00'
2–5	UBIN	SFOff	X'00000000'–X'FFFFFFFE'	Structured field offset	M	X'06'
			X'FFFFFFFF'	If bytes 6–9 are not specified, object is outside document		
6–9	UBIN	SFOffHi	X'00000000'–X'FFFFFFFF'	Structured field offset, high-order bytes	O	X'00'

Triplet X'58' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Structured Field Offset triplet.

SFOff Specifies the offset, in structured fields, of the Begin structured field of an indexed object from the beginning of the document. The first structured field in the document, which is the Begin Document (BDT) structured field, has an offset of 0. The second structured field, which immediately follows the BDT, has an offset of 1, and the *n*th structured field in the document has an offset of (*n*–1). The structured field offset has a range of X'00000000' to X'FFFFFFFE'. A value of X'FFFFFFFF' signifies that the indexed object is outside the document.

SFOffHi If specified, indicates that this triplet specifies the structured field offset as an 8-byte parameter, where SFOff specifies the low-order 4 bytes and SFOffHi specifies the high-order 4 bytes. In that case, the value SFOff = X'FFFFFFFF' is a real offset value and does *not* signify that the indexed object is outside the document.

Structured Fields Using Triplet X'58'

- “Index Element (IEL)” on page 176

Object Structured Field Extent Triplet X'59'

The Object Structured Field Extent triplet is used to specify the number of structured fields contained in an object, starting with the Begin Object structured field and ending with the End Object structured field.

Triplet X'59' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6, 10	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'59'	Identifies the Object Structured Field Extent triplet	M	X'00'
2–5	UBIN	SFExt	X'00000002'–X'FFFFFFFF'	Number of structured fields in Object	M	X'06'
6–9	UBIN	SFExtHi	X'00000000'–X'FFFFFFFF'	Number of structured fields in object, high-order bytes	O	X'00'

Triplet X'59' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Structured Field Extent triplet.

SFExt Specifies the number of structured fields contained in the object. The Begin Object structured field is counted as the first structured field in the object, and the End Object structured field is counted as the last structured field of the object.

SFExtHi If specified, indicates that this triplet specifies the structured field extent as an 8-byte parameter, where SFExt specifies the low-order 4 bytes and SFExtHi specifies the high-order 4 bytes.

Structured Fields Using Triplet X'59'

- “Index Element (IEL)” on page 176

Object Offset Triplet X'5A'

The Object Offset triplet specifies the number of objects of a particular type that precede a selected object in the document. If the object being counted is a document, this triplet specifies the number of documents that precede the selected object in the print file.

Triplet X'5A' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	8, 12	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'5A'	Identifies the Object Offset triplet	M	X'00'
2	CODE	ObjTpe	X'A8', X'AF'	Object type to be counted: **X'A8'** Document **X'AF'** Page	M	X'06'
3				Reserved; must be zero	M	X'06'
4–7	UBIN	ObjOset	X'00000000'–X'FFFFFFFF'	Number of objects that precede the selected object in the document or print file	M	X'06'
8–11	UBIN	ObjOstHi	X'00000000'–X'FFFFFFFF'	Number of objects that precede the selected object, high-order bytes	O	X'00'

Triplet X'5A' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Offset triplet.

ObjTpe Specifies the object type to be counted. An object may occur at multiple levels. For instance, a page object may occur directly in a document, which would be considered a first-level occurrence of the page object, or it may occur in a page group in the document, which would be considered a second-level occurrence of the page object, and so on.

Value	Description
X'A8'	The object is a document. The ObjOset and optional ObjOstHi parameters specify the number of documents that precede the selected object in the print file.
X'AF'	The object is a page. The ObjOset and optional ObjOstHi parameters specify the number of pages that precede the selected object in the document.

Note: If a page is included with an Include Page (IPG) structured field in document state or page-group state, it is counted as a page object. If the IPG occurs in page state, the included page becomes part of the containing page, therefore only the containing page is counted as a page object.

All others Reserved

ObjOset Specifies the number of objects, whose type is identified by ObjTpe, that precede the selected object. Only complete objects, that is, objects bounded by a Begin and an End, are counted. For example, if this triplet occurs on the BNG of a nested page group Gn, the page group containing Gn is not counted since its End structured field does not precede Gn. For a given object type being counted, the offset to the *n*th occurrence of that object type is (*n*−1). For example, if pages are being counted, the page offset of the first page in the document is 0, the page offset of the second page is 1, and the page offset of the *n*th page is (*n*−1). A page included with an IPG is also counted, but only when the IPG occurs in document state or page-group state, not when it occurs in page state. Unless otherwise specified, all complete object occurrences at all levels are counted.

ObjOstHi If specified, indicates that this triplet specifies the number of preceding objects as an 8-byte parameter, where ObjOset specifies the low-order 4 bytes and ObjOstHi specifies the high-order 4 bytes.

Structured Fields Using Triplet X'5A'

- "Index Element (IEL)" on page 176
- "Include Page (IPG)" on page 191
- "Medium Finishing Control (MFC)" on page 236

Font Horizontal Scale Factor Triplet X'5D'

The Font Horizontal Scale Factor triplet is used to carry information to support anamorphic scaling of an outline technology font.

Triplet X'5D' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'5D'	Identifies the Font Horizontal Scale Factor triplet	M	X'00'
2–3	UBIN	Hscale	1–32767	Specifies the horizontal scale factor in 1440ths of an inch	M	X'06'

Triplet X'5D' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Font Horizontal Scale Factor triplet.

Hscale Specifies the horizontal scale factor that is to be applied to the horizontal font dimension when scaling an outline technology font. This scale factor is specified in 1440ths of an inch. If the font horizontal scale factor is the same as the specified vertical font size, the font scaling is uniform. If the font horizontal scale factor is not the same as the specified vertical font size, the font scaling is anamorphic, and the graphic characters are stretched or compressed in the horizontal direction relative to the vertical direction by the ratio of font horizontal scale factor divided by the specified vertical font size.

Structured Fields Using Triplet X'5D'

- “Map Coded Font (MCF) Format 2” on page 213

Object Count Triplet X'5E'

The Object Count triplet specifies the number of subordinate objects of a particular type contained in an object.

Triplet X'5E' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	8, 12	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'5E'	Identifies the Object Count triplet	M	X'00'
2	CODE	SubObj	X'AF'	Subordinate object type: **X'AF'** Page	M	X'04'
3				Reserved; must be zero	M	X'06'
4–7	UBIN	SObjNum	X'00000000'– X'FFFFFFFF'	Number of subordinate objects contained in this object	M	X'06'
8–11	UBIN	SObjNmHi	X'00000000'– X'FFFFFFFF'	Number of subordinate objects, high-order bytes	O	X'00'

Triplet X'5E' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Object Count triplet.

SubObj Specifies the subordinate object type. A subordinate object may occur at multiple levels within an object. For instance, a page object may occur directly in a page group, which would be considered a first-level occurrence of the subordinate object, or it may occur in a page group that is nested in the first page group, which would be considered a second-level occurrence of the subordinate object, and so on.

Value	Description
X'AF'	The subordinate object is a page. The SObjNum and optional SObjNmHi parameters specify the number of pages contained in the object. **Note:** If a page is included with an Include Page (IPG) structured field in document state or page-group state, it is counted as a page object. If the IPG occurs in page state, the included page becomes part of the containing page, therefore only the containing page is counted as a page object.
All others	Reserved

SObjNum Specifies the number of subordinate objects, whose type is identified by SubObj, that are contained in this object. Unless otherwise specified, all subordinate-object occurrences at all levels are counted.

SObjNmHi If specified, indicates that this triplet specifies the count of subordinate objects as an 8-byte parameter, where SObjNum specifies the low-order 4 bytes and SObjNmHi specifies the high-order 4 bytes.

Structured Fields Using Triplet X'5E'

- “Begin Named Page Group (BNG)” on page 126
- “Index Element (IEL)” on page 176

Local Date and Time Stamp Triplet X'62'

The Local Date and Time Stamp triplet specifies a date and time stamp to be associated with an object.

Triplet X'62' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	17	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'62'	Identifies the Local Date and Time Stamp triplet	M	X'00'
2	CODE	StampType	X'00'–X'01', X'03'	Specifies the date and time stamp type: **X'00'** Creation **X'01'** Retired value **X'03'** Revision	M	X'06'
3	CODE	THunYear	X'40', X'F0'–X'F9'	Hundreds position and implied thousands position of year AD: **X'40'** 19*xx* **X'F0'–X'F9'** 20*xx*–29*xx*	M	X'06'
4–5	CODE	TenYear	X'F0F0'–X'F9F9'	Tens and units position of year AD	M	X'06'
6–8	CODE	Day	X'F0F0F1'–X'F3F6F6'	Day of year	M	X'06'
9–10	CODE	Hour	X'F0F0'–X'F2F3'	Hour of day	M	X'06'
11–12	CODE	Minute	X'F0F0'–X'F5F9'	Minute of hour	M	X'06'
13–14	CODE	Second	X'F0F0'–X'F5F9'	Second of minute	M	X'06'
15–16	CODE	HundSec	X'F0F0'–X'F9F9'	Hundredth of second	M	X'06'

Triplet X'62' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Local Date and Time Stamp triplet.

StampType Specifies the type of date and time stamp.

Value	Description
X'00'	Object creation date and time stamp
X'01'	Retired date and time stamp type. See “Retired Parameters” on page 515.
X'03'	Object revision date and time stamp
All others	Reserved

THunYear Implies the thousands position (the millenium) of the year AD and specifies the hundreds position, using the Gregorian calendar. The 20*xx*s are encoded as X'F0', the 21*xx*s as X'F1', the 22*xx*s as X'F2', and so on. To differentiate the 19*xx*s (9*xx*s in the second millenium AD) from the 29*xx*s (9*xx*s in the third millenium AD), the 19*xx*s are encoded as X'40'. This parameter therefore generates the CC component of a date in the format *CCYYDDD* as defined in ISO

8601:1988(E), *Data elements and interchange formats—Information Interchange—Representation of dates and times*.

TenYear Specifies the tens position and the units position of the year AD, using the Gregorian calendar. Forms the *YY* component of a date in the format *CCYYDDD*.

This parameter, together with the ThunYear parameter, specifies the year AD. For example, the year 1999 AD is encoded as X'40F9F9', the year 2000 AD is encoded as X'F0F0F0', and the year 2001 AD is encoded as X'F0F0F1'.

Day Specifies the day of the year, using the Gregorian calendar. Forms the *DDD* component of a date in the format *CCYYDDD*.

As an example, the date February 1, 1972 is restructured as "72032" and encoded as X'40F7F2F0F3F2', the date December 31, 1999 is restructured as "99365" and encoded as X'40F9F9F3F6F5', the date January 1, 2000 is restructured as "000001" and encoded as X'F0F0F0F0F0F1', and the date February 3, 2072 is restructured as "072034" and encoded as X'F0F7F2F0F3F4'.

Hour Specifies the hour of the day. Forms the *HH* component of a timestamp in the format *HHMMSShh*.

Minute Specifies the minute of the hour. Forms the *MM* component of a timestamp in the format *HHMMSShh*.

Second Specifies the second of the minute. Forms the *SS* component of a timestamp in the format *HHMMSShh*.

HundSec Specifies hundredth of a second. Forms the *hh* component of a timestamp in the format *HHMMSShh*.

As an example, the time 4:35:21.56 PM is encoded as X'F1F6F3F5F2F1F5F6'.

Architecture Note: This triplet specifies an EBCDIC encoding for numbers used to record date and time. This encoding represents a number in the range 0–9 with a code point X'F*n*', where *n* is the number.

Application Note: This triplet is also used on the following private font object structured fields in AFP environments:
- Begin Code Page (BCP)
- Begin Font Character Set (BFN)

Structured Fields Using Triplet X'62'

Either this triplet or the Universal Date and Time Stamp (X'72') triplet may occur once.
- "Begin Bar Code Object (BBC)" on page 105
- "Begin Document Index (BDI)" on page 112
- "Begin Form Map (BFM)" on page 116
- "Begin Graphics Object (BGR)" on page 118
- "Begin Image Object (BIM)" on page 120
- "Begin Overlay (BMO)" on page 124
- "Begin Object Container (BOC)" on page 129
- "Begin Page Segment (BPS)" on page 137
- "Begin Presentation Text Object (BPT)" on page 139
- "Begin Resource Group (BRG)" on page 141

Comment Triplet X'65'

The Comment triplet is used to include comments for documentation purposes within a structured field.

Triplet X'65' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3–254	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'65'	Identifies the Comment triplet	M	X'00'
2–*n*	CHAR	Comment		Text of the comment	M	X'06'

Triplet X'65' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Comment triplet.

Comment Is a character string which has meaning only to the generator of this MO:DCA document. There can be no semantics associated with this character string. Therefore, the content of the triplet may be ignored by receivers of the MO:DCA document.

Structured Fields Using Triplet X'65'

- "Begin Active Environment Group (BAG)" on page 104
- "Begin Bar Code Object (BBC)" on page 105
- "Begin Color Attribute Table (BCA)" on page 107
- "Begin Document Environment Group (BDG)" on page 111
- "Begin Document Index (BDI)" on page 112
- "Begin Document (BDT)" on page 114
- "Begin Form Map (BFM)" on page 116
- "Begin Graphics Object (BGR)" on page 118
- "Begin Image Object (BIM)" on page 120
- "Begin Medium Map (BMM)" on page 122
- "Begin Overlay (BMO)" on page 124
- "Begin Named Page Group (BNG)" on page 126
- "Begin Object Container (BOC)" on page 129
- "Begin Object Environment Group (BOG)" on page 133
- "Begin Page (BPG)" on page 134
- "Begin Page Segment (BPS)" on page 137
- "Begin Presentation Text Object (BPT)" on page 139
- "Begin Resource (BRS)" on page 143
- "Begin Resource Group (BRG)" on page 141
- "Begin Resource Environment Group (BSG)" on page 148

Medium Orientation Triplet X'68'

The Medium Orientation triplet may be used to specify the orientation of the medium presentation space on the physical medium.

Triplet X'68' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'68'	Identifies the Medium Orientation triplet	M	X'00'
2	CODE	MedOrient	X'00'–X'05'	Orientation of the medium presentation space: **X'00'** Portrait **X'01'** Landscape **X'02'** Reverse Portrait **X'03'** Reverse Landscape **X'04'** Portrait 90 **X'05'** Landscape 90	M	X'06'

Triplet X'68' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Medium Orientation triplet.

MedOrient Specifies the position and orientation of the medium presentation space on the physical medium.

Value **Description**

X'00' **Portrait**. The origin of the medium presentation space is positioned such that the top of the presentation space (X_m axis) is parallel to a short side of the physical medium as shown in the Portrait column of Figure 70 on page 387.

X'01' **Landscape**. The origin of the medium presentation space is positioned such that the top of the presentation space (X_m axis) is parallel to a long side of the physical medium as shown in the Landscape column of Figure 70 on page 387.

X'02' **Reverse Portrait**. The origin of the medium presentation space is positioned such that the top of the presentation space (X_m axis) is parallel to a short side of the physical medium as shown in the Reverse Portrait column of Figure 70 on page 387.

X'03' **Reverse Landscape**. The origin of the medium presentation space is positioned such that the top of the presentation space (X_m axis) is parallel to a long side of the physical medium as shown in the Reverse Landscape column of Figure 70 on page 387.

X'04' **Portrait 90**. The origin of the medium presentation space is positioned such that the top of the presentation space (X_m axis) is parallel to a long side of the physical medium as shown in the Portrait 90 column of Figure 70 on page 387.

X'05' **Landscape 90**. The origin of the medium presentation space is positioned such that the top of the presentation space (X_m axis) is parallel to a short side of the physical medium as shown in the Landscape 90 column of Figure 70.

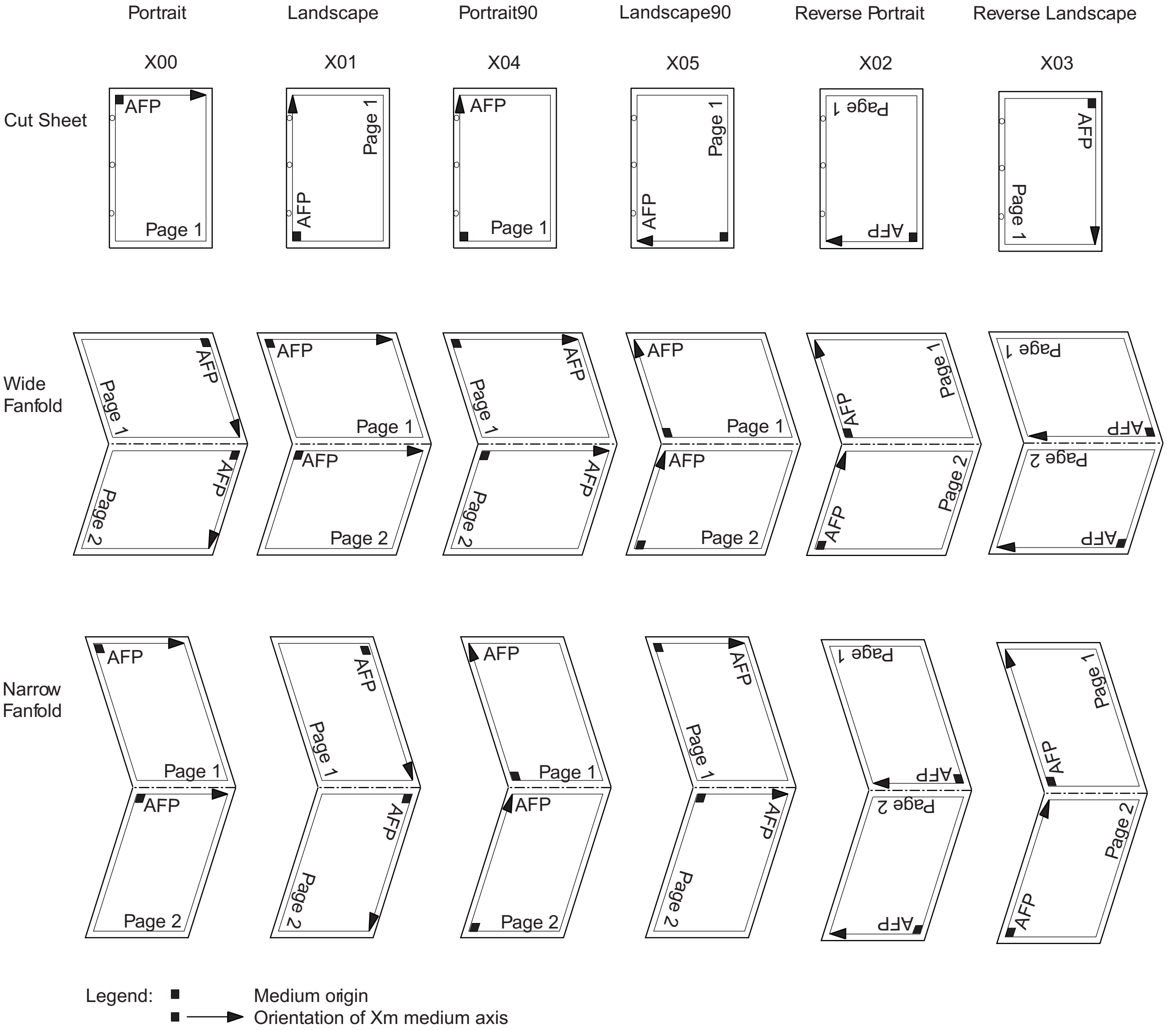


Figure 70. Landscape and Portrait Orientation and Layout

Note: In Figure 70, the text “AFP”, “Page 1”, and “Page 2” is printed in the 0° text orientation for the Portrait, Landscape, Reverse Portrait, and Reverse Landscape medium orientations, and in the 90° text orientation for the Portrait 90 and Landscape 90 medium orientations.

See Figure 58 on page 290 to Figure 69 on page 296 for a complete description of medium orientations with N-up presentation.

Structured Fields Using Triplet X'68'

- “Medium Descriptor (MDD)” on page 220

Resource Object Include Triplet X'6C'

The Resource Object Include triplet identifies an object to be included on a presentation space at a specified position.

Triplet X'6C' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	17, 19	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'6C'	Identifies the Resource Object Include triplet	M	X'00'
2	CODE	ObjType	X'DF', X'5F'	Specifies the object type: **X'DF'** Overlay object **X'5F'** Retired for private use	M	X'06'
3–10	CHAR	ObjName		Name of the object	M	X'06'
11–13	SBIN	XobjOset	0–32767	X axis origin for the object	M	X'06'
14–16	SBIN	YobjOset	0–32767	Y axis origin for the object	M	X'06'
17–18	CODE	ObOrent	X'0000', X'2D00', X'5A00', X'8700'	The overlay's X-axis rotation from the X axis of the including presentation system **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	O	X'00'

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'	Retired for private use
All others	Reserved

ObjName Specifies the object name.

XobjOset Specifies the offset along the X axis of the including presentation space coordinate system to the origin of the X axis for the object.

YobjOset Specifies the offset along the Y axis of the including presentation space coordinate system to the origin of the Y axis for the object.

ObOrent This is an optional parameter that is only supported for ObjType X'DF' = Overlay object; the parameter is ignored for other object types. Specifies the amount of rotation of the overlay's X axis, X_{ol}, about the overlay origin relative to the X axis of the including presentation space. Note that if this triplet is specified on a Page Modification Control (PMC) structured field, the including presentation space is a page, and the rotation is measured with respect to the X_p axis of the page coordinate system. Valid values are the following:

Value	Character Rotation
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.

Application Note: This triplet is used in AFP line-data environments on an LND structured field in a Page Definition object to position overlays (ObjType = X'DF') and page segments (ObjType = X'5F') with respect to line data. When used in this non-MO:DCA application, the range for the XobjOset and YobjOset parameters is −32768 to +32767. For a description of the Page Definition object and the processing of line data in AFP environments, see the *Advanced Function Presentation: Programming Guide and Line Data Reference, S544-3884*

Structured Fields Using Triplet X'6C'

- “Page Modification Control (PMC)” on page 297

Presentation Space Reset Mixing Triplet X'70'

This triplet is used to specify the resulting appearance when data in a new presentation space is merged with data in an existing presentation space.

Triplet X'70' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'70'	Identifies the Presentation Space Reset Mixing triplet	M	X'00'
2	BITS	BgMxFlag	See "Triplet X'70' Semantics" for details.	Background mixing flags	M	X'04'

Triplet X'70' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Presentation Space Reset Mixing triplet.

BgMxFlag Specifies the type of presentation space mixing as follows:

Bit **Description**

0 Reset Flag

B'0' Do not reset to the color of the medium prior to placing data into this MO:DCA presentation space. This results in the new presentation space mixing with the existing presentation space in accordance with the default MO:DCA mixing rule. Specifically, the background of the new presentation space underpaints both the background and the foreground of the existing presentation space, and the foreground of the new presentation space overpaints the background and the foreground of the existing presentation space.

B'1' Reset to the color of the medium prior to placing data into this MO:DCA presentation space. The presentation space becomes foreground data that is colored with the color of medium before any data is placed into this space. This results in the new presentation space mixing with the existing presentation space in an opaque manner. Specifically, the new presentation space, which is all foreground data, overpaints the background and foreground of the existing presentation space.

All others
Reserved

Note: If this triplet is omitted, the architected default value for the Reset Flag is B'0'—do not reset to color of medium.

Structured Fields Using Triplet X'70'

- "Include Object (IOB)" on page 180
- "Object Area Descriptor (OBD)" on page 270
- "Page Descriptor (PGD)" on page 279

Presentation Space Mixing Rules Triplet X'71'

This triplet is used to specify the rules for establishing the color attribute of areas formed by the intersection of two presentation spaces. It is specified on structured fields associated with a presentation space that is to be merged onto an existing presentation space.

Triplet X'71' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	2–10	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'71'	Identifies the Presentation Space Mixing Rules triplet	M	X'00'
2–*n*	CODE	Zero or one occurrences of each of the keywords in the following table, in ascending order				

Keyword ID	Parameter Range	Meaning	M/O	Exc
X'70'	X'01'–X'03', X'FF'	Mixing rule for background-on-background mixing	O	X'02'
X'71'	X'01'–X'03', X'FF'	Mixing rule for background-on-foreground mixing	O	X'02'
X'72'	X'01'–X'03', X'FF'	Mixing rule for foreground-on-background mixing	O	X'02'
X'73'	X'01'–X'03', X'FF'	Mixing rule for foreground-on-foreground mixing	O	X'02'

Triplet X'71' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Presentation Space Mixing Rules triplet.

Keywords One or more keywords that specify the rules for presentation space mixing. Each keyword may appear once and specifies one of the four mixing types along with the mixing rule for that mixing type. In the definitions that follow, the existing presentation space is identified by the subscript *e*, the new presentation space that is merged with the existing presentation space and that contains the Presentation Space Mixing Rules triplet is identified by the subscript *n*, the letter “B” stands for “Background”, and the letter “F” stands for “Foreground”. The Presentation Space Mixing Rules triplet appears on structures associated with the new presentation space. To completely specify the mixing of two presentation spaces, this triplet must contain four mixing rule keywords, one for each mixing type. If no keyword is specified for a particular mixing type, the MO:DCA default mixing rule is applied to this mixing type.

Keyword X'70*nn*' May occur once. Specifies the mixing rule for B_n on B_e (background on background) mixing.

Keyword X'71*nn*' May occur once. Specifies the mixing rule for B_n on F_e (background on foreground) mixing.

Keyword X'72*nn*' May occur once. Specifies the mixing rule for F_n on B_e (foreground on background) mixing.

Keyword X'73*nn*' May occur once. Specifies the mixing rule for F_n on F_e (foreground on foreground) mixing.

The following mixing rule specifications are supported in the data bytes for keywords X'70'–X'73'. For a definition of these mixing rules, see "Mixing Rules" on page 46.

Value	**Definition**
X'01'	Overpaint
X'02'	Underpaint
X'03'	Blend
X'FF'	MO:DCA default mixing rule
All others	Reserved

Note: If this triplet is not supported by a receiver, the architected default is to use the default mixing rule when mixing the new presentation space with the existing presentation space.

Implementation Note: The Presentation Space Mixing Rules (X'71') triplet is currently not used in AFP environments.

Structured Fields Using Triplet X'71'

- "Include Object (IOB)" on page 180
- "Object Area Descriptor (OBD)" on page 270
- "Page Descriptor (PGD)" on page 279

Universal Date and Time Stamp Triplet X'72'

The Universal Date and Time Stamp triplet specifies a date and time in accordance with the format defined in ISO 8601: 1988 (E).

Triplet X'72' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	13	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'72'	Identifies the Universal Date and Time Stamp triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3–4	UBIN	YearAD	0–65535	Year AD using Gregorian calendar	M	X'06'
5	UBIN	Month	1–12	Month of the year	M	X'06'
6	UBIN	Day	1–31	Day of the month	M	X'06'
7	UBIN	Hour	0–23	Hour of the day in 24-hour format	M	X'06'
8	UBIN	Minute	0–59	Minute of the hour	M	X'06'
9	UBIN	Second	0–59	Second of the minute	M	X'06'
10	CODE	TimeZone	X'00'–X'02'	Relationship of time to UTC: **X'00'** Coordinated Universal Time (UTC) **X'01'** Ahead of UTC **X'02'** Behind UTC	M	X'06'
11	UBIN	UTCDiffH	0–23	Hours ahead of or behind UTC	M	X'06'
12	UBIN	UTCDiffM	0–59	Minutes ahead of or behind UTC	M	X'06'

Triplet X'72' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Universal Date and Time Stamp triplet.

YearAD Specifies the year AD using the Gregorian calendar. For example, the year 1999 is specified as X'07CF', the year 2000 as X'07D0', and the year 2001 asX'07D1'. Represents the *YYYY* component of a date in the format *YYYYMMDD*.

Month Specifies the month of the year. January is specified as X'01', and subsequent months are numbered in ascending order. Represents the MM component of a date in the format *YYYYMMDD*.

Day Specifies the day of the month. The first day of any month is specified as X'01', and subsequent days are numbered in ascending order. Represents the DD component of a date in the format *YYYYMMDD*. For example, the date December 31, 1999 is specified as X'07CF0C1F', and January 1, 2000 is specified as X'07D00101'.

Hour Specifies the hour of the day in 24-hour format. Represents the *hh* component of a time in the format *hhmmss*.

Minute Specifies the minute of the hour. Represents the *mm* component of a time in the format *hhmmss*.

Second Specifies the second of the minute. Represents the *ss* component of a time in the format *hhmmss*. For example, the time 4:35:21 PM is specified as X'102315'.

TimeZone Defines the relation of the specified time with respect to Coordinated Universal Time (UTC). This parameter, along with the UTCDiffH and UTCDiffM parameters, is used to accommodate differences between a specified local time and UTC because of time zones and daylight savings programs. For example, Mountain Time in the US is seven hours behind UTC when daylight savings is inactive, and six hours behind UTC when daylight savings is active.

Value	Description
X'00'	Time is specified in Coordinated Universal Time (UTC). With this value, the UTCDiffH and UTCDiffM parameters should be set to X'00'. When this time is displayed or printed, the equivalence with UTC time is normally indicated with a **Z** suffix, that is, *hhmmss***Z**.
X'01'	Specified time is ahead of UTC. The number of hours ahead of UTC is specified by the UTCDiffH parameter; and the number of minutes ahead of UTC is specified by the UTCDiffM parameter. When this time is displayed or printed, the relationship with UTC time is normally indicated with a **+** character, followed by the actual time difference in hours and minutes, that is *hhmmss***+***hhmm*.
X'02'	Specified time is behind UTC. The number of hours behind UTC is specified by the UTCDiffH parameter; and the number of minutes behind UTC is specified by the UTCDiffM parameter. When this time is displayed or printed, the relationship with UTC time is normally indicated with a **–** character, followed by the actual time difference in hours and minutes, that is *hhmmss***–***hhmm*.
All others	Reserved

UTCDiffH Indicates how many hours the specified time is ahead of UTC or behind UTC. If the TimeZone parameter is X'00', this value is ignored.

UTCDiffM Indicates how many minutes the specified time is ahead of UTC or behind UTC. If the TimeZone parameter is X'00', this value is ignored.

Structured Fields Using Triplet X'72'

Either this triplet or the Local Date and Time Stamp (X'62') triplet may occur once. Only the Universal Date and Time Stamp (X'72') triplet is allowed on the BDT.

- “Begin Bar Code Object (BBC)” on page 105
- “Begin Document Index (BDI)” on page 112
- “Begin Document (BDT)” on page 114
- “Begin Form Map (BFM)” on page 116
- “Begin Graphics Object (BGR)” on page 118
- “Begin Image Object (BIM)” on page 120
- “Begin Overlay (BMO)” on page 124
- “Begin Object Container (BOC)” on page 129
- “Begin Page Segment (BPS)” on page 137
- “Begin Presentation Text Object (BPT)” on page 139
- “Begin Resource Group (BRG)” on page 141

Toner Saver Triplet X'74'

The Toner Saver triplet activates a toner saver mode for printing. The toner saver control specified by this triplet overrides any other toner saver controls that may be active in the printer.

Triplet X'74' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'74'	Identifies the Toner Saver triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3	CODE	TSvCtrl	X'00'–X'01', X'FF'	Specifies controls for the toner saver function: **X'00'** Deactivate toner saver **X'01'** Activate toner saver **X'FF'** Use device default toner saver setting	M	X'06'
4–5				Reserved; must be zero	M	X'06'

Triplet X'74' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Toner Saver triplet.

TSvCtrl Specifies how the toner saver function is to be applied to data in the presentation device. Valid values are the following:

Value	Description
X'00'	Deactivate the toner saver function.
X'01'	Activate the toner saver function. Toner saver is applied to presentation data in a device-dependent manner. In general, this may degrade print quality, and may also impact performance.
X'FF'	Use the printer default toner saver setting. Some printers allow a default for toner saving (activate or deactivate) to be set by the operator at the printer console.

If this triplet is not specified, the architected default is TSvCtrl = X'FF' (use the device default toner saver setting).

Architecture Note: Toner Saver for color printers is a function that is based on the principle that equal amounts of cyan, magenta, and yellow generate a monochromatic gray level. This leads to procedures where, given a CMY color that has some percentage of equal amounts of CMY, a percentage of CMY toner is removed (“undercolor removal”) and replaced with a percentage of K (“gray replacment”). In practice, such procedures may result in poorer color quality and may incur a performance hit.

Structured Fields Using Triplet X'74'

- “Presentation Fidelity Control (PFC)” on page 277

Color Fidelity Triplet X'75'

The Color Fidelity triplet is used to specify the exception continuation and reporting rules for invalid or unsupported color-value exceptions. This triplet also specifies a color substitution rule to be used by the presentation process when continuing after such exceptions. A color-value exception is detected when the color specification in the data stream cannot be rendered as specified by the presentation process.

Triplet X'75' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	8	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'75'	Identifies the Color Fidelity triplet	M	X'00'
2	CODE	StpCoEx	X'01'–X'02'	Color exception continuation rule: **X'01'** Stop presentation at point of first color exception and report exception **X'02'** Do not stop presentation because of color exceptions	M	X'06'
3				Reserved; must be zero	M	X'06'
4	CODE	RepCoEx	X'01'–X'02'	Color exception reporting rule if exception does not stop presentation: **X'01'** Report color exception **X'02'** Do not report color exception	M	X'06'
5				Reserved; must be zero	M	X'06'
6	CODE	ColSub	X'01'	Color substitution rule if exception does not stop presentation **X'01'** Any color substitution is permitted	M	X'06'
7				Reserved; must be zero	M	X'06'

Triplet X'75' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Color Fidelity triplet.

StpCoEx Is a parameter that specifies whether presentation should be continued when a color exception is detected. Valid values are:

Value	Description
X'01'	Stop presentation at the point of the first color exception. A color exception that stops presentation must be reported.

X'02'	Do not stop presentation because of color exceptions.
All others	Reserved

RepCoEx Is a parameter that specifies whether color exceptions should be reported if they do not stop presentation. Valid values are:

Value	**Description**
X'01'	Report color exceptions that do not stop presentation.
X'02'	Do not report color exceptions that do not stop presentation.
All others	Reserved

ColSUb Is a parameter that specifies color substitutions that the presentation process may use in order to continue presentation following a color exception. Valid values are:

Value	**Description**
X'01'	Any color or grayscale may be substituted for a color that cannot be rendered by the presentation process.
All others	Reserved

Implementation Note: The following rules describe how AFP presentation servers process the color fidelity triplet.

- If the Color Fidelity triplet is specified and is supported by the printer, the triplet is sent to the printer.
- If the Color Fidelity triplet is specified and is not supported by the printer, then
  - If StpCoEx = X'01' (stop and report), the server issues an error message and the job will not be printed.
  - If StpCoEx = X'02' (do not stop), the job will be printed.
- If the Color Fidelity triplet is not specified but is supported by the printer, the printer is instructed to reset color fidelity controls to defaults.
- If the Color Fidelity triplet is not specified and is also not supported by the printer, presentation system defaults determine how color exceptions are handled.

Structured Fields Using Triplet X'75'

- “Presentation Fidelity Control (PFC)” on page 277

Font Fidelity Triplet X'78'

The Font Fidelity triplet is used to specify the exception continuation rules for font resolution exceptions. Font resolution exceptions are generated when either:

- the font referenced in an MCF structured field is not available to the presentation system at the resolution specified in a Font Resolution and Metric Technology (X'84') triplet, or
- the resolution of the font selected by the presentation server does not match the resolution of the presentation device.

Triplet X'78' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'78'	Identifies the Font Fidelity triplet	M	X'00'
2	CODE	StpFntEx	X'01'–X'02'	Font resolution exception continuation rule: **X'01'** Stop presentation at point of first font resolution exception and report exception **X'02'** Do not stop presentation because of font resolution exceptions	M	X'06'
3–6				Reserved; must be zero	M	X'04'

Triplet X'78' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Font Fidelity triplet.

StpFntEx Is a parameter that specifies whether presentation should be continued when a font resolution exception is detected. Valid values are:

Value	Description
X'01'	Stop presentation at the point of the first font resolution exception. A font resolution exception that stops presentation must be reported.
X'02'	Do not stop presentation because of font resolution exceptions. Presentation continues either with the font at a different resolution, which may require the presentation device to apply resolution correction, or with an outline-technology version of the font.
All others	Reserved

Structured Fields Using Triplet X'78'

- “Presentation Fidelity Control (PFC)” on page 277

Attribute Qualifier Triplet X'80'

The Attribute Qualifier triplet is used to specify a qualifier for a document attribute.

Triplet X'80' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	10	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'80'	Identifies the Attribute Qualifier triplet	M	X'00'
2–5	UBIN	SeqNum	X'00000000'–X'7FFFFFFF'	Sequence Number	M	X'06'
6–9	UBIN	LevNum	X'00000000'–X'7FFFFFFF'	Level Number	M	X'06'

Triplet X'80' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Attribute Qualifier triplet.

SeqNum Is a number used to distinguish multiple instances of the same attribute.

LevNum Is a number used to maintain a hierarchical relationship between groups of attributes.

Structured Fields Using Triplet X'80'

- “Tag Logical Element (TLE)” on page 310

Page Position Information Triplet X'81'

The Page Position Information triplet is used to tag a page with the Page Position (PGP) structured field repeating group information that is used to present the page. The PGP is specified in the medium map referenced by the FQN type X'8D'—Begin Medium Map Reference triplet. This information is used for viewing the page with a particular form map, which is normally the form map that the document containing this page was archived with.

This triplet is not used for printing and is ignored by print servers.

Triplet X'81' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'81'	Identifies the Page Position Information triplet	M	X'00'
2	UBIN	PGPRG	1–8	PGP repeating group number	M	X'06'

Triplet X'81' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Page Position Information triplet.

PGPRG Identifies the PGP repeating group that is used to view the page. The PGP is specified in the medium map referenced by the FQN type X'8D' triplet. PGP repeating groups are numbered sequentially from 1 to a maximum of 8, where the first repeating group is number 1.

Structured Fields Using Triplet X'81'

- “Begin Page (BPG)” on page 134
- “Index Element (IEL)” on page 176

Parameter Value Triplet X'82'

The Parameter Value triplet is used to pass parameter values to an executable program such as an object handler or a system command interpreter.

Triplet X'82' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4–(*n*+1)	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'82'	Identifies the Parameter Value triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3	CODE	ParmSyn	X'00'–X'06'	Parameter syntax: **X'00'** Undefined **X'01'** Unsigned binary number **X'02'** Signed binary number **X'03'** Bit string **X'04'** Defined constant **X'05'** Character string **X'06'** Name	M	X'06'
4–*n*		ParmVal		Parameter value passed	O	X'00'

Triplet X'82' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Parameter Value triplet.

ParmSyn Specifies the syntax of the parameter whose value is to be passed.

Value	Description
X'00'	Syntax is undefined, data type is UNDF
X'01'	Unsigned binary number, data type is UBIN
X'02'	Signed binary number, data type is SBIN
X'03'	Bit string, where each bit can be individually and independently assigned a value, data type is BITS
X'04'	Defined or architected constant, data type is CODE
X'05'	Encoded character data, data type is CHAR
X'06'	Name, data type is CHAR
All others	Reserved

ParmVal Specifies the parameter value that is passed. If omitted, the value of the parameter is specified to be null; that is, no value is passed.

Structured Fields Using Triplet X'82'

- “Link Logical Element (LLE)” on page 199

Presentation Control Triplet X'83'

The Presentation Control triplet specifies flags that control the presentation of an object.

Triplet X'83' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'83'	Identifies the Presentation Control triplet	M	X'00'
2	BITS	PRSFlg	See "Triplet X'83' Semantics" for bit definitions	Flags that control the presentation of an object.	M	X'06'

Triplet X'83' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Presentation Control triplet

PRSFlg Specifies presentation control flags as follows:

Bit **Description**

0 Object view control.

B'0' The specified object is intended for viewing. This is the architected default if the triplet is omitted.

B'1' The specified object is not intended for viewing.

1 Object indexing control.

B'0' The specified object is intended to be indexed. This is the architected default if the triplet is omitted.

B'1' The specified object is not intended to be indexed.

2–7 Reserved

Structured Fields Using Triplet X'83'

- "Index Element (IEL)" on page 176
- "Begin Named Page Group (BNG)" on page 126
- "Begin Page (BPG)" on page 134

Font Resolution and Metric Technology Triplet X'84'

The Font Resolution and Metric Technology specifies certain metric characteristics of a raster-technology font character set which may have affected the formatting of the document with this font. This information, as carried by the X'84' triplet, may be used by presentation servers and presentation devices to select the best-matching coded font for presentation.

Triplet X'84' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'84'	Identifies the Font Resolution and Metric Technology triplet	M	X'00'
2	CODE	MetTech	X'01'–X'02'	Metric Technology: **X'01'** Fixed-metric technology **X'02'** Relative-metric technology	M	X'06'
3	CODE	RPuBase	X'00'	Raster-pattern resolution unit base: **X'00'** 10 inches	M	X'06'
4–5	UBIN	RPUnits	X'0960', X'0BB8'	Raster-pattern resolution units per unit base	M	X'06'

Triplet X'84' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Font Resolution and Metric Technology triplet.

MetTech Specifies the metric technology used by this raster font. For a description of fixed-metric and relative-metric technologies, see the *Intelligent Printer Data Stream Reference* and the *Font Object Content Architecture Reference*.

RPuBase Specifies the unit base for the raster font’s resolution.

RPUnits Specifies the number of pels per unit base of the font’s raster-pattern shape data.

Structured Fields Using Triplet X'84'

- “Map Coded Font (MCF) Format 2” on page 213

Finishing Operation Triplet X'85'

The Finishing Operation triplet is used to specify finishing operations that are to be applied to media.

Architecture Note: The format for specifying finishing operations and their associated parameters is based on the Document Printing Application (DPA) ISO/IEC DLS 10175:1991 draft standard. The definition of an operation or parameter in this triplet does not guarantee its support in a MO:DCA-P presentation system. To see which operations and parameters are supported by AFP printers, consult the appropriate product documentation.

Triplet X'85' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	9–253	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'85'	Identifies the Finishing Operation triplet	M	X'00'
2	CODE	FOpType	X'01'–X'03', X'04', X'05'–X'07', X'08',X'0A', X'12'	Finishing operation type: **X'01'** Corner staple **X'02'** Saddle stitch out **X'03'** Edge stitch **X'04'** Fold in **X'05'** Separation cut **X'06'** Perforation cut **X'07'** Z-fold **X'08'** Center fold in **X'0A'** Punch **X'12'** Saddle stitch in	M	X'06'
3–4				Reserved; must be zero	M	X'06'
5	CODE	RefEdge	X'00'–X'03', X'FF'	Finishing operation reference corner or edge: **X'00'** Bottom-right corner, bottom edge **X'01'** Top-right corner, right edge **X'02'** Top-left corner, top edge **X'03'** Bottom-left corner, left edge **X'FF'** Device default reference corner or edge	M	X'06'
6	UBIN	FOpCnt	X'00'–X'7A'	Finishing operation count: **X'00'** Not specified; use OpPos parameters or device default **X'01'–X'7A'** Number of operations to apply; must match number of OpPos parameters if they are specified	M	X'06'

Offset	Type	Name	Range	Meaning	M/O	Exc
7–8	UBIN	AxOffst	0–32767	Finishing operation axis offset in millimeters	M	X'06'
			X'FFFF'	Device default axis offset		
Zero or more occurrences of the following parameters:						
0–1	UBIN	OpPos	0–32767	Operation position on finishing operation axis in millimeters	O	X'02'

Triplet X'85' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Finishing Operation triplet.

FOpType Is a parameter that specifies the type of finishing operation. The operation is applied either on a reference corner or along a finishing operation axis that is offset from a reference edge.

Value	Operation Type
X'01'	Corner staple. A staple is driven into the media at the reference corner. The offset of the staple from the corner and the staple angle are device-dependent. The AxOffst, FOpCnt, and OpPos parameters are ignored for this operation. This operation is applied to collected media, not to individual media.
X'02'	Saddle stitch out. One or more staples are driven into the media along the finishing operation axis, which is positioned at the center of the media parallel to the reference edge. The AxOffst parameter is ignored for this operation. This operation also includes a fold of the media outward along the finishing operation axis so that the front-side of the first sheet in the collection is on the outside of the media collection. This operation is applied to collected media, not to individual media. Note that the pages in the datastream must already be properly ordered for this operation.
X'03'	Edge stitch. One or more staples are driven into the media along the finishing operation axis. This operation is applied to collected media, not to individual media.
X'04'	Fold in.The media is folded inward on the front sheet-side. If applied to a collection of media, the collection is folded inward on the front sheet-side of the first sheet, and at the end of this operation the back side of the last sheet of the collection is on the outside. The folding is performed along the finishing operation axis. The FOpCnt and OpPos parameters are ignored for this operation. Note

that if applied to a collection of media, the pages in the datastream must already be properly ordered for this operation.

X'05' Separation cut. A separation cut is applied to the media along the finishing operation axis. The FOpCnt and OpPos parameters are ignored for this operation.

X'06' Perforation cut. A perforation cut is applied to the media along the finishing operation axis. The FOpCnt and OpPos parameters are ignored for this operation.

X'07' Z-fold. A Z-fold is applied to each medium, or sheet. The medium is first folded in half inwards along a line parallel to the reference edge. The half of the medium furthest from the reference edge is then again folded in half outwards along a line parallel to the reference edge. When applied to an 11×17-inch sheet with the reference edge specified as the top edge, the result is an 8.5×11-inch fold-out.

Note: If additional finishing operations are applied to the Z-folded sheet, the original reference edge becomes the left edge of the Z-folded sheet. In the example above, the reference edge for the Z-fold was the top (11-inch) edge. After Z-folding is applied, the sheet is re-oriented so that this reference edge now becomes the *left* edge for additional finishing operations. Therefore if the Z-folded sheets are to be stapled to the left edge of 8.5×11–inch sheets, the stapling reference edge for both sets of sheets is the left edge.

Architecture Note: There is an exception to the rule for reorientation after Z-fold. If the media is sized such that the reference edge is less than half the size of the other sheet dimension, the reorientation causes the reference edge to become the new top edge for additional finishing operations instead of the new left edge.

The FOpCnt, AxOffst, and OpPos parameters are ignored for this operation. Note that the Z-fold is applied to each individual medium, not to the collected media.

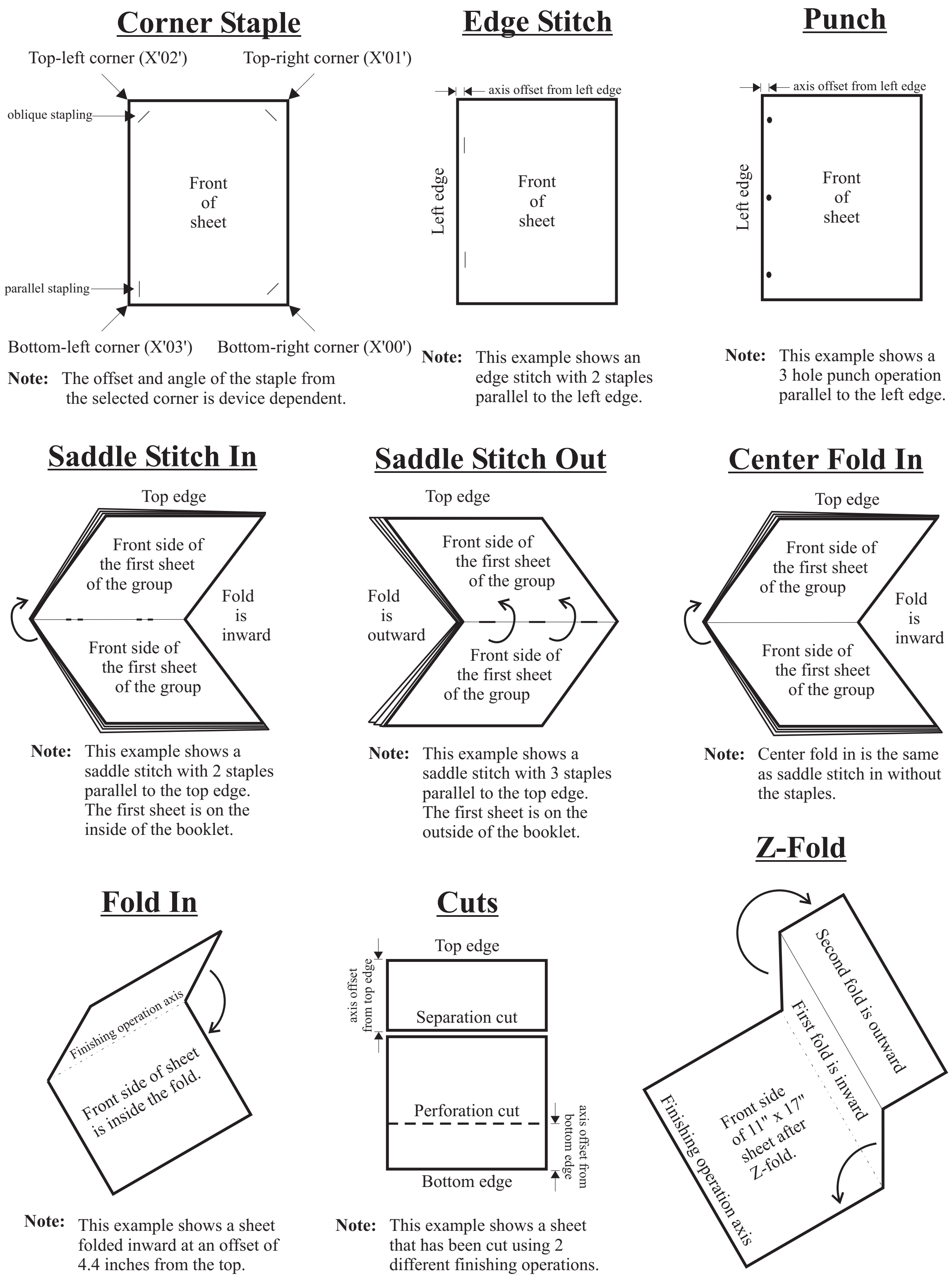
X'08' Center fold in. The media is folded inward on the front sheet-side. If applied to a collection of media, the collection is folded inward on the front sheet-side of the first sheet, and at the end of this operation the back side of the last sheet of the collection is on the outside. The folding is performed along the center line that is parallel to the finishing operation axis. The FOpCnt and OpPos parameters are ignored for this operation. Note that if applied to a collection of media, the pages in the datastream must already be properly ordered for this operation.

X'0A' Punch. One or more holes are punched or drilled into the media along the finishing operation axis. This operation is applied to collected media, not to individual media.

X'12' Saddle stitch in. One or more staples are driven into the media along the finishing operation axis, which is positioned at the center of the media parallel to the reference edge. The AxOffst parameter is ignored for this operation. This operation also includes a fold of the media inward along the finishing operation axis so that the front-side page of the first sheet in the collection is on the inside of the media collection. This operation is applied to collected media, not to individual media. Note that the pages in the datastream must already be properly ordered for this operation.

All others Reserved

Figure 71 on page 411 shows examples of finishing operations.


Top-left corner (X'02')
Top-right corner (X'01')
oblique stapling
parallel stapling
Front of sheet
Bottom-left corner (X'03')
Bottom-right corner (X'00')
axis offset from left edge
Left edge
Front of sheet
axis offset from left edge
Left edge
Front of sheet
Top edge
Front side of the first sheet of the group
Fold is inward
Front side of the first sheet of the group
Top edge
Front side of the first sheet of the group
Fold is outward
Front side of the first sheet of the group
Top edge
Front side of the first sheet of the group
Fold is inward
Front side of the first sheet of the group
Finishing operation axis
Front side of sheet is inside the fold.
Top edge
axis offset from top edge
Separation cut
Perforation cut
axis offset from bottom edge
Bottom edge
Second fold is outward
First fold is inward
Front side of 11" x 17" sheet after Z-fold.
Finishing operation axis

Corner Staple

Note: The offset and angle of the staple from the selected corner is device dependent.

Edge Stitch

Note: This example shows an edge stitch with 2 staples parallel to the left edge.

Punch

Note: This example shows a 3 hole punch operation parallel to the left edge.

Saddle Stitch In

Note: This example shows a saddle stitch with 2 staples parallel to the top edge. The first sheet is on the inside of the booklet.

Saddle Stitch Out

Note: This example shows a saddle stitch with 3 staples parallel to the top edge. The first sheet is on the outside of the booklet.

Center Fold In

Note: Center fold in is the same as saddle stitch in without the staples.

Z-Fold

Fold In

Note: This example shows a sheet folded inward at an offset of 4.4 inches from the top.

Cuts

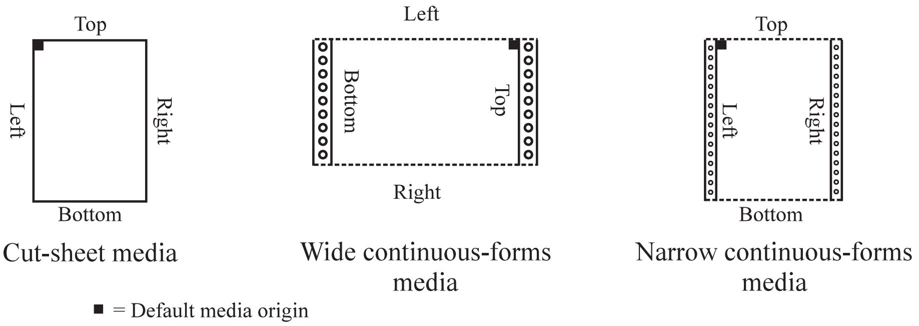
Note: This example shows a sheet that has been cut using 2 different finishing operations.

Figure 71. Examples of Finishing Operations

RefEdge Is a parameter that selects the medium reference corner and the medium reference edge for finishing operations. Edge and corner definitions for cut-sheet and continuous-forms media are shown in Figure 72 on page 412. Valid values are:

Value	Description
X'00'	Bottom-right corner, bottom edge
X'01'	Top-right corner, right edge
X'02'	Top-left corner, top edge
X'03'	Bottom-left corner, left edge
X'FF'	Presentation device default reference corner or edge
All others	Reserved

Note: For all types of media shown in Figure 72, the top-left corner is defined to be the default media origin of the front side. A change in the orientation of the medium presentation space does not change the finishing corners or edges. For continuous-forms media, the carrier strips are not considered to be part of the physical media.



Cut-sheet media

Wide continuous-forms media

Narrow continuous-forms media

Figure 72. Media Reference Edge and Corner Definitions

FOpCnt Is a parameter that specifies the number of discrete finishing operations that are to be applied by this operation type along the finishing operation axis. For example, if the operation type is edge-stitch, the FOpCnt parameter specifies how many staples are to be applied along the finishing operation axis. Valid values are:

Value	Description
X'00'	Count not specified. Use the count implied by the number of OpPos parameters if they are specified or use the presentation device default count if OpPos parameters are not specified.
X'01'–X'7A'	Apply the specified number of finishing operations. This count must match the number of OpPos parameters if OpPos parameters are specified; if OpPos parameters are not specified, presentation device default positions are used.
All others	Reserved

AxOffst Is a parameter that specifies the offset of the finishing operation axis from the reference edge. The offset is measured in millimeters from the reference edge toward the center of the medium. A value of X'FFFF' indicates that the presentation device default finishing operation axis offset is to be used.

OpPos Is a parameter that specifies the offset of the finishing operation along the finishing operation axis. The offset is measured in millimeters from the point where the finishing operation axis intersects either the bottom edge or the left edge of the medium, toward the center of the medium. Each consecutive OpPos parameter is used to position a single finishing operation centered on the specified point on the finishing operation axis. This continues until the last OpPos parameter has been processed.

Structured Fields Using Triplet X'85'

- “Medium Finishing Control (MFC)” on page 236

Text Fidelity Triplet X'86'

The Text Fidelity triplet is used to specify the exception continuation and reporting rules for text exceptions. A text exception is detected when an unrecognized or unsupported text control sequence is encountered in a PTOCA text object.

Triplet X'86' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'86'	Identifies the Text Fidelity triplet	M	X'00'
2	CODE	StpTxtEx	X'01'–X'02'	Text exception continuation rule: **X'01'** Stop presentation at point of first text exception and report exception **X'02'** Do not stop presentation because of text exceptions	M	X'06'
3				Reserved; must be zero	M	X'06'
4	CODE	RepTxtEx	X'01'–X'02'	Text exception reporting rule if exception does not stop presentation: **X'01'** Report text exception **X'02'** Do not report text exception	M	X'06'
5–6				Reserved; must be zero	M	X'06'

Triplet X'86' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Text Fidelity triplet.

StpTxtEx Is a parameter that specifies whether presentation should be continued when a text exception is detected. Valid values are:

Value	Description
X'01'	Stop presentation at the point of the first text exception. A text exception that stops presentation must be reported. **Application Note:** When presentation is terminated, the print file is put into a state where it can be re-submitted when the text can be rendered without exceptions.
X'02'	Do not stop presentation because of text exceptions.

All others	Reserved

RepTxtEx Is a parameter that specifies whether text exceptions should be reported if they do not stop presentation. Valid values are:

Value	Description
X'01'	Report text exceptions that do not stop presentation.
X'02'	Do not report text exceptions that do not stop presentation.
All others	Reserved

Implementation Note: The following rules describe how AFP presentation servers process the Text Fidelity triplet.

- If the Text Fidelity triplet is specified and is supported by the printer, the triplet is sent to the printer and processed by both server and printer. If StpTxtEx = X'02' and a text exception is detected, the text control sequence that generated the exception is skipped or processed in non-optimal fashion and printing continues with the next text control sequence.
- If the Text Fidelity triplet is specified and is not supported by the printer, the triplet is processed by the server. Text exceptions will flow from the printer to the server. If StpTxtEx = X'02' and a text exception is detected, printing continues after the remainder of the text object - which could encompass the whole page - is skipped.
- If the Text Fidelity triplet is not specified, presentation system defaults determine how text exceptions are handled.

Structured Fields Using Triplet X'86'

- “Presentation Fidelity Control (PFC)” on page 277

Media Fidelity Triplet X'87'

The Media Fidelity triplet is used to specify the continuation rule if a request for a specific media or a specific media bin cannot be satisfied.

Triplet X'87' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'87'	Identifies the Media Fidelity triplet	M	X'00'
2	CODE	StpMedEx	X'01'–X'02'	Media exception continuation rule: **X'01'** Terminate job and hold **X'02'** Continue with defaults	M	X'06'
3–6				Reserved; must be zero	M	X'06'

Triplet X'87' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Media Fidelity triplet.

StpMedEx Is a parameter that specifies the continuation rule for the presentation system if the requested media or the requested media bin is not available in the presentation device. Valid values are:

Value **Description**

X'01' Terminate presentation

Application Note: When presentation is terminated, the print file is put into a state where it can be resubmitted when the proper media is loaded or when the proper media source is made available.

X'02' Continue with the presentation system defaults

All others Reserved

Implementation Note: AFP print servers will attempt to select the media using the following priority:

1. Attempt to find an available media source containing the media type that matches the specified media OID. The media source can not be an inserter bin.
2. Attempt to find an available media source containing the media type that matches the specified media name. The media source can not be an inserter bin.
3. Attempt to find an available media source whose ID matches the specified ID.

4. If the continuation rule is X'02' (continue with defaults), use the presentation process defaults for finding an available media source. If the continuation rule is X'01', presentation is terminated.

Structured Fields Using Triplet X'87'

- "Presentation Fidelity Control (PFC)" on page 277

Finishing Fidelity Triplet X'88'

The Finishing Fidelity triplet is used to specify the exception continuation and reporting rules for finishing exceptions. A finishing exception is detected when the specified finishing operation cannot be satisfied.

Triplet X'88' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	7	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'88'	Identifies the Finishing Fidelity triplet	M	X'00'
2	CODE	StpFinEx	X'01'–X'02'	Finishing exception continuation rule: **X'01'** Stop presentation at point of first finishing exception and report exception **X'02'** Do not stop presentation due to finishing exceptions	M	X'06'
3				Reserved; must be zero	M	X'06'
4	CODE	RepFinEx	X'01'–X'02'	Finishing exception reporting rule if exception does not stop presentation **X'01'** Report finishing exception **X'02'** Do not report finishing exception	M	X'06'
5–6				Reserved; must be zero	M	X'06'

Triplet X'88' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Finishing Fidelity triplet.

StpFinEx Is a parameter that specifies whether presentation should be continued when a finishing exception is detected. Valid values are:

Value	Description
X'01'	Stop presentation at point of first finishing exception. A finishing exception that stops presentation must be reported. **Application Note:** When presentation is terminated, the print file is put into a state where it can be re-submitted when the finishing operation can be performed.
X'02'	Do not stop presentation due to finishing

Value	Description
	exceptions. Presentation continues without applying the finishing operation that cannot be satisfied.
All others	Reserved

RepFinEx Is a parameter that specifies whether finishing exceptions should be reported if they do not stop presentation. Valid values are:

Value	**Description**
X'01'	Report finishing exceptions that do not stop presentation.
X'02'	Do not report finishing exceptions that do not stop presentation.
All others	Reserved

Note: This triplet covers finishing operations that the printer is incapable of processing such as a stapling operation on a device that does not have a stapler attached. It does not cover temporary exceptions such as out-of-finishing-supplies conditions, which result in a printer intervention condition that is cleared as soon as supplies are added.

Implementation Note: The following rules describe how AFP presentation servers process the Finishing Fidelity triplet.

- If the Finishing Fidelity triplet is specified and is supported by the printer, the triplet is sent to the printer and processed by both server and printer.
- If the Finishing Fidelity triplet is specified and is not supported by the printer, the triplet is processed by the server. Finishing exceptions will flow from the printer to the server; this may cause a performance degradation. If StpFinEx = X'02' and RepFinEx = X'02', the server will suppress the finishing error messages.
- If the Finishing Fidelity triplet is not specified, the job is printed and the finishing operations that cannot be satisfied are not applied. Finishing exceptions are reported.

Structured Fields Using Triplet X'88'

- “Presentation Fidelity Control (PFC)” on page 277

Data-Object Font Descriptor Triplet X'8B'

The Data-Object Font Descriptor triplet is used to specify the parameters needed to render a data-object font. Data-object fonts are non-FOCA font resources, such as TrueType and OpenType fonts. An MDR structured field is used to map a data-object font as a resource.

Triplet X'8B' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	16	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'8B'	Identifies the Data-Object Font Descriptor triplet	M	X'00'
2				Reserved; must be zero	M	X'06'
3	CODE	FontTech	X'20'	Font technology: **X'20'** TrueType/ OpenType	M	X'06'
4–5	UBIN	VFS	1–32767	Specified vertical font size	M	X'06'
6–7	UBIN	HFS	1–32767	Horizontal scale factor	M	X'06'
			X'0000'	Not specified		
8–9	CODE	CharRot	X'0000', X'2D00', X'5A00', X'8700'	Clockwise character rotation in degrees **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	M	X'06'
10–11	CODE	EncEnv	X'0003'	Encoding environment **X'0003'** Microsoft	M	X'06'
12–13	CODE	EncID	X'0001'	Environment-specific encoding identifier **X'0001'** Unicode	M	X'06'
14–15				Reserved; must be zero	M	X'06'

Triplet X'8B' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the Data-Object Font Descriptor triplet.

FontTech Identifies the font technology of the font. Valid values are:

Value	Description
X'20'	TrueType/OpenType
All others	Reserved

VFS Specifies the vertical font size in 1440ths of an inch. The specified vertical font size is the desired distance between adjacent character baselines when character rotation is zero degrees and no external leading is used. The desired vertical size of the font is often called ″point size″ because formatting programs typically specify this size in point units (1/72 inch); in this case, the vertical font size can be calculated by multiplying the desired point size by 20.

HSF Specifies the horizontal scale factor in 1440ths of an inch. The horizontal scale factor specifies the numerator of a scale factor for the horizontal direction in 1440ths of an inch. The character shapes and metrics are stretched or compressed in the horizontal direction by the ratio of HSF/VFS. When the vertical font size and the horizontal scale factor are identical or when HSF=X'0000' is specified, a uniform scaling occurs; when these two parameters are different, an anamorphic scaling occurs.

CharRot Specifies the clockwise character rotation in degrees. This parameter specifies a clockwise rotation of a character pattern (glyph) from the character baseline. For a description of character rotation, see the *Font Object Content Architecture (FOCA) Reference*, S544-3285. The four allowed character rotations provide for different writing modes (left-to-right, top-to-bottom, right-to-left, and bottom-to-top). A normal (right-side-up) character has a character rotation of 0 degrees; an up-side-down character has a character rotation of 180 degrees. A character rotation of 270 degrees is normally used for vertical writing. The valid character rotation values are:

X'0000'	0 degrees (left-to-right writing)
X'2D00'	90 degrees (bottom-to-top writing)
X'5A00'	180 degrees (right-to-left writing)
X'8700'	270 degrees (top-to-bottom writing)

Figure 73 on page 422 shows the placement of characters based on the character rotation value and the PTOCA inline and baseline direction values.

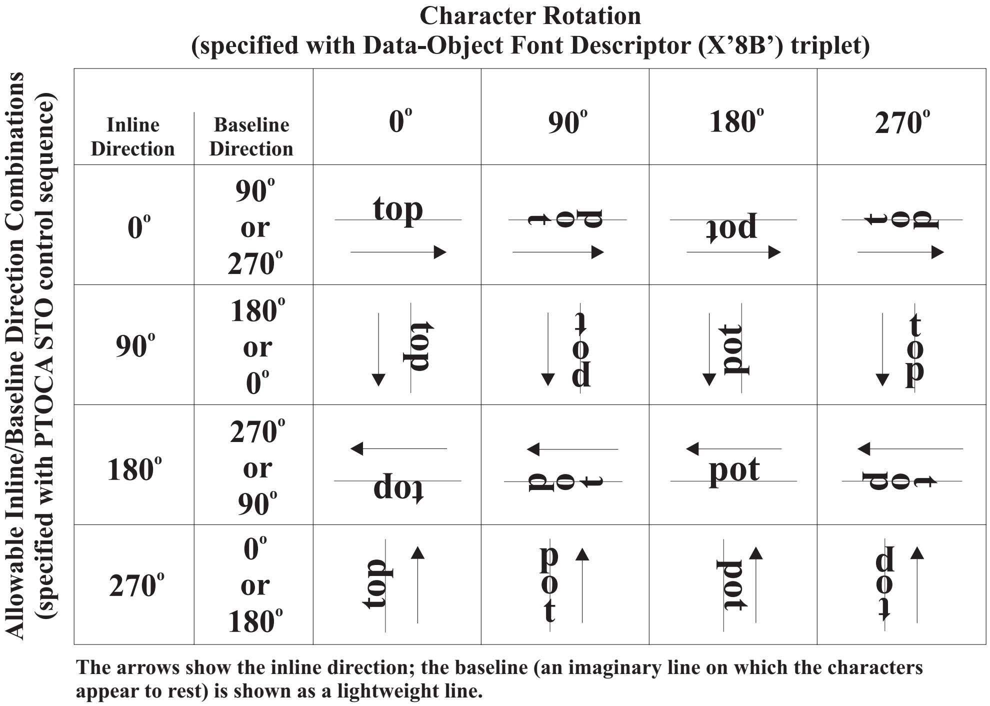

Character Rotation
(specified with Data-Object Font Descriptor (X'8B') triplet)
Allowable Inline/Baseline Direction Combinations
(specified with PTOCA STO control sequence)
Inline Direction
Baseline Direction
0°
90°
180°
270°
0°
90° or 270°
90°
180° or 0°
180°
270° or 90°
270°
0° or 180°
top
The arrows show the inline direction; the baseline (an imaginary line on which the characters appear to rest) is shown as a lightweight line.

Figure 73. Character Placement Based on Character Rotation and Inline and Baseline Direction

TrueType fonts provide two sets of metrics to allow character placement for different writing modes. The metrics for horizontal writing are used when the character rotation is 0 degrees, and a modified version of the horizontal metrics is used for a 180 degree character rotation. Likewise, the metrics for vertical writing are used when the character rotation is 270 degrees, and a modified version of the vertical metrics is used for a 90 degree character rotation.

Architecture Notes:

1. The character rotation parameter is used in PTOCA text objects along with the current inline and baseline directions to determine the character orientation with respect to the page (X_p, Y_p) coordinate system.
2. The character-rotation parameter applies only to characters used in PTOCA text objects or BCOCA bar code objects. For GOCA graphics objects, the Set Character Angle drawing order provides analogous function.

EncEnv Specifies the environment for the encoding in the font.

Architecture Note: In TrueType/OpenType font files, this parameter is called the *Platform ID*.

Value	**Description**
X'0003'	Microsoft
All others	Reserved.

This parameter, along with the EncID parameter, identifies a character encoding within the font that is used to map code points to glyphs and metrics. Note that different font technologies use different methods to achieve this purpose:

- The TrueType/OpenType font technology uses an internal cmap table for this purpose; most TrueType fonts contain a Unicode cmap subtable and some TrueType fonts also contain additional cmap subtables to allow the font to be used with a variety of character encoding schemes. The cmap subtable is indexed with the EncEnv and EncID parameters.

  Note: A TrueType/OpenType font can also be used with user data that is encoded to be rendered with a traditional IBM FOCA font. Such FOCA fonts use an IBM code page to map code points to graphic character identifiers. To support the presentation of such data with TrueType/OpenType fonts, the user data encoding and the corresponding code page are specified on the MDR that is used to reference the TrueType/OpenType font. A mapping function in the presentation system is invoked to map the IBM graphic character identifiers to Unicode code points in the TrueType/OpenType font's cmap subtable defined by EncEnv = Microsoft (X'0003') and EncID = Unicode (X'0001').

The valid encoding-environment values for each font technology are:

Font Technology	Encoding Environment
TrueType/OpenType	Microsoft (X'0003')

EncID Specifies the character encoding that is to be used to interpret the meaning of each code point.

Architecture Note: In TrueType/OpenType font files, this parameter is called the *Encoding ID*.

The values that are valid for the encoding identifier depend on the specified encoding environment parameter. For the Microsoft encoding environment (EncEnv = X'0003'), the following encoding identifiers are supported:

Value	Description
X'0001'	Unicode

Structured Fields Using Triplet X'8B'

- “Map Data Resource (MDR)” on page 223

UP3i Finishing Operation Triplet X'8E'

The UP3i Finishing Operation triplet is used to specify finishing operations that are to be applied to media. More specifically, this triplet is a carrier for finishing operations and parameters that are defined by the UP3i consortium in the UP3i Specification.

Triplet X'8E' Syntax

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	13–254	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'8E'	Identifies the UP3i Finishing Operation triplet	M	X'00'
2–3				Reserved; must be zero	M	X'06'
4–*n*		UP3iDat		Finishing operation data as defined in the UP3i Specification; this parameter contains bytes 4–end of the UP3i Form Finishing Operating (X'03') triplet	M	X'06'

Triplet X'8E' Semantics

Tlength Contains the length of the triplet.

Tid Identifies the UP3i Finishing Operation triplet.

UP3iDat Specifies finishing operations and parameters defined by the UP3i consortium. This parameter contains bytes 4–end of the UP3i Form Finishing Operating (X'03') triplet. At least bytes 4–12 of the UP3i Form Finishing Operating (X'03') triplet are mandatory and must be specified for the UP3iDat parameter; additional bytes are optional. The semantics of the bytes are defined by the UP3i Specification. For a definition of the UP3i Form Finishing Operating (X'03') triplet, see the current UP3i Specification. This specification is available on the UP3i home page at

http://www.up3i.org

.

Structured Fields Using Triplet X'8E'

- “Medium Finishing Control (MFC)” on page 236

Chapter 7. MO:DCA Interchange Sets

This chapter:

- Describes compliance in terms of interchange sets
- Defines classes of interchange set compliance
- Outlines MO:DCA compliance rules
- Identifies which interchange sets are part of SAA
- Provides complete syntactic and semantic descriptions of
  - MO:DCA Presentation Interchange Set 1
  - MO:DCA Presentation Interchange Set 2
  - MO:DCA Resource Interchange Set

Interchange Sets

MO:DCA interchange compliance is based on, and controlled by, interchange sets. An interchange set is used to describe the functional content of a MO:DCA document type and to identify the level of support that products must provide to generate and receive that document type. A product is in compliance with the MO:DCA interchange architecture when it supports, without deviation, at least one interchange set.

There are two types of MO:DCA data streams: MO:DCA data streams for presentation documents (MO:DCA-P), and MO:DCA-L data streams for resource documents (MO:DCA-L). See “Documents” on page 12 for a discussion of these document types. Interchange sets for these two document types are completely independent and unrelated. However, within a specific document type, all interchange sets are defined with a subset/superset architecture. This means that a higher interchange level includes all the functions of a lower interchange set level.

This edition of the *Mixed Object Document Content Architecture Reference* contains two interchange sets for MO:DCA-P (MO:DCA-P IS/1 and MO:DCA-P IS/2), and one for MO:DCA-L.

While an interchange set cannot be defined that violates the overall MO:DCA architecture, the interchange set definition can include restrictions that are not part of the overall architecture. These restrictions may limit:

- What structured fields may or must appear
- Where the structured fields may or must appear
- The order in which the structured fields may or must appear
- What structured field parameters may or must appear
- The order in which the structured field parameters may or must appear
- What structured field parameter values may or must appear

Interchange Set Compliance Requirements

The MO:DCA architecture defines four distinct classes of interchange set compliance. These classes are as follows:

Generator
Any product that produces a valid subset of the interchange set. A valid subset of an interchange set is one in which all generated structured fields belong to the interchange set and comply with all of its ordering and pairing requirements, and all parameter values fall within the ranges specified by the interchange set. While a generator also may receive its own version of the interchange set, it is not considered a MO:DCA-compliant receiver unless it receives the entire interchange for which compliance is claimed.

Receiver
Any product that properly interprets all MO:DCA structured fields in the interchange set for which compliance is claimed. A compliant receiver need not process all of the OCAs that are associated with the interchange set. An example of this would be an image only editor receiving an interchange set consisting of text, image, and graphics. The editor would be able to process the image content regardless of its location in the data stream, but could not handle the OCA-dependent portions of the text and graphics structured fields.

Filtering receiver-generator
Any product that contains:

- A compliant generator, and
- A compliant receiver that, after receiving a MO:DCA data stream, discards all portions of it that pertain to OCAs that are not supported.

This product can regenerate only those portions of a received MO:DCA data stream that pertain to OCAs that are supported. An example of this would be a FAX product that receives a MO:DCA document, extracts and processes only the image portion, and regenerates a MO:DCA document that contains only the image portion.

Preserving receiver-generator
Any product that contains:

- A compliant generator, and
- A compliant receiver that retains all portions of a MO:DCA data stream without regard to its OCA content.

This product can regenerate a received MO:DCA document with absolute fidelity if it has made no changes to the OCA-dependent portions that it can process. If it has made changes to the OCA-dependent portions, the unrecognized portions of the received document should appear in the same context in the retransmitted document unless the changes have resulted in their deletion. An example of this would be a text editor that receives a MO:DCA document, modifies the text portion, and regenerates it as a MO:DCA document. If any of the modifications involve the deletion of pages containing graphics or image objects, the graphics or image objects contained in those pages also are deleted. The regenerated document contains the modified text portion along with any surviving graphics and image portions that appeared in the received document.

In order to claim MO:DCA interchange compliance with a specific interchange set, a product must satisfy the following requirements:

- The product must support, within the scope of its assigned class, at least one interchange set.
- A generator product must generate MO:DCA documents that are completely valid syntactically.
- A receiver product must be able to receive MO:DCA documents that are completely valid syntactically.
- Receiver products must detect the exception conditions defined by the MO:DCA architecture that apply to the highest interchange set supported, within the scope of the supported OCAs. Exception conditions detected should be reported to an exception handler within the receiver.

  Note: In general, the actions to be taken by the exception handler are product dependent and not defined by the MO:DCA architecture. However, receiver products must be capable of skipping over unrecognized structured fields and parameters when instructed to do so by the exception handler.

In order to claim compliance, products must support at least one OCA from among those belonging to an interchange set. MO:DCA-compliant products are obligated only to process the information and function in a received document that belongs to their supported interchange set, within the scope of the supported OCAs. All products should identify, within their product documentation, which class of compliance they claim and which interchange set they support. All products should identify, within their product documentation, which OCAs they support.

Specific interchange sets may have additional compliance rules. See the specific interchange set definition for more information.

Note: The primary intent of the MO:DCA architecture is the interchange of data among products that support one or more defined interchange sets. However, products may also use MO:DCA data streams for their own private use or for data exchange with other known products. Usually, this type of data stream is patterned after one of the defined interchange sets but is not fully compliant with it. For example, a product may have a need to support presentation page sizes that are larger than those supported by the existing presentation interchange sets. Products that generate this type of data stream must ensure that the MO:DCA Interchange Set triplet is not included on their data stream's Begin Document structured field. If this type of data stream is then inadvertently sent to a compliant receiver, that receiver may safely reject or ignore the entire data stream after determining from the Begin Document structured field that the data stream does not represent a defined interchange set.

SAA Interchange Sets

The interchange sets that are part of Systems Application Architecture (SAA) are:

- MO:DCA Presentation Interchange Set 1 (MO:DCA-P IS/1)
- MO:DCA Presentation Interchange Set 2 (MO:DCA-P IS/2)
- MO:DCA Resource Interchange Set (MO:DCA-L)

MO:DCA Presentation Interchange Set 1

This section defines the MO:DCA Presentation Interchange Set 1 (MO:DCA-P IS/1) used for presentation documents.

For information on the level of function required for the OCAs included in this interchange set, refer to the MO:DCA environment appendix in the following IBM documents:

GOCA
Graphics Object Content Architecture for Advanced Function Presentation Reference, S544-5498

IOCA *Image Object Content Architecture Reference*, SC31-6805

PTOCA
Presentation Text Object Content Architecture Reference, SC31-6803

Data Stream Syntax Structure

The groupings of MO:DCA structured fields that follow identify those structured fields which appear within each begin-end structured field pair or state. This section specifies the structured fields allowed within a MO:DCA Presentation Interchange Set 1 data stream. It shows the MO:DCA state hierarchy and the validity of structured fields within each state.

If a structured field that is not identified as being part of this interchange set appears anywhere within the data stream, a X'40' exception condition exists. If a structured field appears within any state where it is not permitted, or if it appears out of the stated order or more than the permitted number of times, a X'20' exception condition exists. If a structured field that is identified as required does not appear within a specific state, a X'08' exception condition exists.

The conventions used in these structured field groupings are:

( ) The structured field acronym and identifier are shown in parentheses. The presence of dots or periods in the identifier indicates that the item is not a structured field, but instead is a structure, for example a page. The structure is composed of an assortment of structured fields, and is defined separately.

[ ] Brackets indicate optional structured fields. When a structured field is shown without brackets, it *must* appear between the begin and end structured fields.

+ Plus signs indicate structured fields may appear in any order relative to those that precede or succeed it except when the preceding or succeeding structured field does not have a plus (+) sign. Then the order is as listed.

(S) The enclosed (S) indicates that the structured field may be repeated. When it is present on a required structured field, at least one occurrence of the structured field is required, but multiple instances of it may occur.

F2 An F2 indicates that the structured field is a format two structured field. See “Structured Field Formats” on page 25 for further details.

Notes:

1. The Begin Document and End Document structured fields are required in a MO:DCA data stream.

2. The No Operation structured field may appear within any begin-end domain and thus is not listed in the structured field groupings.
3. The architecture that owns and controls the content of each of the data and resource objects carried in a MO:DCA data stream is identified in the following structured field groupings. Please refer to the referenced documentation for further details.
4. The Flag byte (byte 5) in the Structured Field Introducer (SFI) must be set to X'00'. MO:DCA-P IS/1 does not support SFI extension, structured field segmentation, or structured field padding.

Document

```
Begin Document  (BDT, D3A8A8)
 +   [  (IMM,  D3ABCC)    Invoke Medium Map                           (S)  ]
 +   [  (      D3..AF)    Page                                        (S)  ]
End Document  (EDT, D3A9A8)
```

Figure 74. MO:DCA-P IS/1: Document Structure

Page

```
Begin Page  (BPG, D3A8AF)
        (      D3..C9)    Active Environment Group
 +   [  (      D3..BB)    Graphics Object                             (S)  ]
 +   [  (      D3..FB)    Image Object                                (S)  ]
 +   [  (IPO,  D3AFD8)    Include Page Overlay                        (S)  ]
 +   [  (      D3..9B)    Presentation Text Object                    (S)  ]
End Page  (EPG, D3A9AF)
```

Figure 75. MO:DCA-P IS/1: Page Structure

Active Environment Group (AEG)

```
Begin Active Environment Group  (BAG, D3A8C9)
     [  (MCF,  D3AB8A)    Map Coded Font                          F2  (S)  ]  2
     [  (MPO,  D3ABD8)    Map Page Overlay                            (S)  ]  1
        (PGD,  D3A6AF)    Page Descriptor
     [  (OBD,  D3A66B)    Object Area Descriptor                           ]  3
     [  (OBP,  D3AC6B)    Object Area Position                             ]  3
        (PTD,  D3B19B)    Presentation Text Data Descriptor       F2          4
End Active Environment Group  (EAG, D3A9C9)
```

Figure 76. MO:DCA-P IS/1: Active Environment Group Structure

Graphics Object (GOCA DR/2V0)

[1] For purposes of Print Services Facility resource management, each overlay included on a page with an IPO must first be mapped to a local ID with an MPO in the AEG for that page. Note that the MPO is only specified in the AEG for a page; it is not allowed in the AEG for an overlay.

[2] For purposes of Print Services Facility resource management, an MCF mapping the same font must be specified in the AEG whenever an MCF is specified in a graphics OEG. The local ID used in the page or overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG.

[3] Used for presentation text objects only and is optional. For graphics and image objects, the OBD and OBP must be specified in the OEG associated with the graphic or image object.

[4] Required only when the associated page contains one or more presentation text objects.

```
Begin Graphics Object  (BGR, D3A8BB)
        (      D3..C7)    Object Environment Group
     [  (GAD,  D3EEBB)    Graphics Data                                    (S)  ]
End Graphics Object  (EGR, D3A9BB)
```

Figure 77. MO:DCA-P IS/1: Graphics Object Structure

Note: Refer to the *Graphics Object Content Architecture for Advanced Function Presentation Reference* for a full description of the GOCA DR/2V0 content, syntax, and semantics for MO:DCA-P IS/1.

Object Environment Group (OEG) for Graphics Object

```
Begin Object Environment Group  (BOG, D3A8C7)
        (OBD,  D3A66B)    Object Area Descriptor
        (OBP,  D3AC6B)    Object Area Position
     [  (MGO,  D3ABBB)    Map Graphics Object                                       ]
     [  (MCF,  D3AB8A)    Map Coded Font                                     F2     ]  2
        (GDD,  D3A6BB)    Graphics Data Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 78. MO:DCA-P IS/1: Object Environment Group for Graphics Object Structure

Image Object (IOCA FS10)

```
Begin Image Object  (BIM, D3A8FB)
        (      D3..C7)    Object Environment Group
     [  (IPD,  D3EEFB)    Image Picture Data                                    (S)  ]
End Image Object  (EIM, D3A9FB)
```

Figure 79. MO:DCA-P IS/1: Image Object Structure

Note: Refer to the *Image Object Content Architecture Reference* for a full description of the IOCA FS10 content, syntax, and semantics for MO:DCA-P IS/1.

Object Environment Group (OEG) for Image Object

```
Begin Object Environment Group  (BOG, D3A8C7)
        (OBD,  D3A66B)    Object Area Descriptor
        (OBP,  D3AC6B)    Object Area Position
     [  (MIO,  D3ABFB)    Map Image Object                                           ]
        (IDD,  D3A6FB)    Image Data Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 80. MO:DCA-P IS/1: Object Environment Group for Image Object Structure

Presentation Text Object (PTOCA PT1)

```
Begin Presentation Text Object  (BPT, D3A89B)
     [  (PTX,  D3EE9B)    Presentation Text Data                              (S)  ]
End Presentation Text Object  (EPT, D3A99B)
```

Figure 81. MO:DCA-P IS/1: Presentation Text Object Structure

Note: Refer to the *Presentation Text Object Content Architecture Reference* for a full description of the PTOCA PT1 content, syntax, and semantics for MO:DCA-P IS/1.

Resource Syntax Structure

The following groupings of MO:DCA structured fields identify those structured fields which may/must appear within the Begin/End structured field pair for each supported resource object. The same conventions used for the data stream syntax structure apply.

Note: Only those resources that may be included from within the data stream are described here.

Overlay

```
Begin Overlay  (BMO, D3A8DF)
       (      D3..C9)    Active Environment Group
  +  [  (      D3..BB)    Graphics Object                                     (S)  ]
  +  [  (      D3..FB)    Image Object                                        (S)  ]
  +  [  (      D3..9B)    Presentation Text Object                            (S)  ]
End Overlay  (EMO, D3A9DF)
```

Figure 82. MO:DCA-P IS/1: Overlay Structure

Permitted Structured Fields

This section describes the parameters and ranges of values supported for each of the structured fields contained in this interchange set.

The structured fields are listed alphabetically and described using tables. The table heading for each structured field contains the structured field's acronym, its three-byte hexadecimal identifier, and its full name. Also included is the page number in the document where a detailed description of the structured field can be found.

Structured Field Parameters

In general, the structured field tables contain the following information for each parameter:

1. The offset from the beginning of the data portion of the structured field or from the beginning of the triplet.
2. Values and description:
   - When a specific parameter value is required, the specific value or the range of acceptable values is specified, followed by → and an explanation or description of the parameter.

- When no specific value is required, or when a choice of values is required, the parameter name or a description of the parameter is given. If a choice of values is required, the choices are identified in the table.

3. For those parameters defined and owned by the MO:DCA architecture, occurrence is specified either as a lowercase *n* indicating that the occurrence is unlimited by the interchange set, or as a number representing the maximum number of times the parameter may appear within the containing structured field, repeating group, or triplet.
4. For those parameters defined and owned by the MO:DCA architecture, optionality is specified as:
   O Optional. The parameter may or may not appear.
   M Mandatory. The parameter must always appear.
   C Conditional. The parameter is mandatory under certain conditions, but is optional or not allowed under other conditions.

Unless a specific order is required, self-identifying parameters are listed in alphanumeric sequence by identifier and include the page number in the document where a detailed description of the parameter is located.

In general, no exception conditions are identified within the interchange set definition for the structured fields or their parameters. The page numbers provided for each structured field and each triplet provide the source for determining what exception conditions may be anticipated. However, the following general rules apply:

- For those structured fields where a parameter order is stated, if a parameter appears outside that stated order, a X'01' exception condition exists.
- If a parameter value appears that is outside the range specified for that parameter, a X'02' exception condition exists.
- If a parameter that is identified as mandatory does not appear on a specific structured field, a X'04' exception condition exists.
- Unless otherwise stated, if any unrecognized parameter or triplet appears on any structured field, a X'10' exception condition exists.

Notes:

1. Any triplet encountered on any of the *Begin* structured fields listed below that is not explicitly defined as being valid for that structured field should be ignored and should not cause an exception condition.
2. If specified, the name contained in the name parameter on an *End* structured field must match that specified in the name parameter on its matching *Begin* structured field, or a X'01' exception condition exists.

Begin Active Environment Group

BAG X'D3A8C9' Begin Active Environment Group (See "Begin Active Environment Group (BAG)" on page 104)			
0–7	Active Environment Group name (8 characters)	1	O

Begin Document

BDT X'D3A8A8' Begin Document (See "Begin Document (BDT)" on page 114)			
0–7	Document name (8 characters)	1	M
8–9	**X'0000'** → Reserved, must be binary zero	1	M
10–*n*	The following triplets, in any order:		

BDT X'D3A8A8' Begin Document (See "Begin Document (BDT)" on page 114)			
Coded Graphic Character Set Global Identifier Triplet (See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317)		1	M
0–1	X'0601' → Triplet length and identifier	1	M
2–5	Character set and code page identification	1	M
Fully Qualified Name Triplet (See "Fully Qualified Name Triplet X'02'" on page 320)		1	O
0–1	**X'*nn02*'** → Triplet length and identifier	1	M
2–3	**X'0100'** → FQN type and format. Replace first GID Name.	1	M
4–*n*	Name of the document. It may be 1 to 250 bytes in length.	1	M
MO:DCA Interchange Set Triplet (See "MO:DCA Interchange Set Triplet X'18'" on page 339)		1	M
0–1	**X'0518'** → Triplet length and identifier	1	M
2	**X'01'** → Interchange set type, presentation	1	M
3–4	**X'0900'** → Interchange set identifier (MO:DCA-P IS/1)	1	M
Object Function Set Specification Triplet (See "Object Function Set Specification Triplet X'21'" on page 346)		1	C
0–1	**X'*nn*21'** → Triplet length and identifier	1	M
2	**X'02'** → Object type, *presentation text*	1	M
3	**X'00'** → Architecture version	1	M
4–5	**X'8000'** → MO:DCA function set definition	1	M
6–7	**X'0000'** → Presentation text function set definition (PT/1)	1	M
8–*n*	Reserved, not checked	1	O
Note: One and only one instance of this triplet is *mandatory* when the data stream contains a presentation text object. If the data stream does not contain a presentation text object, this triplet should not appear.			
Object Function Set Specification Triplet (See "Object Function Set Specification Triplet X'21'" on page 346)		1	C
0–1	**X'*nn*21'** → Triplet length and identifier	1	M
2	**X'03'** → Object type, *graphics*	1	M
3	**X'00'** → Architecture version	1	M
4–5	**X'8000'** → MO:DCA function set definition	1	M
6–7	**X'4000'** → Graphics function set definition (DR/2V0)	1	M
8–*n*	Reserved, not checked	1	O
Note: One and only one instance of this triplet is *mandatory*: when the data stream contains a graphics object. If the data stream does not contain a graphics object, this triplet should not appear.			
Object Function Set Specification Triplet (See "Object Function Set Specification Triplet X'21'" on page 346)		1	C
0–1	**X'*nn*21'** → Triplet length and identifier	1	M
2	**X'06'** → Object type, *image*	1	M
3	**X'00'** → Architecture version	1	M
4–5	**X'8000'** → MO:DCA function set definition	1	M
6–7	**X'8000'** → Image function set definition (FS10)	1	M
8–*n*	Reserved, not checked	1	O

BDT X'D3A8A8' Begin Document (See “Begin Document (BDT)” on page 114)
Note: One and only one instance of this triplet is *mandatory* when the data stream contains an image object. If the data stream does not contain an image object, this triplet should not appear.

Begin Graphics Object

BGR X'D3A8BB' Begin Graphics Object (See “Begin Graphics Object (BGR)” on page 118)			
0–7	Graphics Object name (8 characters)	1	O

Begin Image Object

BIM X'D3A8FB' Begin Image Object (See “Begin Image Object (BIM)” on page 120)			
0–7	Image Object name (8 characters)	1	O

Begin Object Environment Group

BOG X'D3A8C7' Begin Object Environment Group (See “Begin Object Environment Group (BOG)” on page 133)			
0–7	Object Environment Group name (8 characters)	1	O

Begin Overlay

BMO X'D3A8DF' Begin Overlay (See “Begin Overlay (BMO)” on page 124)			
0–7	Overlay name (8 characters)	1	M

Begin Page

BPG X'D3A8AF' Begin Page (See “Begin Page (BPG)” on page 134)			
0–7	Page name (8 characters)	1	O

Begin Presentation Text Object

BPT X'D3A89B' Begin Presentation Text Object (See “Begin Presentation Text Object (BPT)” on page 139)			
0–7	Presentation Text Object name (8 characters)	1	O

End Active Environment Group

EAG X'D3A9C9' End Active Environment Group (See “End Active Environment Group (EAG)” on page 151)			
0–7	Active Environment Group name (8 characters)	1	O

End Document

EDT X'D3A9A8' End Document (See “End Document (EDT)” on page 157)			
0–7	Document name (8 characters)	1	O

End Graphics Object

EGR X'D3A9BB' End Graphics Object (See “End Graphics Object (EGR)” on page 159)			
0–7	Graphics Object name (8 characters)	1	O

End Image Object

EIM X'D3A9FB' End Image Object (See "End Image Object (EIM)" on page 160)			
0–7	Image Object name (8 characters)	1	O

End Object Environment Group

EOG X'D3A9C7' End Object Environment Group (See "End Object Environment Group (EOG)" on page 165)			
0–7	Object Environment Group name (8 characters)	1	O

End Overlay

EMO X'D3A9DF' End Overlay (See "End Overlay (EMO)" on page 162)			
0–7	Overlay name (8 characters)	1	O

End Page

EPG X'D3A9AF' End Page (See "End Page (EPG)" on page 166)			
0–7	Page name (8 characters)	1	O

End Presentation Text Object

EPT X'D3A99B' End Presentation Text Object (See "End Presentation Text Object (EPT)" on page 168)			
0–7	Presentation Text Object name (8 characters)	1	O

Graphics Data

GAD X'D3EEBB' Graphics Data (See "Graphics Data (GAD)" on page 173)	
0–*n*	Up to 8192 bytes of graphics data as defined by GOCA DR/2V0

Graphics Data Descriptor

GDD X'D3A6BB' Graphics Data Descriptor (See "Graphics Data Descriptor (GDD)" on page 174)	
0–*n*	Graphics descriptor data as defined by GOCA

Image Data Descriptor

IDD X'D3A6FB' Image Data Descriptor (See "Image Data Descriptor (IDD)" on page 175)	
0–*n*	Image descriptor data as defined by IOCA FS10

Image Picture Data

IPD X'D3EEFB' Image Picture Data (See "Image Picture Data (IPD)" on page 190)	
0–*n*	Up to 8192 bytes of image segment data as defined by IOCA FS10

Include Page Overlay

IPO X'D3AFD8' Include Page Overlay (See "Include Page Overlay (IPO)" on page 194)			
0–7	Page overlay reference name.	1	M

IPO X'D3AFD8' Include Page Overlay (See “Include Page Overlay (IPO)” on page 194)			
8–10	Page overlay origin, X-coordinate. It must be one of the following: **X'000000'–X'001555' →** In the range of 0 to 5461 when using 240 units per inch for the page X measurement units **X'000000'–X'007FFF' →** In the range of 0 to 32767 when using 1440 units per inch for the page X measurement units	1	M
11–13	Page overlay origin, Y-coordinate. It must be one of the following: **X'000000'–X'001555' →** In the range of 0 to 5461 when using 240 units per inch for the page Y measurement units **X'000000'–X'007FFF' →** In the range of 0 to 32767 when using 1440 units per inch for the page Y measurement units	1	M

Invoke Medium Map

IMM X'D3ABCC' Invoke Medium Map (See “Invoke Medium Map (IMM)” on page 178)			
0–7	External name of the medium map to be invoked (8 characters)	1	M

Map Coded Font, Format 2

MCF X'D3AB8A' Map Coded Font (See “Map Coded Font (MCF) Format 2” on page 213)				
0–1	**X’00*nn*’** → Length of this repeating group		254	M
2–*n*	The following triplets, in any order:			
	Fully Qualified Name Triplet (See “Fully Qualified Name Triplet X'02'” on page 320) **Note:** See “MCF Font Names” on page 438 for details.		2	M
	0–1	X'0C02' → Triplet length and identifier	1	M
	2	The FQN type. It must be one of the following: **X'84'** → Coded Font Reference **X'85'** → Code Page Reference **X'86'** → Font Character Set Reference	1	M
	3	**X'00'** → FQN format	1	M
	4–11	External name of the coded font, code page, or font character set.	1	M
	Fully Qualified Name Triplet (See “Fully Qualified Name Triplet X'02'” on page 320)		1	O
	0–1	**X’*nn*02’** → Triplet length and identifier	1	M
	2–3	**X'0800'** → FQN type and format, Font Typeface Name	1	M
	4–*n*	External name of the font typeface. It may be 1 to 32 bytes in length.	1	M
	Font Descriptor Specification Triplet (See “Font Descriptor Specification Triplet X'1F'” on page 341)		1	O
	0–1	**X'141F'** → Triplet length and identifier	1	M
	2	**X'01'–X'09'** → Font Weight Class. It must be in the range of 1 to 9.	1	M
	3	**X'01'–X'09'** → Font Width Class. It must be in the range of 1 to 9.	1	M
	4–5	**X'0000'–X'7FFF'** → Font Height. It must be in the range of 0 to 32767 1440ths of an inch.	1	M
	6–7	**X'0000'–X'7FFF'** → Font Width. It must be in the range of 0 to 32767 1440ths of an inch.	1	M

MCF X'D3AB8A' Map Coded Font (See “Map Coded Font (MCF) Format 2” on page 213)			
8	Font Descriptor Flags, as follows: **Bit** **Description** **0** Italics **1** Underscored **2** Reserved, must be B'0' **3** Hollow **4** Overstruck **5** Proportional **6** Kerned characters (pairwise) **7** Reserved, must be B'0'	1	M
9–19	Reserved	1	M
Font Coded Graphic Character Set Global Identifier Triplet (See “Font Coded Graphic Character Set Global Identifier Triplet X'20'” on page 345)		1	O
0–1	**X'0620'** → Triplet length and identifier	1	M
2–5	The GCSGID and CPGID for the font	1	M
Resource Local Identifier Triplet (See “Resource Local Identifier Triplet X'24'” on page 350)		1	M
0–1	**X'0424'** → Architecture version	1	M
2	**X'05'** → Resource type, coded font	1	M
3	Resource Local Identifier. It must be one of the following: **X'01'–X'7F'** → It must be in the range of 1 to 127 when used for mapping a font. **X'FE'** → It must be 254 when used for resource management purposes in the AEG.	1	M
Resource Section Number Triplet (See “Resource Section Number Triplet X'25'” on page 351)		1	O
0–1	**X'0325'** → Triplet length and identifier	1	M
2	Resource Section Number. It must be one of the following: **X'00'** → It must be 0 when referencing an EBCDIC Presentation single-byte coded font (encoding scheme ID X’61*xx*’) or all sections of an EBCDIC Presentation double-byte coded font (encoding scheme ID X’62*xx*’). **X'41'–X'FE'** → It must be in the range of 65 to 254 when referencing a specific section of an EBCDIC Presentation double-byte coded font (encoding scheme ID X’62*xx*’).	1	M
Character Rotation Triplet (See “Character Rotation Triplet X'26'” on page 352)		1	O
0–1	**X'0426'** → Triplet length and identifier	1	M
2–3	Character Rotation. It must be one of the following: **X'0000'** → 0-degree character rotation **X'2D00'** → 90-degree character rotation **X'5A00'** → 180-degree character rotation **X'8700'** → 270-degree character rotation	1	M

MCF Font Names: The MCF must have one of the following:

- A type X’84’ (Coded Font Reference) Fully Qualified Name (X’02’) triplet. To support existing products, the coded font name must be specified as a global resource identifier (GRID). For a definition of the GRID, see “Global Resource Identifier (GRID) Definition” on page 329.
- Both a type X’85’ (Code Page Name Reference) and a type X’86’ (Font Character Set Name Reference) Fully Qualified Name (X’02’) triplet. To support existing products, the names of the code page and font character set must be eight characters in length and must match the external names of these objects in their respective resource libraries.

Map Graphics Object

MGO X'D3ABBB' Map Graphics Object (See “Map Graphics Object (MGO)” on page 245)				
0–1	**X'0005'** →Length of this repeating group is 5 bytes		1	M
2–4	The following triplet:			
	Mapping Option Triplet (See “Mapping Option Triplet X'04'” on page 332)		1	M
	0–1	**X'0304'** →Triplet length and identifier	1	M
	2	Output Option. It must be one of the following: **X'10'** → Position and trim **X'20'** → Scale to fit **X'30'** → Center and trim	1	M

Note: If this structured field is not specified, the architected default is *scale to fit*.

Map Image Object

MIO X'D3ABFB' Map Image Object (See “Map Image Object (MIO)” on page 246)				
0–1	**X'0005'** →Length of this repeating group is 5 bytes		1	M
2–4	The following triplet:			
	Mapping Option Triplet (See “Mapping Option Triplet X'04'” on page 332)		1	M
	0–1	**X'0304'** →Triplet length and identifier	1	M
	2	Output Option. It must be one of the following: **X'10'** → Position and trim **X'20'** → Scale to fit **X'30'** → Center and trim	1	M

Note: If this structured field is not specified, the architected default is *scale to fit*.

Map Page Overlay

MPO X'D3ABD8' Map Page Overlay (See “Map Page Overlay (MPO)” on page 265)				
0–1	**X'0012'** →Length of this repeating group is 18 bytes		127	M
2–17	The following triplet:			
	Fully Qualified Name Triplet (See “Fully Qualified Name Triplet X'02'” on page 320)		1	M
	0–1	**X'0C02'** →Triplet length and identifier	1	M
	2–3	**X'8400'** →FQN type and format, reference to overlay	1	M
	4–11	External name of the overlay.	1	M
	Resource Local Identifier Triplet (See “Resource Local Identifier Triplet X'24'” on page 350)		1	M

MPO X'D3ABD8' Map Page Overlay (See "Map Page Overlay (MPO)" on page 265)				
	0–1	**X'0424'** →Triplet length and identifier	1	M
	2	**X'02'** →Resource type, page overlay	1	M
	3	**X'01'–X'7F'** →Resource Local Identifier. It must be in the range of 1 to 127.	1	M

No Operation

NOP X'D3EEEE' No Operation (See "No Operation (NOP)" on page 269)	
0–*n*	Up to 32759 bytes of data.

Object Area Descriptor

OBD X'D3A66B' Object Area Descriptor (See "Object Area Descriptor (OBD)" on page 270)				
0–*n*	The following triplets, in any order:			
	Descriptor Position Triplet (See "Descriptor Position Triplet X'43'" on page 355)		1	M
	0–1	**X'0343'** →Triplet length and identifier	1	M
	2	**X'01'–X'7F'** →Descriptor position ID. It must be in the range of 1 to 127.	1	M
	Measurement UnitsTriplet (See "Measurement Units Triplet X'4B'" on page 364)		1	M
	0–1	**X'084B'** →Triplet length and identifier	1	M
	2–3	**X'0000'** →Object area measurement units base for X and Y	1	M
	4–5	Object area measurement units value for X. It must be one of the following: **X'0960'** → 2400 units per unit base (240 units per inch) **X'3840'** → 14400 units per unit base (1440 units per inch)		
	6–7	Object area measurement units value for Y. It must be identical to bytes 4–5.	1	M
	Object Area Size Triplet (See "Object Area Size Triplet X'4C'" on page 365)		1	M
	0–1	**X'094C'** →Triplet length and identifier	1	M
	2	**X'02'** →Type, actual object area size	1	M
	3–5	Object area size in the X direction. It must be one of the following: **X'000001'–X'001555'** → In the range of 1 to 5461 when using 240 units per inch for the object area X measurement units **X'000001'–X'007FFF'** → In the range of 1 to 32767 when using 1440 units per inch for the object area X measurement units	1	M
	6–8	Object area size in the Y direction. It must be one of the following: **X'000001'–X'001555'** → In the range of 1 to 5461 when using 240 units per inch for the object area Y measurement units **X'000001'–X'007FFF'** → In the range of 1 to 32767 when using 1440 units per inch for the object area Y measurement units	1	M

Note: If the presentation text Object Area Descriptor structured field appears in the AEG, the measurement units and extents specified on it must match those specified on the Page Descriptor structured field, or a X'01' exception condition exists. If the presentation text Object Area Descriptor structured field is omitted, the architected default is to use the measurement units and extents specified on the Page Descriptor structured field for the presentation text object area. Thus, the presentation text object area and the page are always the same size and points within their respective coordinate systems are always coincident.

Object Area Position

OBP X'D3AC6B' Object Area Position (See "Object Area Position (OBP)" on page 272)			
0	**X'01'–X'7F'** →Object Area Position ID. It must be in the range of 1 to 127.	1	M
1	**X'17'** →Length of this repeating group is 23 bytes	1	M
2–4	Object area origin for X. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page or overlay X measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page or overlay X measurement units	1	M
5–7	Object area origin for Y. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page or overlay Y measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page or overlay Y measurement units	1	M
8–11	Object Area orientation, X and Y coordinates. It must be one of the following: **X'0000 2D00'** → X=0 degrees, Y=90 degrees **X'2D00 5A00'** → X=90 degrees, Y=180 degrees **X'5A00 8700'** → X=180 degrees, Y=270 degrees **X'8700 0000'** → X=27 degrees, Y=0 degrees	1	M
12	**X'00'** →Reserved, must be binary zero	1	M
13–15	Object content origin for X. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the object area X measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the object area X measurement units	1	M
16–18	Object content origin for Y. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the object area Y measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the object area Y measurement units	1	M
19–20	**X'0000'** →Object content orientation, X (0 degrees)	1	M
21–22	**X'2D00'** →Object content orientation, Y (90 degrees)	1	M
23	Referenced coordinate system. It must be one of the following: **X'00'** → Current coordinate system **X'01'** → Page or overlay coordinate system	1	M

Notes:

1. If the presentation text Object Area Position structured field appears in the AEG, the X and Y values for the object area origin and the object content origin must be set to zero, or a X'01' exception condition exists. If the presentation text

Object Area Position structured field is omitted, the architected default is to set the X and Y values for the object area origin and the object content origin to zero. For presentation text, the data object presentation space origin is positioned coincident with the object content origin. Thus, the presentation text object presentation space, the presentation text object area, and the page always have the same origin.

2. If the presentation text OBP appears in the AEG, the object area orientation must be set to X'0000 2D00' (0°,90°). If it is omitted, the architected default is to set the object area orientation to X'0000 2D00' (0°,90°).
3. For this interchange set, the values X'00' and X'01' in byte 23 specify the same function since positioning with respect to a page segment offset is not part of the interchange set definition. That is, both values specify that the object area is to be positioned with respect to the including page or overlay coordinate system.

Page Descriptor

PGD X'D3A6AF' Page Descriptor (See "Page Descriptor (PGD)" on page 279)			
0–1	**X'0000'** →Page measurement units base for X and Y	1	M
2–3	Page measurement units value for X. It must be one of the following: **X'0960'** → 2400 units per unit base (240 units per inch) **X'3840'** → 14400 units per unit base (1440 units per inch)	1	M
4–5	Page measurement units value for Y. It must be identical to bytes 2–3.	1	M
6–8	Page size in the X direction. It must be one of the following: **X'000001'–X'001555'** → In the range of 1 to 5461 when using 240 units per inch for the page X measurement units **X'000001'–X'007FFF'** → In the range of 1 to 32767 when using 1440 units per inch for the page X measurement units	1	M
9–11	Page size in the Y direction. It must be one of the following: **X'000001'–X'001555'** → In the range of 1 to 5461 when using 240 units per inch for the page Y measurement units **X'000001'–X'007FFF'** → In the range of 1 to 32767 when using 1440 units per inch for the page Y measurement units	1	M
12–14	**X'000000'** →Reserved, must be binary zero	1	M

Application Note: The IS/1 and IS/2 interchange set definitions limit the page size to 22.75 inches in the X and Y directions. To specify a larger page size, 240 units per inch should be specified in the PGD for the page measurement units. Using a range of 1 to 32767, this will allow a maximum page size in the X and Y directions of 136.5 inches, is supported by all IPDS printers, and keeps the complete page presentation space within the range of two-byte addressing parameters in the IPDS and PTOCA architectures.

Presentation Text Data

PTX X'D3EE9B' Presentation Text Data (See "Presentation Text Data (PTX)" on page 309)	
0–*n*	Up to 8192 bytes of presentation text data as defined by PTOCA PT1

Presentation Text Data Descriptor, Format 2

PTD X'D3B19B' Presentation Text Data Descriptor (See "Presentation Text Data Descriptor (PTD) Format 2" on page 308)	
0–*n*	Presentation text descriptor data as defined by PTOCA

Note: When the PTD is included in the AEG for a page, some AFP print servers require that the measurement units in the PTD match the measurement units in the Page Descriptor (PGD). It is therefore strongly recommended that whenever the PTD is included in the AEG, the same measurement units are specified in both the PTD and PGD.

MO:DCA Presentation Interchange Set 2

This section defines the MO:DCA Presentation Interchange Set 2 (MO:DCA-P IS/2) used for presentation documents.

For information on the level of function required for the OCAs included in this interchange set, refer to the MO:DCA environment appendix in the following IBM documents:

BCOCA	*Bar Code Object Content Architecture Reference*, S544-3766
GOCA	*Graphics Object Content Architecture for Advanced Function Presentation Reference*, S544-5498
IOCA	*Image Object Content Architecture Reference*, SC31-6805
PTOCA	*Presentation Text Object Content Architecture Reference*, SC31-6803

Note: MO:DCA-P IS/2 is a proper superset of MO:DCA-P IS/1 and therefore contains all of the function defined by MO:DCA-P IS/1. Generators of data streams that contain only MO:DCA-P IS/1 function may choose to identify those data streams as either MO:DCA-P IS/1 or MO:DCA-P IS/2 data streams. However, be aware that identifying them as MO:DCA-P IS/2 potentially limits the receivers of the data stream to only those that claim to support MO:DCA-P IS/2.

Data Stream Syntax Structure

The groupings of MO:DCA structured fields that follow identify those structured fields which appear within each begin-end structured field pair or state. This section specifies the structured fields allowed within a MO:DCA Presentation Interchange Set 2 data stream and shows both the MO:DCA state hierarchy and the validity of structured fields within each state.

If a structured field that is not identified as being part of this interchange set appears anywhere within the data stream, a X’40’ exception condition exists. If a structured field appears within any state where it is not permitted, or if it appears out of the stated order or more than the permitted number of times, a X’20’ exception condition exists. If a structured field that is identified as required does not appear within a specific state, a X’08’ exception condition exists.

The conventions used in these structured field groupings are:

( ) The structured field acronym and identifier are shown in parentheses. The presence of dots or periods in the identifier indicates that the item is not a structured field, but instead is a structure, for example a page. The structure is composed of an assortment of structured fields, and is defined separately.

[ ] Brackets indicate optional structured fields. When a structured field is shown without brackets, it *must* appear between the begin and end structured fields.

+ Plus signs indicate structured fields may appear in any order relative to those that precede or succeed it except when the preceding or succeeding structured field does not have a plus (+) sign. Then the order is as listed.

(S) The enclosed (S) indicates that the structured field may be repeated. When present on a required structured field, at least one occurrence of the structured field is required, but multiple instances of it may occur.

F2 An F2 indicates that the structured field is a format two structured field. See “Structured Field Formats” on page 25 for further details.

Notes:

1. The Begin Document and End Document structured fields are required in a MO:DCA data stream.
2. The No Operation structured field may appear within any begin-end domain and thus is not listed in the structured field groupings.
3. The architecture that owns and controls the content of each of the data and resource objects carried in a MO:DCA data stream is identified in the following structured field groupings. Please refer to the referenced documentation for further details.
4. The Flag byte (byte 5) in the Structured Field Introducer (SFI) must be set to X'00'. MO:DCA-P IS/2 does not support SFI extension, structured field segmentation, or structured field padding.

Document

```
Begin Document  (BDT, D3A8A8)
    [  (      D3..A7)    Document Index                                      ]
 +  [  (IMM,  D3ABCC)    Invoke Medium Map                              (S)  ]
 +  [  (      D3..AF)    Page                                           (S)  ]
End Document  (EDT, D3A9A8)
```

Figure 83. MO:DCA-P IS/2: Document Structure

Document Index

```
Begin Document Index  (BDI, D3A8A7)
        (IEL,  D3B2A7)    Index Element                                  (S)
End Document Index  (EDI, D3A9A7)
```

Figure 84. MO:DCA-P IS/2: Document Index Structure

Note: These structured fields are used for informational purposes only. Thus, there is no requirement that these fields be processed by a receiver. A compliant receiver must be able to recognize the document index structure, but it may elect to simply skip the entire structure without processing its content.

Resource Group

```
Begin Resource Group  (BRG, D3A8C6)
  +  [  (      D3..DF)    Overlay                                        (S)  ]
End Resource Group  (ERG, D3A9C6)
```

Figure 85. MO:DCA-P IS/2: Resource Group Structure

Page

```
Begin Page  (BPG, D3A8AF)
     [  (      D3..C6)    Resource Group                                  ]
        (      D3..C9)    Active Environment Group
  +  [  (      D3..EB)    Bar Code Object                           (S)  ]
  +  [  (      D3..BB)    Graphics Object                           (S)  ]
  +  [  (      D3..FB)    Image Object                              (S)  ]
  +  [  (IPO,  D3AFD8)    Include Page Overlay                      (S)  ] [5]
  +  [  (      D3..9B)    Presentation Text Object                  (S)  ]
End Page  (EPG, D3A9AF)
```

Figure 86. MO:DCA-P IS/2: Page Structure

Overlay

```
Begin Overlay  (BMO, D3A8DF)
        (      D3..C9)    Active Environment Group
  +  [  (      D3..EB)    Bar Code Object                           (S)  ]
  +  [  (      D3..BB)    Graphics Object                           (S)  ]
  +  [  (      D3..FB)    Image Object                              (S)  ]
  +  [  (      D3..9B)    Presentation Text Object                  (S)  ]
End Overlay  (EMO, D3A9DF)
```

Figure 87. MO:DCA-P IS/2: Overlay Structure

Active Environment Group

```
Begin Active Environment Group  (BAG, D3A8C9)
     [  (MCF,  D3AB8A)    Map Coded Font                        F2  (S)  ] [8]
     [  (MPO,  D3ABD8)    Map Page Overlay                          (S)  ] [5]
        (PGD,  D3A6AF)    Page Descriptor
     [  (OBD,  D3A66B)    Object Area Descriptor                         ] [6]
     [  (OBP,  D3AC6B)    Object Area Position                           ] [6]
     [  (PTD,  D3B19B)    Presentation Text Data Descriptor     F2 [7]
End Active Environment Group  (EAG, D3A9C9)
```

Figure 88. MO:DCA-P IS/2: Active Environment Group Structure

[5] For purposes of Print Services Facility resource management, each overlay included on a page with an IPO must first be mapped to a local ID with an MPO in the AEG for that page. Note that the MPO is only specified in the AEG for a page; it is not allowed in the AEG for an overlay.

[6] Used for presentation text objects only and is optional. For graphics, bar code, and image objects, the OBD and OBP must be specified in the OEG associated with the graphic, bar code, or image object.

[7] Required only when the associated page contains one or more presentation text objects.

[8] For purposes of Print Services Facility resource management, an MCF mapping the same font must be specified in the AEG whenever an MCF is specified in a bar code or graphics OEG. The local ID used in the page or overlay AEG need not match the ID in the object OEG. ID X'FE' may be used in the AEG for fonts mapped in the AEG solely due to their presence in an object's OEG.

Bar Code Object (BCOCA BCD1)

```
Begin Bar Code Object  (BBC, D3A8EB)
       (      D3..C7)    Object Environment Group
    [  (BDA,  D3EEEB)    Bar Code Data                                     (S)  ]
End Bar Code Object  (EBC, D3A9EB)
```

Figure 89. MO:DCA-P IS/2: Bar Code Object Structure

Note: Refer to the *Bar Code Object Content Architecture Reference* for a full description of the BCOCA content, syntax, and semantics for MO:DCA-P IS/2.

Object Environment Group (OEG) for Bar Code Object

```
Begin Object Environment Group  (BOG, D3A8C7)
       (OBD,  D3A66B)    Object Area Descriptor
       (OBP,  D3AC6B)    Object Area Position
    [  (MBC,  D3ABEB)    Map Bar Code Object                                     ]
    [  (MCF,  D3AB8A)    Map Coded Font                                F2  (S)  ]8
       (BDD,  D3A6EB)    Object Area Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 90. MO:DCA-P IS/2: Object Environment Group for Bar Code Object Structure

Graphics Object (GOCA DR/2V0)

```
Begin Graphics Object  (BGR, D3A8BB)
       (      D3..C7)    Object Environment Group
    [  (GAD,  D3EEBB)    Graphics Data                                     (S)  ]
End Graphics Object  (EGR, D3A9BB)
```

Figure 91. MO:DCA-P IS/2: Graphics Object Structure

Note: Refer to the *Graphics Object Content Architecture for Advanced Function Presentation Reference* for a full description of the GOCA DR/2V0 content, syntax, and semantics for MO:DCA-P.

Object Environment Group (OEG) for Graphics Object

```
Begin Object Environment Group  (BOG, D3A8C7)
       (OBD,  D3A66B)    Object Area Descriptor
       (OBP,  D3AC6B)    Object Area Position
    [  (MGO,  D3ABBB)    Map Graphics Object                                    ]
    [  (MCF,  D3AB8A)    Map Coded Font                               F2  (S)  ] 8
       (GDD,  D3A6BB)    Graphics Data Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 92. MO:DCA-P IS/2: Object Environment Group for Graphics Object Structure

Image Object (IOCA FS10 or FS11)

```
Begin Image Object  (BIM, D3A8FB)
        (      D3..C7)    Object Environment Group
     [  (IPD,  D3EEFB)    Image Picture Data                              (S)  ]
End Image Object  (EIM, D3A9FB)
```

Figure 93. MO:DCA-P IS/2: Image Object Structure

Note: Refer to the *Image Object Content Architecture Reference* for a full description of the IOCA FS10 and FS11 content, syntax, and semantics for MO:DCA-P IS/2.

Object Environment Group (OEG) for Image Object

```
Begin Object Environment Group  (BOG, D3A8C7)
        (OBD,  D3A66B)    Object Area Descriptor
        (OBP,  D3AC6B)    Object Area Position
     [  (MIO,  D3ABFB)    Map Image Object                                     ]
        (IDD,  D3A6FB)    Image Data Descriptor
End Object Environment Group  (EOG, D3A9C7)
```

Figure 94. MO:DCA-P IS/2: Object Environment Group for Image Object Structure

Presentation Text Object (PTOCA PT1)

```
Begin Presentation Text Object  (BPT, D3A89B)
     [  (PTX,  D3EE9B)    Presentation Text Data                          (S)  ]
End Presentation Text Object  (EPT, D3A99B)
```

Figure 95. MO:DCA-P IS/2: Presentation Text Object Structure

Note: Refer to the *Presentation Text Object Content Architecture Reference* for a full description of the PTOCA PT1 content, syntax, and semantics for MO:DCA-P.

Permitted Structured Fields

This section describes the parameters and ranges of values supported for each of the structured fields contained in this interchange set.

The structured fields are listed alphabetically and described using tables. The table heading for each structured field contains the structured field's acronym, its three-byte hexadecimal identifier, and its full name. Also included is the page number in the document where a detailed description of the structured field can be found.

Structured Field Parameters

In general, the structured field tables contain the following information for each parameter:

1. The offset from the beginning of the data portion of the structured field or from the beginning of the triplet.
2. Values and description:

- When a specific parameter value is required, the specific value or the range of acceptable values is specified, followed by → and an explanation or description of the parameter.
- When no specific value is required, or when a choice of values is required, the parameter name or a description of the parameter is given. If a choice of values is required, the choices are identified in the table.

3. For those parameters defined and owned by the MO:DCA architecture, occurrence is specified either as a lowercase *n* indicating that the occurrence is unlimited by the interchange set, or as a number representing the maximum number of times the parameter may appear within the containing structured field, repeating group, or triplet.
4. For those parameters defined and owned by the MO:DCA architecture, optionality is specified as:
   O Optional. The parameter may or may not appear.
   M Mandatory. The parameter must always appear.
   R Retired. A *receiver must be able* to receive this parameter, but a *generator should not* generate it.

Unless a specific order is required, self-identifying parameters are listed in alphanumeric sequence by identifier and include the page number in the document where a detailed description of the parameter is located.

In general, no exception conditions are identified within the interchange set definition for the structured fields or their parameters. The page numbers provided for each structured field and each triplet provide the source for determining what exception conditions may be anticipated. However, the following general rules apply:

- For those structured fields where a parameter order is stated, if a parameter appears outside that stated order, a X’01’ exception condition exists.
- If a parameter value appears that is outside the range specified for that parameter, a X’02’ exception condition exists.
- If a parameter that is identified as mandatory does not appear on a specific structured field, a X’04’ exception condition exists.
- Unless otherwise stated, if any unrecognized parameter or triplet appears on any structured field, a X’10’ exception condition exists.

Notes:

1. Any triplet encountered on any of the *Begin* structured fields listed below that is not explicitly defined as being valid for that structured field should be ignored and should not cause an exception condition.
2. If specified, the name contained in the name parameter on an *End* structured field must match that specified in the name parameter on its matching *Begin* structured field, or a X'01' exception condition exists.

Bar Code Data

BDA X'D3EEEB' Bar Code Data (See “Bar Code Data (BDA)” on page 109)	
0–*n*	Up to 8192 bytes of bar code data as defined by BCOCA BCD1

Bar Code Data Descriptor

BDD X'D3A6EB' Bar Code Data Descriptor (See “Bar Code Data Descriptor (BDD)” on page 110)	
0–*n*	Bar Code descriptor data as defined by BCOCA BCD1

Begin Active Environment Group

BAG X'D3A8C9' Begin Active Environment Group (See “Begin Active Environment Group (BAG)” on page 104)			
0–7	Active Environment Group name (8 characters)	1	O

Begin Bar Code Object

BBC X'D3A8EB' Begin Bar Code Object (See “Begin Bar Code Object (BBC)” on page 105)			
0–7	Bar Code Object name (8 characters)	1	O

Begin Document Index

BDI X'D3A8A7' Begin Document Index (See “Begin Document Index (BDI)” on page 112)			
0–7	Document Index name (8 characters)	1	O

Begin Document

BDT X'D3A8A8' Begin Document (See “Begin Document (BDT)” on page 114)				
0–7	Document name (8 characters)		1	M
8–9	**X'0000'** → Reserved, must be binary zero		1	M
10–*n*	The following triplets, in any order:			
Coded Graphic Character Set Global Identifier Triplet (See “Coded Graphic Character Set Global Identifier Triplet X'01'” on page 317)				
	0–1	**X'0601'** → Triplet length and identifier	1	M
	2–5	Character set and code page identification	1	M
	Fully Qualified Name (See “Fully Qualified Name Triplet X'02'” on page 320)		1	O
	0–1	**X’*nn*02’** → Triplet length and identifier	1	M
	2–3	**X'0100'** → FQN type and format. Replace first GID Name.	1	M
	4–*n*	Name of the document. It may be 1 to 250 bytes in length.	1	M
	MO:DCA Interchange Set Triplet (See “MO:DCA Interchange Set Triplet X'18'” on page 339)		1	M
	0–1	**X'0518'** → Triplet length and identifier	1	M
	2	**X'01'** → Interchange set type, presentation	1	M
	3–4	**X'0C00'** → Interchange set identifier (MO:DCA-P IS/2)	1	M
	Object Function Set Specification Triplet (See “Object Function Set Specification Triplet X'21'” on page 346)		1	R
	0–1	**X’*nn*21’** → Triplet length and identifier	1	M
	2	**X'02'** → Object type, *presentation text*	1	M
	3	**X'00'** → Architecture version	1	M
	4–5	**X'8000'** → MO:DCA function set definition	1	M
	6–7	**X'0000'** → Presentation text function set definition (PT/1)	1	M
	8–*n*	Reserved, not checked	1	O
Note: For compatibility with MO:DCA-P IS/1, one instance of this triplet is *permitted* when the data stream contains a PT1 presentation text object. However, this triplet has been retired and should not be included in MO:DCA-P IS/2 data streams.				

BDT X'D3A8A8' Begin Document (See "Begin Document (BDT)" on page 114)			
	Object Function Set Specification Triplet (See "Object Function Set Specification Triplet X'21'" on page 346)	1	R
0–1	**X'*nn*21'** → Triplet length and identifier	1	M
2	**X'03'** → Object type, *graphics*	1	M
3	**X'00'** → Architecture version	1	M
4–5	**X'8000'** → MO:DCA function set definition	1	M
6–7	**X'4000'** → Graphics function set definition (DR/2V0)	1	M
8–*n*	Reserved, not checked	1	O
Note: For compatibility with MO:DCA-P IS/1, one instance of this triplet is *permitted* when the data stream contains a DR/2V0 graphics object. However, this triplet has been retired and should not be included in MO:DCA-P IS/2 data streams.			
	Object Function Set Specification Triplet (See "Object Function Set Specification Triplet X'21'" on page 346)	1	R
0–1	**X'*nn*21'** → Triplet length and identifier	1	M
2	**X'06'** → Object type, *image*	1	M
3	**X'00'** → Architecture version	1	M
4–5	**X'8000'** → MO:DCA function set definition	1	M
6–7	**X'8000'** → Image function set definition (FS10)	1	M
8–*n*	Reserved, not checked	1	O
For compatibility with MO:DCA-P IS/1, one instance of this triplet is *permitted* when the data stream contains an FS10 image object. However, this triplet has been retired and should not be included in MO:DCA-P IS/2 data streams. For this reason, no value has been provided for IOCA FS11.			

Begin Graphics Object

BGR X'D3A8BB' Begin Graphics Object (See "Begin Graphics Object (BGR)" on page 118)			
0–7	Graphics Object name (8 characters)	1	O

Begin Image Object

BIM X'D3A8FB' Begin Image Object (See "Begin Image Object (BIM)" on page 120)			
0–7	Image Object name (8 characters)	1	O

Begin Object Environment Group

BOG X'D3A8C7' Begin Object Environment Group (See "Begin Object Environment Group (BOG)" on page 133)			
0–7	Object Environment Group name (8 characters)	1	O

Begin Overlay

BMO X'D3A8DF' Begin Overlay (See "Begin Overlay (BMO)" on page 124)			
0–7	Overlay name (8 characters)	1	M

Begin Page

BPG X'D3A8AF' Begin Page (See "Begin Page (BPG)" on page 134)			
0–7	Page name (8 characters)	1	O

Begin Presentation Text Object

BPT X'D3A89B' Begin Presentation Text Object (See "Begin Presentation Text Object (BPT)" on page 139)			
0–7	Presentation Text Object name (8 characters)	1	O

Begin Resource Group

BRG X'D3A8C6' Begin Resource Group (See "Begin Resource Group (BRG)" on page 141)			
0–7	Resource Group name (8 characters)	1	O

End Active Environment Group

EAG X'D3A9C9' End Active Environment Group (See "End Active Environment Group (EAG)" on page 151)			
0–7	Active Environment Group name (8 characters)	1	O

End Bar Code Object

EBC X'D3A9EB' End Bar Code Object (See "End Bar Code Object (EBC)" on page 152)			
0–7	Bar Code Object name (8 characters)	1	O

End Document Index

EDI X'D3A9A7' End Document Index (See "End Document Index (EDI)" on page 156)			
0–7	Document Index name (8 characters)	1	O

End Document

EDT X'D3A9A8' End Document (See "End Document (EDT)" on page 157)			
0–7	Document name (8 characters)	1	O

End Graphics Object

EGR X'D3A9BB' End Graphics Object (See "End Graphics Object (EGR)" on page 159)			
0–7	Graphics Object name (8 characters)	1	O

End Image Object

EIM X'D3A9FB' End Image Object (See "End Image Object (EIM)" on page 160)			
0–7	Image Object name (8 characters)	1	O

End Object Environment Group

EOG X'D3A9C7' End Object Environment Group (See "End Object Environment Group (EOG)" on page 165)			
0–7	Object Environment Group name (8 characters)	1	O

End Overlay

EMO X'D3A9DF' End Overlay (See "End Overlay (EMO)" on page 162)			
0–7	Overlay name (8 characters)	1	O

End Page

EPG X'D3A9AF' End Page (See "End Page (EPG)" on page 166)			
0–7	Page name (8 characters)	1	O

End Presentation Text Object

EPT X'D3A99B' End Presentation Text Object (See "End Presentation Text Object (EPT)" on page 168)			
0–7	Presentation Text Object name (8 characters)	1	O

End Resource Group

ERG X'D3A9C6' End Resource Group (See "End Resource Group (ERG)" on page 170)			
0–7	Resource Group name (8 characters)	1	O

Graphics Data

GAD X'D3EEBB' Graphics Data (See "Graphics Data (GAD)" on page 173)	
0–*n*	Up to 8192 bytes of graphics data as defined by GOCA DR/2V0

Graphics Data Descriptor

GDD X'D3A6BB' Graphics Data Descriptor (See "Graphics Data Descriptor (GDD)" on page 174)	
0–*n*	Graphics descriptor data as defined by GOCA

Image Data Descriptor

IDD X'D3A6FB' Image Data Descriptor (See "Image Data Descriptor (IDD)" on page 175)	
0–*n*	Image descriptor data as defined by IOCA FS10 and FS11

Image Picture Data

IPD X'D3EEFB' Image Picture Data (See "Image Picture Data (IPD)" on page 190)	
0–*n*	Up to 8192 bytes of image segment data as defined by IOCA FS10 or FS11

Include Page Overlay

IPO X'D3AFD8' Include Page Overlay (See "Include Page Overlay (IPO)" on page 194)			
0–7	Page overlay reference name.	1	M

IPO X'D3AFD8' Include Page Overlay (See "Include Page Overlay (IPO)" on page 194)				
8–10		Page overlay origin, X-coordinate. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page X measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page X measurement units	1	M
11–13		Page overlay origin, Y-coordinate. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page Y measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page Y measurement units	1	M
14–15		**X'0000'** → Overlay orientation of 0 degrees	1	O
16–*n*		The following triplets, in any order:		
		Page Overlay Conditional Processing Triplet (See "Page Overlay Conditional Processing Triplet X'46'" on page 361)	*n*	O
	0–1	**X'*nn*46'** Triplet length and identifier	1	M
	2	Page Overlay Type. It must be one of the following: **X'00'** → Type 0 (No conditional processing) **X'01'** → Type 1 (Annotation)	1	M
	3	**X'01'–X'FE'** → Level. It must be in the range of 1 to 254.	1	O
		Resource Usage Attribute Triplet (See "Resource Usage Attribute Triplet X'47'" on page 363)	1	O
	0–1	**X'0347'** → Triplet length and identifier	1	M
	2	Frequency of use. It must be one of the following: **X'00'** → Low **X'FF'** → High	1	M

Index Element

IEL X'D3B2A7' Index Element (See "Index Element (IEL)" on page 176)				
0–*n*		The following triplets, in any order:		
		Fully Qualified Name Triplet (See "Fully Qualified Name Triplet X'02'" on page 320)	1	M
	0–1	**X'*nn*02'** → Triplet length and identifier	1	M
	2–3	**X'CA00'** → FQN type and format, Index Element Name	1	M
	4–*n*	Name of this IEL. It may be 1 to 250 bytes in length.	1	M
		Object Byte Offset Triplet (See "Object Byte Offset Triplet X'2D'" on page 353)	1	M
	0–1	**X'062D'** → Triplet length and identifier	1	M
	2–5	Direct byte offset. It must be one of the following: **X'00000000'–X'7FFFFFFF'** → Byte offset from beginning of document containing indexed element **X'FFFFFFFF'** → Indexed element is outside the document	1	M

Invoke Medium Map

IMM X'D3ABCC' Invoke Medium Map (See “Invoke Medium Map (IMM)” on page 178)			
0–7	External name of the medium map to be invoked (8 characters)	1	M

Map Bar Code Object

MBC X'D3ABEB' Map Bar Code Object (See “Map Bar Code Object (MBC)” on page 206)				
0–1	**X'0005'** → Length of this repeating group is 5 bytes		1	M
2–4	The following triplet:			
	Mapping Option Triplet (See “Mapping Option Triplet X'04'” on page 332)		1	M
	0–1	**X'0304'** → Triplet length and identifier	1	M
	2	**X'00'** → Output option (position)	1	M

Note: If this structured field is not specified, the architected default is *position*.

Map Coded Font, Format 2

MCF X'D3AB8A' Map Coded Font (See “Map Coded Font (MCF) Format 2” on page 213)				
0–1	**X’00*nn*’** → Length of this repeating group		254	M
2–*n*	The following triplets, in any order:			
	Fully Qualified Name Triplet (See “Fully Qualified Name Triplet X'02'” on page 320) **Note:** See “MCF Font Names” on page 456 for details.		2	M
	0–1	**X'0C02'** → Triplet length and identifier	1	M
	2	The FQN type. It must be one of the following: **X'84'** → Coded Font Reference **X'85'** → Code Page Reference **X'86'** → Font Character Set Reference	1	M
	3	**X'00'** → FQN format	1	M
	4–11	External name of the coded font, code page, or font character set.	1	M
	Fully Qualified Name Triplet (See “Fully Qualified Name Triplet X'02'” on page 320)		1	O
	0–1	**X’*nn*02’** → Triplet length and identifier	1	M
	2–3	**X'0800'** → FQN type and format, Font Typeface Name	1	M
	4–*n*	External name of the font typeface. It may be 1 to 32 bytes in length.	1	M
	Font Descriptor Specification Triplet (See “Font Descriptor Specification Triplet X'1F'” on page 341)		1	O
	0–1	**X'141F'** → Triplet length and identifier	1	M
	2	**X'01'–X'09'** → Font Weight Class. It must be in the range of 1 to 9.	1	M
	3	**X'01'–X'09'** → Font Width Class. It must be in the range of 1 to 9.	1	M
	4–5	**X'0000'–X'7FFF'** → Font Height. It must be in the range of 0 to 32767 1440ths of an inch.	1	M
	6–7	**X'0000'–X'7FFF'** → Font Width. It must be in the range of 0 to 32767 1440ths of an inch.	1	M

MCF X'D3AB8A' Map Coded Font (See “Map Coded Font (MCF) Format 2” on page 213)			
8	Font Descriptor Flags, as follows: **Bit** **Description** **0** Italics **1** Underscored **2** Reserved, must be B'0' **3** Hollow **4** Overstruck **5** Proportional **6** Kerned characters (pairwise) **7** Reserved, must be B'0'	1	M
9–19	Reserved	1	M
	Font Coded Graphic Character Set Global Identifier Triplet (See “Font Coded Graphic Character Set Global Identifier Triplet X'20'” on page 345)	1	O
0–1	**X'0620'** → Triplet length and identifier	1	M
2–5	The GCSGID and CPGID for the font.	1	M
	Resource Local Identifier Triplet (See “Resource Local Identifier Triplet X'24'” on page 350)	1	M
0–1	**X'0424'** → Triplet length and identifier	1	M
2	**X'05'** → Resource type, coded font	1	M
3	Resource Local Identifier. It must be one of the following: **X'01'–X'7F'** → It must be in the range of 1 to 127 when used for mapping a font. **X'FE'** → It must be 254 when used for resource management purposes in the AEG.	1	M
	Resource Section Number Triplet (See “Resource Section Number Triplet X'25'” on page 351)	1	O
0–1	**X'0325'** → Triplet length and identifier	1	M
2	Resource Section Number. It must be one of the following: **X'00'** → It must be 0 when referencing an EBCDIC Presentation single-byte coded font (encoding scheme ID X’61*xx*’) or all sections of an EBCDIC Presentation double-byte coded font (encoding scheme ID X’62*xx*’). **X'41'–X'FE'** → It must be in the range of 65 to 254 when referencing a specific section of an EBCDIC Presentation double-byte coded font (encoding scheme ID X’62*xx*’).	1	M
	Character Rotation Triplet (See “Character Rotation Triplet X'26'” on page 352)	1	O
0–1	**X'0426'** → Triplet length and identifier	1	M
2–3	Character Rotation. It must be one of the following: **X'0000'** → 0-degree character rotation **X'2D00'** → 90-degree character rotation **X'5A00'** → 180-degree character rotation **X'8700'** → 270-degree character rotation	1	M

MCF Font Names: The MCF must have one of the following:

- A type X’84’ (Coded Font Reference) Fully Qualified Name (X’02’) triplet. To support existing products, the coded font name must be specified as a global resource identifier (GRID). For a definition of the GRID, see “Global Resource Identifier (GRID) Definition” on page 329.
- Both a type X’85’ (Code Page Name Reference) and a type X’86’ (Font Character Set Name Reference) Fully Qualified Name (X’02’) triplet. To support existing products, the names of the code page and font character set must be eight characters in length and must match the external names of these objects in their respective resource libraries.

Map Graphics Object

MGO X'D3ABBB' Map Graphics Object (See “Map Graphics Object (MGO)” on page 245)				
0–1		**X'0005'** → Length of this repeating group is 5 bytes	1	M
2–4		The following triplet:		
		Mapping Option Triplet (See “Mapping Option Triplet X'04'” on page 332)	1	M
	0–1	**X'0304'** → Triplet length and identifier	1	M
	2	Output Option. It must be one of the following: **X'10'** → Position and trim **X'20'** → Scale to fit **X'30'** → Center and trim	1	M

Note: If this structured field is not specified, the architected default is *scale to fit*.

Map Image Object

MIO X'D3ABFB' Map Image Object (See “Map Image Object (MIO)” on page 246)				
0–1		**X'0005'** → Length of this repeating group is 5 bytes	1	M
2–4		The following triplet:		
		Mapping Option Triplet (See “Mapping Option Triplet X'04'” on page 332)	1	M
	0–1	**X'0304'** → Triplet length and identifier	1	M
	2	Output Option. It must be one of the following: **X'10'** → Position and trim **X'20'** → Scale to fit **X'30'** → Center and trim	1	M

Note: If this structured field is not specified, the architected default is *scale to fit*.

Map Page Overlay

MPO X'D3ABD8' Map Page Overlay (See “Map Page Overlay (MPO)” on page 265)				
0–1		**X’*nnnn*’** → Length of this repeating group	127	M
2–17		The following triplet:		
		Fully Qualified Name Triplet (See “Fully Qualified Name Triplet X'02'” on page 320)	1	M
	0–1	**X'0C02'** → Triplet length and identifier	1	M
	2–3	**X'8400'** → FQN type and format, reference to overlay	1	M
	4–11	External name of the overlay.	1	M
		Resource Local Identifier Triplet (See “Resource Local Identifier Triplet X'24'” on page 350)	1	M

MPO X'D3ABD8' Map Page Overlay (See “Map Page Overlay (MPO)” on page 265)			
0–1	**X'0424'** → Triplet length and identifier	1	M
2	**X'02'** → Resource type, page overlay	1	M
3	**X'01'–X'7F'** → Resource Local Identifier. It must be in the range of 1 to 127.	1	M
	Page Overlay Conditional Processing Triplet (See “Page Overlay Conditional Processing Triplet X'46'” on page 361)	*n*	O
0–1	**X’*nn*46’** → Triplet length and identifier	1	M
2	Page Overlay Type. It must be one of the following: **X'00'** → Type 0 (No conditional processing) **X'01'** → Type 1 (Annotation)	1	M
3	**X'01'–X'FE'** → It must be in the range of 1 to 254.	1	O
	Resource Usage Attribute Triplet (See “Resource Usage Attribute Triplet X'47'” on page 363)	1	O
0–1	**X'0347'** → Triplet length and identifier	1	M
2	Frequency of use. It must be one of the following: **X'00'** → Low **X'FF'** → High	1	M

No Operation

NOP X'D3EEEE' No Operation (See “No Operation (NOP)” on page 269)	
0–*n*	Up to 32759 bytes of data.

Object Area Descriptor

OBD X'D3A66B' Object Area Descriptor (See “Object Area Descriptor (OBD)” on page 270)			
0–*n*	The following triplets, in any order:		
	Descriptor Position Triplet (See “Descriptor Position Triplet X'43'” on page 355)	1	M
0–1	**X'0343'** → Triplet length and identifier	1	M
2	**X'01'–X'7F'** → Descriptor position ID. It must be in the range of 1 to 127.	1	M
	Measurement Units Triplet (See “Measurement Units Triplet X'4B'” on page 364)	1	M
0–1	**X'084B'** → Triplet length and identifier	1	M
2–3	**X'0000'** → Object area measurement units base for X and Y	1	M
4–5	Object area measurement units value for X. It must be one of the following: **X'0960'** → 2400 units per unit base (240 units per inch) **X'3840'** → 14400 units per unit base (1440 units per inch)	1	M
6–7	Object area measurement units value for Y. It must be identical to bytes 4–5.	1	M
	Object Area Size Triplet (See “Object Area Size Triplet X'4C'” on page 365).	1	M
0–1	**X'094C'** → Triplet length and identifier	1	M
2	**X'02'** → Type, actual object area size	1	M

OBD X'D3A66B' Object Area Descriptor (See "Object Area Descriptor (OBD)" on page 270)			
3–5	Object area size in the X direction. It must be one of the following: **X'000001'–X'001555'** → In the range of 1 to 5461 when using 240 units per inch for the object area X measurement units **X'000001'–X'007FFF'** → In the range of 1 to 32767 when using 1440 units per inch for the object area X measurement units	1	M
6–8	Object area size in the Y direction. It must be one of the following: **X'000001'–X'001555'** → In the range of 1 to 5461 when using 240 units per inch for the object area Y measurement units **X'000001'–X'007FFF'** → In the range of 1 to 32767 when using 1440 units per inch for the object area Y measurement units	1	M
Presentation Space Reset Mixing Triplet (See "Presentation Space Reset Mixing Triplet X'70'" on page 390)		1	O
0–1	**X'0370'** → Triplet length and identifier	1	M
2	Mixing Flags, as follows: **Bit** **Description** **0** Reset **0** Do not reset to color of medium **1** Reset to color of medium **1–7** Reserved, must be zero	1	M
Note: This triplet is *only* permitted on Object Area Descriptor structured fields that are contained within a page overlay. The page overlay itself *must* be carried within the inline page resource group. If specified on any other Object Area Descriptor structured field, a X'01' exception condition exists.			

Note: If the presentation text Object Area Descriptor structured field appears in the AEG, the measurement units and extents specified on it must match those specified on the Page Descriptor structured field, or a X'01' exception condition exists. If the presentation text Object Area Descriptor structured field is omitted, the architected default is to use the measurement units and extents specified on the Page Descriptor structured field for the presentation text object area. Thus, the presentation text object area and the page are always the same size and points within their respective coordinate systems are always coincident.

Object Area Position

OBP X'D3AC6B' Object Area Position (See "Object Area Position (OBP)" on page 272)			
0	**X'01'–X'7F'** → Object Area Position ID. It must be in the range of 1 to 127.	1	M
1	**X'17'** → Length of this repeating group is 23 bytes	1	M
2–4	Object area origin for X. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page or overlay X measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page or overlay X measurement units	1	M

OBP X'D3AC6B' Object Area Position (See "Object Area Position (OBP)" on page 272)			
5–7	Object area origin for Y. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page or overlay Y measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page or overlay Y measurement units	1	M
8–11	Object Area orientation, X and Y coordinates. It must be one of the following: **X'0000 2D00'** → X=0 degrees, Y=90 degrees **X'2D00 5A00'** → X=90 degrees, Y=180 degrees **X'5A00 8700'** → X=180 degrees, Y=270 degrees **X'8700 0000'** → X=270 degrees, Y=0 degrees	1	M
12	**X'00'** → Reserved, must be binary zero	1	M
13–15	Object content origin for X. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page or overlay X measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page or overlay X measurement units	1	M
16–18	Object content origin for Y. It must be one of the following: **X'000000'–X'001555'** → In the range of 0 to 5461 when using 240 units per inch for the page or overlay Y measurement units **X'000000'–X'007FFF'** → In the range of 0 to 32767 when using 1440 units per inch for the page or overlay Y measurement units	1	M
19–20	**X'0000'** → Object content orientation, X (0 degrees)	1	M
21–22	**X'2D00'** → Object content orientation, Y (90 degrees)	1	M
23	Referenced coordinate system. It must be one of the following: **X'00'** → Current coordinate system **X'01'** → Page or overlay coordinate system	1	M

Notes:

1. If the presentation text Object Area Position structured field appears in the AEG, the X and Y values for the object area origin and the object content origin must be set to zero, or a X'01' exception condition exists. If the presentation text Object Area Position structured field is omitted, the architected default is to set the X and Y values for the object area origin and the object content origin to zero. For presentation text, the data object presentation space origin is positioned coincident with the object content origin. Thus, the presentation text object presentation space, the presentation text object area, and the page always have the same origin.
2. If the presentation text OBP appears in the AEG, the object area orientation must be set to X'0000 2D00' (0°,90°). If it is omitted, the architected default is to set the object area orientation to X'0000 2D00' (0°,90°).
3. For this interchange set, the values X'00' and X'01' in byte 23 specify the same function since positioning with respect to a page segment offset is not part of the interchange set definition. That is, both values specify that the object area is to be positioned with respect to the including page or overlay coordinate system.

Page Descriptor

PGD X'D3A6AF' Page Descriptor (See “Page Descriptor (PGD)” on page 279)			
0–1	**X'0000'** → Page measurement units base for X and Y	1	M
2–3	Page measurement units value for X. It must be one of the following: **X'0960'** → 2400 units per unit base (240 units per inch) **X'3840'** → 14400 units per unit base (1440 units per inch)	1	M
4–5	Page measurement units value for Y. It must be identical to bytes 2–3.	1	M
6–8	Page size in the X direction. It must be one of the following: **X'000001'–X'001555'** In the range of 1 to 5461 when using 240 units per inch for the page X measurement units **X'000001'–X'007FFF'** In the range of 1 to 32767 when using 1440 units per inch for the page X measurement units	1	M
9–11	Page size in the Y-direction. It must be one of the following: **X'000001'–X'001555'** In the range of 1 to 5461 when using 240 units per inch for the page Y measurement units **X'000001'–X'007FFF'** In the range of 1 to 32767 when using 1440 units per inch for the page Y measurement units	1	M
12–14	**X'000000'** → Reserved, must be binary zero	1	M
15–17	The following triplet:		
	Presentation Space Reset Mixing Triplet (See “Presentation Space Reset Mixing Triplet X'70'” on page 390)	1	O
	0–1 **X'0370'** → Triplet length and identifier	1	M
	2 Mixing Flags, as follows: **Bit** **Description** **0** Reset **0** Do not reset to color of medium **1** Reset to color of medium **1–7**	1	M
Note: This triplet is permitted *only* on Page Descriptor structured fields that are contained within a page overlay. The page overlay itself *must* be carried within the inline page resource group. If specified on any other Page Descriptor structured field, a X’01’ exception condition exists.			

Application Note: The IS/1 and IS/2 interchange set definitions limit the page size to 22.75 inches in the X and Y directions. To specify a larger page size, 240 units per inch should be specified in the PGD for the page measurement units. Using a range of 1 to 32767, this will allow a maximum page size in the X and Y directions of 136.5 inches, is supported by all IPDS printers, and keeps the complete page presentation space within the range of two-byte addressing parameters in the IPDS and PTOCA architectures.

Presentation Text Data

PTX X'D3EE9B' Presentation Text Data (See “Presentation Text Data (PTX)” on page 309)	
0–*n*	Up to 8192 bytes of presentation text data as defined by PTOCA PT1

Presentation Text Data Descriptor, Format 2

PTD X'D3B19B' Presentation Text Data Descriptor (See “Presentation Text Data Descriptor (PTD) Format 2” on page 308)	
0–*n*	Presentation text descriptor data as defined by PTOCA

Note: When the PTD is included in the AEG for a page, some AFP print servers require that the measurement units in the PTD match the measurement units in the Page Descriptor (PGD). It is therefore strongly recommended that whenever the PTD is included in the AEG, the same measurement units are specified in both the PTD and PGD.

MO:DCA Resource Interchange Set

This section defines the MO:DCA Resource Interchange Set (MO:DCA-L) used for resource documents.

For information on the level of function required for the OCAs included in this interchange set, refer to the MO:DCA environment appendix in the following IBM documents:

GOCA	*Graphics Object Content Architecture Reference*, SC31-6804
IOCA	*Image Object Content Architecture Reference*, SC31-6805

Data Stream Syntax Structure

The groupings of MO:DCA structured fields that follow identify those structured fields which appear within each begin-end structured field pair or state. This section specifies the structured fields allowed within a MO:DCA-L data stream and shows both the MO:DCA state hierarchy and the validity of structured fields within each state.

If a structured field that is not identified as being part of this interchange set appears anywhere within the data stream, a X'40' exception condition exists If a structured field appears within any state where it is not permitted, or if it appears out of the stated order or more than the permitted number of times, a X'20' exception condition exists. If a structured field that is identified as required does not appear within a specific state, a X'08' exception condition exists.

The conventions used in these structured field groupings are:

( ) The structured field acronym and identifier are shown in parentheses. The presence of dots or periods in the identifier indicates that the item is not a structured field, but instead is a structure, for example a page. The structure is composed of an assortment of structured fields, and is defined separately.

[ ] Brackets indicate optional structured fields. When a structured field is shown without brackets, it *must* appear between the begin and end structured fields.

+ Plus signs indicate structured fields may appear in any order relative to those that precede or succeed it except when the preceding or succeeding structured field does not have a plus (+) sign. Then the order is as listed.

(S) The enclosed (S) indicates that the structured field may be repeated. When present on a required structured field, at least one occurrence of the structured field is required, but multiple instances of it may occur.

F2 An F2 indicates that the structured field is a format two structured field. See “Structured Field Formats” on page 25 for further details.

Notes:

1. The Begin Document and End Document structured fields are required in a MO:DCA data stream.
2. The No Operation structured field may appear within any begin-end domain and thus is not listed in the structured field groupings.
3. The architecture that owns and controls the content of each of the data and resource objects carried in a MO:DCA data stream is identified in the following structured field groupings. Please refer to the referenced documentation for further details.

4. The Flag byte (byte 5) in the Structured Field Introducer (SFI) must be set to X'00'. MO:DCA-L does not support SFI extension, structured field segmentation, or structured field padding.

Document

```
Begin Document  (BDT, D3A8A8)
        (      D3..C6)    Resource Group
End Document  (EDT, D3A9A8)
```

Figure 96. MO:DCA-L: Document Structure

Document Resource Group

```
Begin Resource Group  (BRG, D3A8C6)
  +     (      D3..77)    Color Attribute Table
  +  [  (      D3..FB)    Image Object                                  (S)  ]
        (      D3..BB)    Graphics Object
End Document  (ERG, D3A9C6)
```

Figure 97. MO:DCA-L: Document Resource Group Structure

Color Attribute Table

```
Begin Color Attribute Table  (BCA, D3A877)
        (CAT,  D3B077)    Color Attribute Table
End Color Attribute Table  (ECA, D3A977)
```

Figure 98. MO:DCA-L: Color Attribute Table Structure

Image Object (IOCA FS20)

```
Begin Image Object  (BIM, D3A8FB)
     [  (      D3..C6)    Resource Group                                       ]
     [  (      D3..C7)    Object Environment Group                             ]
        (IDD,  D3A6FB)    Image Data Descriptor
        (IPD,  D3EEFB)    Image Picture Data                                    [9]
        (IPD,  D3EEFB)    Image Picture Data                            (S)     [9]
End Image Object  (EIM, D3A9FB)
```

Figure 99. MO:DCA-L: Image Object Structure

Note: Refer to the *Image Object Content Architecture Reference* for a full description of the IOCA FS20 content, syntax, and semantics for MO:DCA-L.

[9] At least two IPD structured fields are *mandatory*. The first contains only the IPD parameters, while the second and any subsequent IPD structured fields contain the image data.

Image Resource Group

```
Begin Resource Group  (BRG, D3A8C6)
        (      D3..77)    Color Attribute Table
End Resource Group  (ERG, D3A9C6)
```

Figure 100. MO:DCA-L: Image Resource Group Structure

Object Environment Group (OEG) for Image Object

```
Begin Object Environment Group  (BOG, D3A8C7)
        (MCA,  D3AB77)    Map Color Attribute Table
End Object Environment Group  (EOG, D3A9C7)
```

Figure 101. MO:DCA-L: Object Environment Group for Image Object Structure

Graphics Object (GOCA DR/3V1)

```
Begin Graphics Object  (BGR, D3A8BB)
        (      D3..C7)    Object Environment Group
        (GDD,  D3A6BB)    Graphics Data Descriptor
        (GAD,  D3EEBB)    Graphics Data                           (S)
End Graphics Object  (EGR, D3A9BB)
```

Figure 102. MO:DCA-L: Graphics Object Structure

Note: Refer to the *Graphics Object Content Architecture Reference* for a full description of the GOCA DR/3V1 content, syntax, and semantics for MO:DCA-L.

Object Environment Group (OEG) for Graphics Object

```
Begin Object Environment Group  (BOG, D3A8C7)
 +      (MCA,  D3AB77)    Map Color Attribute Table
 +      (MCF,  D3AB8A)    Map Coded Font                       F2  (S)    10
 +   [  (MDR,  D3ABC3)    Map Data Resource                        (S)  ]
End Object Environment Group  (EOG, D3A9C7)
```

Figure 103. MO:DCA-L: Object Environment Group for Graphics Object Structure

Permitted Structured Fields

This section describes the parameters and ranges of values supported for each of the structured fields contained in this interchange set.

The structured fields are listed alphabetically and described using tables. The table heading for each structured field contains the structured field’s acronym, its three-byte hexadecimal identifier, and its full name. Also included is the page number in the document where a detailed description of the structured field can be found.

[10] At least one MCF structured field is *mandatory* for the default font. A separate MCF structured field is required for each specific coded font desired.

Structured Field Parameters

In general, the structured field tables contain the following information for each parameter:

1. The offset from the beginning of the data portion of the structured field or from the beginning of the triplet.
2. Values and description:
   - When a specific parameter value is required, the specific value or the range of acceptable values is specified, followed by → and an explanation or description of the parameter.
   - When no specific value is required, or when a choice of values is required, the parameter name or a description of the parameter is given. If a choice of values is required, the choices are identified in the table.
3. For those parameters defined and owned by the MO:DCA architecture, occurrence is specified either as a lowercase *n* indicating that the occurrence is unlimited by the interchange set, or as a number representing the maximum number of times the parameter may appear within the containing structured field, repeating group, or triplet.
4. For those parameters defined and owned by the MO:DCA architecture, optionality is specified as:

   O Optional. The parameter may or may not appear.

   M Mandatory. The parameter must always appear.

   C Conditional. The parameter is mandatory under certain conditions, but is optional or not allowed under other conditions.

Unless a specific order is required, self-identifying parameters are listed in alphanumeric sequence by identifier and include the page number in the document where a detailed description of the parameter is located.

In general, no exception conditions are identified within the interchange set definition for the structured fields or their parameters. The page numbers provided for each structured field and each triplet provide the source for determining what exception conditions may be anticipated. However, the following general rules apply:

- For those structured fields where a parameter order is stated, if a parameter appears outside that stated order, a X’01’ exception condition exists.
- If a parameter value appears that is outside the range specified for that parameter, a X’02’ exception condition exists.
- If a parameter that is identified as mandatory does not appear on a specific structured field, a X’04’ exception condition exists.
- Unless otherwise stated, if any unrecognized parameter or triplet appears on any structured field, a X’10’ exception condition exists.

Notes:

1. Any triplet encountered on any of the *Begin* structured fields listed below that is not explicitly defined as being valid for that structured field should be ignored and should not cause an exception condition.
2. If specified, the name contained in the name parameter on an *End* structured field must match that specified in the name parameter on its matching *Begin* structured field, or a X'01' exception condition exists.

Begin Color Attribute Table

BCA X'D3A877' Begin Color Attribute Table (See "Begin Color Attribute Table (BCA)" on page 107)			
0–7	Color Attribute Table name (8 characters)	1	M

Begin Document

BDT X'D3A8A8' Begin Document (See "Begin Document (BDT)" on page 114)			
0–7	Document name (8 characters)	1	M
8–9	**X'0000'** → Reserved, must be binary zero	1	M
10–*n*	The following triplets, in any order:		
	Coded Graphic Character Set Global Identifier Triplet (See "Coded Graphic Character Set Global Identifier Triplet X'01'" on page 317)	1	M
	0–1 **X'0601'** → Triplet length and identifier	1	M
	2–5 **X'03AA0352'** → Character set and code page identification (character set 938, code page 850)	1	M
	MO:DCA Interchange Set Triplet (See "MO:DCA Interchange Set Triplet X'18'" on page 339)	1	M
	0–1 **X'0518'** → Triplet length and identifier	1	M
	2 **X'03'** → Interchange set type, resource	1	M
	3–4 **X'0C00'** → Interchange set identifier (MO:DCA-L)	1	M
	Comment Triplet (See "Comment Triplet X'65'" on page 385)	1	O
	0–1 **X'*nn*65'** → Triplet length and identifier	1	M
	2–*n* Comment used for metafile description of up to 252 bytes	1	M

Begin Graphics Object

BGR X'D3A8BB' Begin Graphics Object (See "Begin Graphics Object (BGR)" on page 118)			
0–7	Graphics Object name (8 characters)	1	M

Begin Image Object

BIM X'D3A8FB' Begin Image Object (See "Begin Image Object (BIM)" on page 120)			
0–7	Image Object name (8 characters)	1	M

Begin Object Environment Group

BOG X'D3A8C7' Begin Object Environment Group (See "Begin Object Environment Group (BOG)" on page 133)			
0–7	Object Environment Group name (8 characters)	1	M

Begin Resource Group

BRG X'D3A8C6' Begin Resource Group (See "Begin Resource Group (BRG)" on page 141)			
0–7	Resource Group name (8 characters)	1	M

Color Attribute Table

CAT X'D3B077' Color Attribute Table (See “Color Attribute Table (CAT)” on page 149)			
0–*n*	Color Attribute Table data as defined in Appendix A, “Color Resources,” on page 473		

End Color Attribute Table

ECA X'D3A977' End Color Attribute Table (See “End Color Attribute Table (ECA)” on page 153)			
0–7	Color Attribute Table name (8 characters)	1	M

End Document

EDT X'D3A9A8' End Document (See “End Document (EDT)” on page 157)			
0–7	Document name (8 characters)	1	M

End Graphics Object

EGR X'D3A9BB' End Graphics Object (See “End Graphics Object (EGR)” on page 159)			
0–7	Graphics Object name (8 characters)	1	M

End Image Object

EIM X'D3A9FB' End Image Object (See “End Image Object (EIM)” on page 160)			
0–7	Image Object name (8 characters)	1	M

End Object Environment Group

EOG X'D3A9C7' End Object Environment Group (See “End Object Environment Group (EOG)” on page 165)			
0–7	Object Environment Group name (8 characters)	1	M

End Resource Group

ERG X'D3A9C6' End Resource Group (See “End Resource Group (ERG)” on page 170)			
0–7	Resource Group name (8 characters)	1	M

Graphics Data

GAD X'D3EEBB' Graphics Data (See “Graphics Data (GAD)” on page 173)			
0–*n*	Up to 32759 bytes of graphics data as defined by GOCA DR/3V1		

Graphics Data Descriptor

GDD X'D3A6BB' Graphics Data Descriptor (See “Graphics Data Descriptor (GDD)” on page 174)			
0–*n*	Graphics descriptor data as defined by GOCA		

Image Data Descriptor

IDD X'D3A6FB' Image Data Descriptor (See "Image Data Descriptor (IDD)" on page 175)	
0–*n*	Image descriptor data as defined by IOCA FS20

Image Picture Data

IPD X'D3EEFB' Image Picture Data (See "Image Picture Data (IPD)" on page 190)	
0–*n*	Up to 32759 bytes of image segment data as defined by IOCA FS20

Note: At least two IPD structured fields are *mandatory*. The first contains only the IPD parameters while the second (and any subsequent ones) contain the image data.

Map Coded Font, Format 2

MCF X'D3AB8A' Map Coded Font (See "Map Coded Font (MCF) Format 2" on page 213)				
0–1	**X'00*nn*'** → Length of this repeating group		1	M
Note: Only one repeating group is permitted on this structured field.				
2–*n*	The following triplets, *in the order specified*:			
	Fully Qualified Name Triplet (See "Fully Qualified Name Triplet X'02'" on page 320)		1	M
	0–1	**X'0C02'** → Triplet length and identifier	1	M
	2–3	**X'8400'** → FQN type and format, reference to coded font	1	M
	4–11	External name of the coded font	1	M
Note: The coded font name is the eight-character name supplied by the CPI call. The default name is indicated by the use of a X'FF' as the first character of the name. The default name is always mapped to LID X'00'.				
	Resource Local Identifier Triplet (See "Resource Local Identifier Triplet X'24'" on page 350)		1	M
	0–1	**X'0424'** → Triplet length and identifier	1	M
	2	**X'05'** → Resource type, coded font	1	M
	3	**X'01'–X'FE'** → Resource Local Identifier. It must be in the range of 1 to 254.	1	M
	Font Descriptor Specification Triplet (See "Font Descriptor Specification Triplet X'1F'" on page 341)		1	C
Note: This triplet is *mandatory* on all Map Coded Font structured fields other than the one for the default font. It is *not permitted* on the Map Coded Font structured field that specifies the default font.				
	0–1	**X'141F'** → Triplet length and identifier	1	M
	2	Font Weight Class. It must be one of the following: **X'05'** → Medium (normal) **X'07'** → Bold	1	M
	3	**X'05'** → Font Width Class of medium (normal)	1	M
	4–5	**X'0001'–X'7FFF'** → Font Height. It must be in the range of 1 to 32767 in world coordinate units.	1	M
	6–7	**X'0001'–X'7FFF'** → Font Width. It must be in the range of 1 to 32767 in world coordinate units.	1	M

MCF X'D3AB8A' Map Coded Font (See "Map Coded Font (MCF) Format 2" on page 213)			
8	Font Descriptor Flags, as follows: **Bits** **Description** **0** Italics **1** Underscored **2** Reserved, must be zero **3** Hollow **4** Overstruck **5** Proportional **6** Kerned characters (pairwise) **7** Reserved, must be zero	1	M
9–18	Reserved	1	M
19	Font Flags, as follows: **Bits** **Description** **0** Reserved, must be set to zero **1** Font type **0** Bitmapped font **1** Outline (vector) font **2** Transform font **0** Font will not be transformed **1** Font may be transformed (scaled, rotated, sheared) **3–7** Reserved, must be zero	1	M
	Font Coded Graphic Character Set Global Identifier Triplet (See "Font Coded Graphic Character Set Global Identifier Triplet X'20'" on page 345)	1	M
0–1	**X'0620'** → Triplet length and identifier	1	M
2–5	The GCSGID and CPGID for the font.	1	M
	Fully Qualified Name Triplet (See "Fully Qualified Name Triplet X'02'" on page 320)	1	C
Note: This triplet is *mandatory* on all Map Coded Font structured fields other than the one for the default font. It is *not permitted* on the Map Coded Font structured field that specifies the default font.			
0–1	**X'2402'** → Triplet length and identifier	1	M
2–3	**X'0800'** → FQN type and format, typeface identifier	1	M
4–35	External name of the font typeface. It must be 32 bytes in length.	1	M

Note: At least one Map Coded Font structured field is *mandatory* for the default font. The default font is indicated by the use of a X'FF' as the first byte of the coded font name in the Begin Resource Object Reference Fully Qualified Name triplet. The default font is always mapped to a X'00' local identifier. The Font Descriptor Specification triplet and the Fully Qualified Name triplet for the font typeface are not permitted on the Map Coded Font structured field for the default font.

Map Color Attribute Table

MCA X'D3AB77' Map Color Attribute Table (See "Map Color Attribute Table (MCA)" on page 207)			
0–1	**X'00*nn*'** → Length of this repeating group is either 14 or 18 bytes	254	M
2–*n*	The following triplets, in any order:		
	Fully Qualified Name Triplet (See "Fully Qualified Name Triplet X'02'" on page 320)	1	M
0–1	**X'0C02'** → Triplet length and identifier	1	M

MCA X'D3AB77' Map Color Attribute Table (See "Map Color Attribute Table (MCA)" on page 207)				
	2–3	**X'8400'** → FQN type and format, reference to color table	1	M
	4–11	External name of the color table	1	M
	Resource Local Identifier Triplet (See "Resource Local Identifier Triplet X'24'" on page 350)		1	C
Note: This triplet is *mandatory* for image, *not permitted* for graphics.				
	0–1	**X'0424'** → Triplet length and identifier	1	M
	2	**X'07'** → Color table resource type	1	M
	3	**X'01'–X'FE'** → Resource Local Identifier. It must be in the range of 1 to 254.	1	M

Map Data Resource

MDR X'D3ABC3' Map Data Resource (See "Map Data Resource (MDR)" on page 223)				
0–1	**X'0015'** → Length of this repeating group is 21 bytes		*n*	M
2–*n*	The following triplets, in any order:			
	Fully Qualified Name Triplet (See "Fully Qualified Name Triplet X'02'" on page 320)		1	M
	0–1	**X'0C02'** → Triplet length and identifier	1	M
	2–3	**X'8400'** → FQN type and format, reference to image object	1	M
	4–11	External name of the image object	1	M
	Extended Resource Local Identifier Triplet (See "Extended Resource Local Identifier Triplet X'22'" on page 348)		1	M
	0–1	**X'0722'** → Triplet length and identifier	1	M
	2	**X'10'** → Resource type, image	1	M
	3–6	**X'00000000'–X'FFFFFFFF'** → Resource Local Identifier (used as bitmap handle)	1	M

No Operation

NOP X'D3EEEE' No Operation (See "No Operation (NOP)" on page 269)	
0–*n*	Up to 32759 bytes of data

Appendix A. Color Resources

This appendix describes color resources that may be used in MO:DCA environments. For a discussion of font resources, see *Font Object Content Architecture Reference*.

Standard OCA Color Value Table

The following table defines the valid color values used to specify named colors in PTOCA, IOCA, GOCA, BCOCA, and IM Image objects. The table also specifies the RGB values for each named color, assuming that each component is specified with 8 bits and that the component intensity range 0 to 1 is mapped to the binary value range 0 to 255. Although all values in this table are syntactically valid in these objects, some objects support only a subset of the colors. For a definition of the supported colors, see the Object Content Architecture references for the individual objects. Note that this table defines the complete set of colors supported by the GOCA Set Extended Color drawing order. The Color Specification (X'4E') triplet also supports these colors for the Standard OCA color space; see "Color Specification Triplet X'4E'" on page 367.

Table 25. Color Values

Value	Color	Red (R)	Green (G)	Blue (B)
X'0000' or X'FF00'	Presentation-process default; see Note 1 on page 474			
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 2 on page 474	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'	Presentation-process default; see Note 3 on page 474	—	—	—
X'FF08'	Color of medium	—	—	—

Table 25. Color Values (continued)

Value	Color	Red (R)	Green (G)	Blue (B)
All others	Reserved	—	—	—

Notes:

1. The presentation-process default specified by X'0000' and X'FF00' is resolved based on data type as follows:
   - For PTOCA text data, it is the presentation device default.
   - For bi-level IOCA Image data (FS10), it is the presentation device default.
   - For IM Image data, it is the presentation device default.
   - For GOCA graphics data, it is the drawing order default defined in the Graphics Data Descriptor (GDD) structured field.
   - For BCOCA bar code data, it is the presentation device default.
2. The color rendered on presentation devices that do not support white is device-dependent. For example, some printers simulate with color of medium, which results in white if white media is used.
3. The presentation-process default specified by X'FF07' is resolved as the presentation device default. This color value is also known in GOCA as neutral white for compatibility with display devices.
4. The value X'FFFF' is not defined in the Standard OCA Color Value Table but is used by some objects as a default indicator as follows:
   - For PTOCA text data, X'FFFF' may be specified in the Set Text Color (STC) control sequence to indicate that the PTOCA default hierarchy is used to generate the color value. Note that X'FFFF' is not supported in the Set Extended Text Color (SEC) control sequence.
   - For IM image data in MO:DCA environments, X'FFFF' may be specified to indicate use of a presentation process default color value. The value X'FFFF' is not valid for IM image in IPDS environments.
   - For bi-level IOCA image data (FS10), X'FFFF' may be specified to indicate use of a presentation process default color.
   - For BCOCA data, X'FFFF' may be specified to indicate use of a presentation device default color.

Converting Colors to Grayscale in MO:DCA-P Environments

Documents containing color specifications may be sent to bi-level devices such as black and white printers. If the presentation process decides, based on user fidelity requirements or on defaults, that the document is to be presented using grayscale substitution, the specified colors in the document should be simulated in a consistent and predictable manner by varying the intensity of the available color. On black and white printers, this means that colors are simulated with a grayscale where the intensity level of the output gray is determined by the lightness (L) of the color being simulated. A lightness of 0 is defined to be black and a lightness of 100 is defined to be white.

The following equations specify how the lightness (L) is derived from a color specified in one of the MO:DCA-supported color spaces.

CIELab Color Space

$$L = L$$

assuming

$$0 <= L <= 100$$

RGB Color Space

First the CIE luminance (Y) is generated:

$$Y = 0.212(R^{2.2}) + 0.701(G^{2.2}) + 0.087(B^{2.2})$$

assuming

$$0 <= R,G,B <= 1$$

Note: In this equation, R, G, B are the gamma-corrected (non-linear) components of the source color.

The lightness (L) is calculated from the CIE luminance (Y) using the following equation:

$$L = 116(Y^{1/3}) - 16 \quad \text{for } Y > 0.008856$$

$$L = 903.3Y \quad \text{for } Y <= 0.008856$$

assuming

$$0 <= Y <= 1$$

CMYK Color Space

First the CIE luminance (Y) is generated:

$$Y = 1 - \min(1,\ 0.212C + 0.701M + 0.087Y + K)$$

assuming

$$0 <= C,M,Y,K <= 1$$

where the function min(a,b) selects the smaller of (a,b).

The lightness (L) is calculated from the CIE luminance (Y) using the following equation:

$$L = 116(Y^{1/3}) - 16 \quad \text{for } Y > 0.008856$$
$$L = 903.3Y \quad \text{for } Y<= 0.008856$$

assuming

$$0 <= Y <= 1$$

Standard OCA Color Space (Named Colors)

Named colors are first converted to RGB values using the mapping defined in the Standard OCA Color Value Table; see "Standard OCA Color Value Table" on page 473. Once the named color is converted to an RGB value, the equations for calculating lightness (L) from RGB are used.

Note: The Standard OCA color space also supports a value for color of medium. This color is not simulated with a grayscale intensity.

Highlight Color Space

In the absence of a color mapping, each highlight color is simulated with black, and % coverage is applied.

The Color Mapping Table Resource

The Color Mapping Table (CMT) is used to map color values specified in a source color space to color values specified in a target color space. This allows colors specified in one or more source documents to be mapped to colors more suitable to the selected presentation device without requiring changes to the applications that generate the documents.

Color Mapping Table in MO:DCA-P Environments

The Color Mapping Table (CMT) is invoked when the print request to present one or more MO:DCA documents is issued. The CMT specified in the print request may be located in the resource group associated with the document; or it may be located in a resource library; or it may be the presentation process default CMT. The scope of the CMT in a MO:DCA presentation environment is the document or documents for which it is invoked. The invoked CMT remains active until another CMT is invoked. If no CMT is active, or if the reset table is active, no color mapping takes place.

The Color Mapping Table is a non-presentation resource object that is carried in a MO:DCA object container with the following structure:

Color Mapping Table Container

```
Begin Object Container  (BOC, D3A892)
     [  (OCD,  D3EE92)    Object Container Data                       (S)  ]
End Object Container  (EOC, D3A992)
```

Figure 104. Color Mapping Table Container

The table may be split on any byte boundary across any number of OCD structured fields. The mandatory Object Classification (X'10') triplet on the BOC structured field specifies the following parameter values:

ObjClass X'30' (set-up file)

StrucFlgs X'DC00' (data is carried within a container, does not include an OEG, and is carried in OCD structured fields)

RegObjId X'06072B120004010114'

Color Mapping Table in IPDS Environments

When a Color Mapping Table is sent to an IPDS printer in a non-presentation object container, it applies to all selected presentation data that is printed from that time on until the CMT is replaced by another CMT or by the reset table. The CMT is not applied to data in a resource object, such as an overlay or page segment, until that resource object is included on a logical page. This means that if the CMT changes between includes of an overlay, the overlay can be printed in different colors. However, this is not true for pages that are being processed and saved as resources in the presentation device. For that case, the CMT that is active when the page is saved is used to map colors in the page, not the CMT that is active when the saved page is included.

Note that if a color specified in the data stream is mapped with a CMT, the determination of color support is based on the CMT output color value, not on the CMT input (data stream) color value. Therefore, if an exception is detected because a color is not supported, the exception applies to the CMT output value, not to the data stream value.

Color Mapping Table Definition

The table definition consists of a base part, followed by zero or more repeating groups. The base part specifies the table to be a color mapping table or a reset color mapping table. If a reset color mapping table is specified, the repeating groups are optional and no color mappings occur when this table is invoked. If a color mapping table is specified, the base part is followed by two or more repeating groups. Each repeating group specifies a color space and a set of color values. Additionally, each repeating group specifies whether the color values are to be treated as sources, in which case it is a source repeating group, or as targets, in which case it is a target repeating group. Source repeating groups also specify the type of source data the color values should be associated with. The color mapping table must contain at least one source repeating group and one target repeating group. One or more source repeating groups can be associated with a single target repeating group by matching the repeating group IDs. While there may be multiple source repeating groups with the same repeating group ID, there cannot be more than one target repeating group with the same ID, and there must be a target repeating group for every source repeating group. If there is more than one target repeating group with the same ID, the first group is used and the rest are ignored. For example, if the table contains two source repeating groups, each with ID X'01', and if it contains a target repeating group with ID X'01', then the color values in both source repeating groups are mapped to the color values in the target repeating group for all object data specified by the source repeating groups. Repeating groups must be ordered such that all source repeating groups are specified first, sorted in ascending order of ID, followed by all target repeating groups sorted in ascending order of ID. Any repeating group that has a lower ID than a previous repeating group and is of the same type (source or target) is ignored, as is any source repeating group that follows a target repeating group.

Once a source repeating group has been matched with a target repeating group, the color values in the source repeating group are mapped sequentially to the color values in the target repeating group. That is, the first color value in the source repeating group is mapped to the first color value in the associated target repeating group, the second color value in the source repeating group is mapped to the second color value in the associated target repeating group, and so on. If there are more source color values than target color values, the source color values that do not have targets are mapped to presentation process default color values. If there are more target color values than source color values, the extra target color values are ignored. If the same source color value is mapped to more than one target color value, the first-specified target color value is used.

The presentation device uses the color mapping table to search the specified data objects for the source color values, and to replace the source color values with the target color values when rendering the data.

Color Mapping Table Syntax

Offset	Type	Name	Range	Meaning	M/O
0–1	UBIN	TBLlngth	6–65535	Table length	M
2–3	CODE	TBLid	1–65534	Table ID	M
4	CODE	TBLtpe	X'01', X'81'	Table type: **X'01'** Color mapping table **X'81'** Reset color mapping table	M
5				Reserved; must be zero	M
For a color mapping table (TBLtpe = X'01'), at least one source and one target repeating group in the following format:					
Source Repeating Group					
0–1	UBIN	RGLngth	30–(*n*+1)	Repeating group length	M
2	UBIN	RGId	1–127	Repeating group ID	M
3	CODE	RGTpe	X'01'	Repeating group type: **X'01'** Source color value repeating group **X'All others'** Reserved	M
4	CODE	ColSpce	X'06', X'40', X'50'	Color space: **X'06'** Highlight color space **X'40'** Standard OCA color space **X'50'** GOCA Pattern Fill space **X'All others'** Reserved	M
5–8				Reserved; must be zero	M
9	UBIN	ColSize1	X'08', X'10'	Number of bits in component 1; see color space definitions	M
10	UBIN	ColSize2	X'00', X'08'	Number of bits in component 2; see color space definitions	M
11	UBIN	ColSize3	X'00', X'08'	Number of bits in component 3; see color space definitions	M
12				Reserved; must be zero	M

Offset	Type	Name	Range	Meaning	M/O
13	CODE	ObjSel	X'6B', X'7B', X'9B', X'AF', X'BB', X'DF', X'EB', X'FB', X'FE', X'FF'	Source object type selector: **X'6B'** Object area **X'7B'** IM Image data **X'9B'** PTOCA data **X'AF'** Page presentation space **X'BB'** GOCA data **X'DF'** Overlay presentation space **X'EB'** BCOCA data **X'FB'** Non-tiled bi-level IOCA image data **X'FE'** All PTOCA, GOCA, BCOCA, non-tiled bi-level IOCA, and IM Image object data **X'FF'** All objects, object areas, and presentation spaces **All others** Reserved	M
14–29				Reserved; must be zero	M
30–*n*		Color Values		Sequential list of color values to be mapped	O
			Target Repeating Group		
0–1	UBIN	RGLngth	13–(*m*+1)	Repeating group length	M
2	UBIN	RGId	1–127	Repeating group ID	M
3	CODE	RGTpe	X'02'	Repeating group type: **X'02'** Target color value repeating group **All others** Reserved	M
4	CODE	ColSpce	X'01', X'04', X'06', X'08'	Color space: **X'01'** RGB **X'04'** CMYK **X'06'** Highlight color space **X'08'** CIELAB **All others** Reserved	M
5–8				Reserved; must be zero	M
9	UBIN	ColSize1	X'01'–X'08', X'10'	Number of bits in component 1; see color space definitions	M
10	UBIN	ColSize2	X'00'–X'08'	Number of bits in component 2; see color space definitions	M
11	UBIN	ColSize3	X'00'–X'08'	Number of bits in component 3; see color space definitions	M
12	UBIN	ColSize4	X'00'–X'08'	Number of bits in component 4; see color space definitions	M
13–*m*		Color Values		Sequential list of color values to be mapped	O

Color Mapping Table Semantics

TBLlngth Contains the length of the table, including this length field, in bytes.

TBLid Contains the identifier for the table.

TBLtpe Is a code that defines the type of table.

Value	Description
X'01'	Color Mapping Table. The table specifies mappings of source color values to target color values.
X'81'	Reset Color Mapping Table. The table resets all source-color-value to target-color-value mappings. The remainder of the table is ignored.

RGlngth Contains the length of the repeating group, including this length field, in bytes. The limits *n* and *m*, defined for source and target repeating groups respectively, are determined by the overall mapping table length limitation, which is 65535, and by the number of repeating groups and their size.

RGid Contains the identifier for the repeating group. This identifer is used to match source color value repeating groups with a target color value repeating group.

RGtpe Is a code that defines the type of repeating group.

Value	Description
X'01'	Source color value repeating group. The repeating group specifies a list of color values that are sources of a color mapping.
X'02'	Target color value repeating group. The repeating group specifies a list of color values that are targets of a color mapping.

ColSpce Is a code that defines the color space and the encoding for the color specification. Color spaces are defined in the MO:DCA Color Specification (X'4E') triplet; see "Color Specification Triplet X'4E'" on page 367. Only color spaces that are not defined in the X'4E' triplet, or color spaces that have a special meaning when used in a CMT, are described here.

Value **Description**

X'06' Highlight color space. This is the same color space as that defined in the Color Specification (X'4E') triplet. In addition, if this color space is specified in a source repeating group, a value of X'FF' for the percent coverage parameter indicates that all percentages of this parameter for the specified highlight color are mapped to the target color.

Application Note: When the Highlight Color space is specified in a target repeating group, the percent coverage parameter is normally only supported for areas such as object areas and graphic fill areas. For other data types this parameter is normally simulated with 100% coverage.

Implementation Note: The percent shading parameter for highlight colors is currently not supported in AFP environments.

X'40' Standard OCA color space. This is the same color space as that defined in the Color Specification (X'4E') triplet. All syntactically valid color values defined in the Standard OCA Color Value Table are supported for mapping. For a list of all valid color values, see "Standard OCA Color Value Table" on page 473.

X'50' GOCA Pattern Fill space. Component 1 defines the GOCA pattern set local ID as specified by the Set Pattern Set drawing order, and must be set to X'00' to select the GOCA default pattern set. ColSize1 is set to X'08' and defines the number of bits used to specify component 1. Component 2 defines a code point, as specified by the Set Pattern Symbol drawing order, that selects a specific pattern symbol from the default pattern set and is in the range X'00'–X'10', X'40'. ColSize2 is set to X'08' and defines the number of bits used to specify component 2. ColSize3 and ColSize4 are reserved and must be set to zero. If this color space is specified in a source repeating group, the pattern fill is replaced by the target color value independent of any color that may have been specified for the pattern in the GOCA data. If the pattern fill is not to be replaced by a color, this pattern should not be mapped. For a description of graphics area fill, pattern sets, and pattern symbols, see the *Graphics Object Content Architecture for Advanced Function Presentation Reference*.

ColSize1–Colsize4
For a definition of these parameters, see the description of the Color Specification (X'4E') triplet.

ObjSel Is a code that defines the data type to which the color values specified in the source repeating group apply.

Value Description

X'00' The parameter is not specified. This value must be used in target repeating groups.

X'6B' The source color values apply to object areas.

X'7B' The source color values apply to data in IM Image objects.

X'9B' The source color values apply to data in PTOCA text objects.

X'AF' The source color values apply to page presentation spaces whose color is specified with a Color Specification (X'4E') triplet.

X'BB' The source color values apply to data in GOCA graphics objects.

X'DF' The source color values apply to overlay presentation spaces whose color is specified with a Color Specification (X'4E') triplet.

X'EB' The source color values apply to data in BCOCA bar code objects.

X'FB' The source color values apply to data in non-tiled bi-level IOCA image objects.

X'FE' The source color values apply to all PTOCA, GOCA, BCOCA, non-tiled bi-level IOCA, and IM Image data objects.

X'FF' The source color values apply to all objects, object areas, and presentation spaces.

Color Values Is a sequential list of color values in the defined format and encoding. For source repeating groups, these values, when encountered in one of the specified source object types, are mapped to target values. For target repeating groups, these are the values that are rendered by the presentation device in place of the corresponding source color values.

Color Mapping Table Exception Condition Summary

An exception condition exists when the following is detected:

- The table is a color mapping table and does not contain at least one source repeating group and one target repeating group
- The table is a color mapping table and contains a source repeating group that does not have a matching target repeating group
- The table contains invalid data.

The Color Table Resource

The Color Table is preceded by the Begin Color Attribute Table structured field and is terminated by the End Color Attribute Table structured field. Within this bracket, the color table definition is carried in a set of Color Attribute Table structured fields.

Note: The Color Table resource is used only in MO:DCA-L data streams.

Color Representation in MO:DCA–L Data Streams and IOCA FS11 Objects

Colors are represented by one basic color model, RGB, and one subsidiary model, grayscale, which has an architected representation and an architected conversion to RGB.

- RGB is the only form allowed for interchange.
- Grayscale is provided for compatibility with existing products.
- Entries can be loaded as explicit values or can be generated by a formula based on a breakdown of each index.
- A variable number of bits per component is supported, for instance, 4 red bits, 5 green bits and 3 blue bits.
- Interchange forms are limited to RGB only, and to a maximum of 8 bits per component.

RGB Representation

The RGB representation of colors is based on the way in which display terminals create color. It defines each color to be composed of various proportions of three primary colors: red, green and blue.

The Color Table assumes the x,y chromaticity coordinates defined by the Society of Motion Picture and Television Engineers (SMPTE) recommended practices. Specifically, the x,y chromaticity coordinates for the three primary colors and the reference white point are defined in SMPTE RP 145-1987 entitled *Color Monitor Colorimetry* and RP 37-1969 entitled *Color Temperature for Color Television Studio Monitors*, respectively. The reference white point is commonly known as *Illuminant* D_{6500} or simply *D65*. The recommended gamma is 2.2.

The SMPTE-defined x,y chromaticity coordinates are:

Red	x = 0.630, y = 0.340
Green	x = 0.310, y = 0.595
Blue	x = 0.155, y = 0.070
White point	x = 0.313, y = 0.329

where x and y are the coordinates within the Commission Internationale de l'Eclairage (CIE) chromaticity diagram.

This does not mean that any particular device is required to produce these exact values. It defines the intent of the user. A system's software and hardware would be expected to produce a reasonable match with these values.

Grayscale Representation

The Y component represents the intensity, and ranges from 0, which corresponds to black, through 1. The basic formula to convert from RGB to grayscale is as follows:

```
Y = 0.299R + 0.587G + 0.114B
```

Color Table Definition in MO:DCA-L Data Streams and IOCA FS11 Objects

The definition consists of a base part, followed by one or more self-defining parameters (SDP). Each SDP defines a set of entries to be loaded into the color table.

There are two types of SDP:
- Element list
- Bit Generator

Each SDP is processed in turn. The entries defined by each SDP replace any entries defined by a previous SDP.

Base Part

Offset	Type	Name	Range	Meaning
0	BITS	FLAGS		
Bit 0			B'0'	Only valid value
Bit 1		RESET	B'0', B'1'	**B'0'** Do not reset LCT **B'1'** Reset LCT
Bits 2–7			B'000000'	Only valid value
1			X'00'	Only valid value
2	CODE	LCTID	X'00'–X'FF'	Local identifier of the color table

The base part defines the initialization conditions for the Color Table.

RESET Has the following values:

Value	**Description**
B'1'	The color table is reset prior to setting according to the following SDPs.
B'0'	No reset is performed.

LCTID Is a local identifier of the color table.

Element List Self-Defining Parameter

Offset	Type	Name	Range	Meaning
0	UBIN	LEN	X'0B'–X'FF'	Length of this parameter
1	CODE		X'01'	Type: **X'01'** Element List
2			X'00'	Only valid value
3	CODE	FORMAT	X'01', X'02'	**X'01'** RGB **X'02'** Grayscale **All others** Reserved
4–6	UBIN	INDEX_1		Starting index for load
7	UBIN	SIZE_1	X'00'–X'FF'	Number of bits in component 1
8	UBIN	SIZE_2	X'00'–X'FF'	Number of bits in component 2
9	UBIN	SIZE_3	X'00'–X'FF'	Number of bits in component 3
10	UBIN	TRILEN	X'00'–X'F4'	Number of bytes in each element
11–*n*		ELEMENTS		Color elements

An Element List SDP defines a contiguous block of entries in the Color Table by defining the explicit content of each entry as a set of values.

FORMAT Specifies the format of each element in the element list:

RGB Each element consists of a set of red, green, blue intensity values. The set is in the order red, green, blue.

Grayscale Each element consists of a Y-component.

INDEX_1 Specifies the position in the Color Table where the first element is to be loaded.

SIZE_1—SIZE_3

Specify the number of bits in each part of the element. For example, if FORMAT is RGB, then SIZE_1 specifies the number of red bits, SIZE_2 specifies the number of green bits and SIZE_3 specifies the number of blue bits.

The maximum integer value of each component of the color is determined by the corresponding SIZE parameter. Thus, M1, M2 and M3 are computed using the following formulas:

$M1 = (2^{SIZE\_1} - 1)$
$M2 = (2^{SIZE\_2} - 1)$
$M3 = (2^{SIZE\_3} - 1)$

TRILEN Specifies the length of each element in bytes.

ELEMENTS Is a variable sized list of elements. Each element contains three components. The size of each component is an integral number of bytes, given by the formula:

```
COMP_i = 1 + INT((SIZE_i – 1) / 8)
```

where SIZE_i is the number of bits in that component. The value of each component is right-aligned in the bytes and padded with zeros, giving a set of integers (I1,I2,I3). I1, I2 and I3 are used to generate values for each component as defined in "Calculation of Color Value" on page 487.

- If TRILEN is greater than the sum of COMP_i, then each element is padded on the left with X'00' to make its length equal to TRILEN.
- If TRILEN is less than the sum of COMP_i, then each element is truncated on the left to make its length equal to TRILEN.

Successive values are loaded into successive positions in the table until the element list is exhausted.

Bit Generator Self-Defining Parameter

Offset	Type	Name	Range	Meaning
0	UBIN	LEN	X'0A'	Length of this parameter
1	CODE		X'02'	Type: **X'02'** Bit Generator
2	BITS	FLAGS		
Bit 0		ASFLAG	B'0', B'1'	**B'0'** Additive **B'1'** Subtractive
Bits 1–7			B'0000000'	Only valid value

Offset	Type	Name	Range	Meaning
3	CODE	FORMAT	X'01', X'02'	**X'01'** RGB **X'02'** Grayscale **Al others** Reserved
4–6	UBIN	INDEX_1		Starting index for load
7	UBIN	SIZE_1	X'00'–X'FF'	Number of bits in component 1
8	UBIN	SIZE_2	X'00'–X'FF'	Number of bits in component 2
9	UBIN	SIZE_3	X'00'–X'FF'	Number of bits in component 3

A Bit Generator SDP defines a contiguous block of entries in the Color Table, by defining how each entry is to be generated from its index value.

ASFLAG Specifies the meaning of the color values:

Additive The maximum color value represents full intensity of that color and the minimum color value represents zero intensity of that color. For example, in a black-and-white system, the minimum color value (usually zero) means black, and the maximum value means white.

Subtractive The minimum color value represents full intensity of that color and the maximum color value represents zero intensity of that color. For example, in a black-and-white system, the minimum color value (usually zero) means white, and the maximum value means black.

FORMAT Specifies the breakdown format for each value:

RGB Each value is to be treated as a set of red, green, blue intensity values. The set is in the order red, green, blue.

Grayscale Each value is to be treated as a Y-component.

INDEX_1 Specifies the position in the Color Table where the first element is to be loaded.

SIZE_1–SIZE_3

Specify the number of bits in each part of the value. The sum of these sizes, N = SIZE_1 + SIZE_2 + SIZE_3, defines how many color values are to be loaded, namely 2^N. The maximum integer value of each component of the color is determined by the corresponding SIZE parameter, giving M1, M2 and M3 respectively, thus:

$$M1 = (2^{SIZE\_1} - 1)$$
$$M2 = (2^{SIZE\_2} - 1)$$
$$M3 = (2^{SIZE\_3} - 1)$$

For each index, from INDEX_1 through (INDEX_1 + 2^N − 1):

1. INDEX_1 is subtracted from the index, giving a value to be broken down.
2. This value is converted to a binary integer of N bits.
3. This integer is then treated as a bit string and broken down, from left to right, into three substrings, with lengths SIZE_1, SIZE_2 and SIZE_3 respectively.

4. Each of these substrings is then converted back to a binary integer, treating the leftmost bit as most significant. This process produces a set of integers (I1,I2,I3). The process is illustrated in “Example of Index Breakdown.”
5. If the ASFLAG is set, then each integer is reversed by subtracting it from the corresponding maximum; M1, M2, or M3.
6. I1, I2 and I3 are then used to generate values for each component as defined in “Calculation of Color Value.”

Successive values are loaded into successive positions in the table until 2^N colors have been loaded.

Calculation of Color Value

Each color value is a set of values (V1,V2,V3) where each value is in the range 0 through 1. The meaning of these values depends on the FORMAT parameter:

FORMAT X'01'
The values represent red, green, blue proportions.

FORMAT X'02'
The values represent Y values.

Each of these values is generated from the corresponding integers, I1, I2 and I3, and the corresponding maximum value, M1, M2 and M3, respectively.

Let I be an integer and M be the corresponding maximum value. Then the formulas defining this conversion are as follows:

```
If  I < M / 2  then  V =  I / (M + 1)
If  I ≥ M / 2  then  V =  (I + 1) / (M + 1)
```

These formulas produce values that *migrate* when the number of bits representing the color is increased. The values also map naturally using a *best* fit to the nearest fraction on an *equal step* device. This is illustrated in Table 26, which assumes a 3-bit representation (integers 0 through 7).

Table 26. MO:DCA-L: Calculating Color Values

	Value		3-bit Device Fit	
Integer	**Fraction**	**Decimal**	**Decimal**	**Fraction**
0	0 / 8	0.000	0.0000	0 / 7
1	1 / 8	0.125	0.1429	1 / 7
2	2 / 8	0.250	0.2857	2 / 7
3	3 / 8	0.375	0.4286	3 / 7
4	5 / 8	0.625	0.5714	4 / 7
5	6 / 8	0.750	0.7143	5 / 7
6	7 / 8	0.875	0.8571	6 / 7
7	8 / 8	1.000	1.0000	7 / 7

Example of Index Breakdown

The process of breaking down an index into three parts, as described under “Bit Generator Self-Defining Parameter” on page 485, is illustrated in the following example.

Suppose that:

```
SIZE_1 = 2
SIZE_2 = 1
SIZE_3 = 3
```

Thus the maximum integer values are:

```
M1 = 3
M2 = 1
M3 = 7
```

Each index value is converted to a 6-bit number, and broken down into substrings of 2, 1, and 3 bits.

Some sample index values are shown in Table 27.

Table 27. MO:DCA-L: Sample Index Values

Index	String	Substrings	I1	I2	I3
3	000011	00 0 011	0	0	3
9	001001	00 1 001	0	1	1
27	011011	01 1 011	1	1	3
45	101101	10 1 101	2	1	5

Interchange

In interchange, there are a number of limitations on the format of the Color Table:

- The only forms permitted are RGB LIST and RGB GENERATOR.
- If an RGB GENERATOR SDP is used, then that must be the only SDP.
- The maximum value permitted in each of SIZE_1, SIZE_2 and SIZE_3 is 8. Thus, each RGB component is limited to values 0 through 255.

Image IDE Structure in IOCA FS11 Objects

The IDE STRUCTURE parameter in IOCA is a subset of the Color Table syntax, and supports only GENERATOR forms of RGB and grayscale. The *starting index* is always zero. Its syntax is defined by IOCA.

Carrying Color Tables in MO:DCA–L Data Streams

Color tables are carried in MO:DCA-L as a required resource within the main Resource Group, or as a resource within the optional image objects.

The following tables show the required formats for the two cases:

Main Resource Group

Either a set of Element List Self-Defining Parameters, one for each table entry, or a single Bit Generator Self-Defining Parameter are required but are mutually exclusive.

Base Part:

Offset	Type	Name	Range	Meaning
0	BITS	FLAGS		
Bit 0			B'0'	Only valid value
Bit 1		RESET	B'1'	Reset LCD
Bits 2–7			B'000000'	Only valid value

Offset	Type	Name	Range	Meaning
1			X'00'	Only valid value
2	CODE	LCTID	X'00'	Local identifier of the color table

Element List Self-Defining Parameter:

Offset	Type	Name	Range	Meaning
0	UBIN	LEN	X'0F'	Length of this parameter
1	CODE		X'01'	Type: **01** Element List
2			X'00'	Only valid value
3	CODE	FORMAT	X'01'	RGB
4–6	UBIN	INDEX_1		Starting index for load
7	UBIN	SIZE_1	X'08'	Number of bits in component 1
8	UBIN	SIZE_2	X'08'	Number of bits in component 2
9	UBIN	SIZE_3	X'08'	Number of bits in component 3
10	UBIN	TRILEN	X'04'	Number of bytes in each element
11–14		ELEMENTS		Color elements

Bit Generator Self-Defining Parameter:

Offset	Type	Name	Range	Meaning
0	UBIN	LEN	X'0A'	Length of this parameter
1	CODE		X'02'	Type: **02** Bit Generator
2	BITS	FLAGS		
Bit 0		ASFLAGS	B'0'	Additive
Bit 1–7			B'0000000'	Only valid value
3	CODE	FORMAT	X'01'	RGB
4–6	UBIN	INDEX_1	X'000000'	Starting index for load
7	UBIN	SIZE_1	X'08'	Number of bits in component 1
8	UBIN	SIZE_2	X'08'	Number of bits in component 2
9	UBIN	SIZE_3	X'08'	Number of bits in component 3

Image Resource Group

A set of Element List Self-Defining Parameters are allowed which must be ordered in ascending index order starting at index zero.

Base Part:

Offset	Type	Name	Range	Meaning
0	BITS	FLAGS		
Bit 0			B'0'	Only valid value
Bit 1		RESET	B'0'	Do not reset LCD
Bits 2–7			B'000000'	Only valid value

Offset	Type	Name	Range	Meaning
1			X'00'	Only valid value
2	CODE	LCTID	X'01'	Local identifier of the color table

Element List Self-Defining Parameter:

Offset	Type	Name	Range	Meaning
0	UBIN	LEN	X'0E'–X'FE'	Length of this parameter
1	CODE		X'01'	Type: **01** Element List
2			X'00'	Only valid value
3	CODE	FORMAT	X'01'	RGB
4–6	UBIN	INDEX_1		Starting index for load
7	UBIN	SIZE_1	X'08'	Number of bits in component 1
8	UBIN	SIZE_2	X'08'	Number of bits in component 2
9	UBIN	SIZE_3	X'08'	Number of bits in component 3
10	UBIN	TRILEN	X'03'	Number of bytes in each element
11–n		ELEMENTS		Color elements

Appendix B. Resource Access Table (RAT)

IBM Font Interchange Information

This appendix formerly contained information on acceptable values that may be used in the MO:DCA Map Coded Font structured field to identify a particular font in the Expanded Core Fonts and DBCS Fonts that IBM supports. Because of the increase in the number of fonts supported in AFP environments, it is no longer practical to maintain this material in an appendix. For detailed information on the Font Object Content Architecture (FOCA) fonts that may be referenced with a Map Coded Font (MCF) structured field in a MO:DCA data stream, please see the font publications listed in "Related Publications" on page vi. In particular, please see the following documents:

- *Font Summary for AFP Font Collection*
- *IBM AFP Fonts: Technical Reference for Code Pages*

Note: The referenced documents use the term *character set* as a short form of the qualified term *font character set*. The latter form is used throughout this book. In this context, the two forms are equivalent.

The Resource Access Table (RAT)

The Resource Access Table (RAT) is used to map a resource name specified in the MO:DCA data stream to information used to find and process the resource on a given system. For TrueType fonts (TTFs) and OpenType fonts (OTFs), the resource name is a full font name. It is specified in the data stream on a Map Data Resource (MDR) structured field.

Resource Access Table in MO:DCA–P Environments

The Resource Access Table (RAT) is installed on a given system by an application program. It is updated whenever new resources that need to be accessed through a RAT are installed on that system, or whenever such resources are updated - such as when a new version of a resource replaces an existing version. The installed RAT remains active until it is updated or replaced. If no RAT is active, resources which require a RAT to be accessed cannot be processed.

The RAT resides in the directory that it represents. There can be multiple RATs in a system, one for each directory. The file names in the RAT do not contain path information.

Implementation Note: In AFP systems, the file name of the RAT in a given directory is hard-coded as *IBM_DataObjectFont.rat*

Resource Access Table in IPDS Environments

The Resource Access Table is not used at the IPDS level.

Resource Access Table Definition

The table definition consists of a table header followed by zero or more variable-length repeating groups. The table header specifies information that applies to the whole table including an identifier for the table, the length of the table, and a table creation/update time stamp. A repeating group consists of a header followed by zero or more variable-length table vectors. Each repeating group specifies the information needed to access and process a specific resource. The repeating group content is defined by the resource object type, which is identified by the resource object-type OID. Repeating groups for a specific resource object type, such as repeating groups for TTFs or OTFs, have the same syntax. Only a single repeating group is allowed for a specific resource object. That is, a single resource object may only be defined and indexed once in the RAT. Repeating groups must be compact. This means that for a table vector that can be repeated, all occurrences of the vector must specify valid content, that is, the vectors cannot be empty unless there is only one occurrence of that vector.

Resource Access Table Syntax

Offset	Type	Name	Range	Meaning	M/O
Resource Access Table Header					
0–3	UBIN	Tlength	18–4,294,967,295	Table length	M
4–5	CODE	TBLid	1–65,534	Table ID	M
6	CODE	TBLtpe	X'02'	Table type: **X'02'** Resource Access Table	M
7–16	CODE	UTStmp		Universal Date and Time Stamp	M
17	CODE	InstInf	X'00', X'01'	Installer information: **X'00'** Installer information not specified; this parameter ends the table header **X'01'** Installer information specified in bytes 18 - 57	
40 bytes of Font Installer information that are only specified if InstInf = X'01'; these bytes are optional as a unit.					
18–49	CHAR	InstNme		Name of Font Installer application	O
50	UBIN	InstVrs		Version number of Font Installer application	O
51	UBIN	InstRel		Release level of Font Installer application	O
52	UBIN	InstMod		Modification level of Font Installer application	O
53	UBIN	InstSrv		Service level of Font Installer application	O
54–57				Reserved; must be zero	O
Zero or more variable-length repeating groups					

Offset	Type	Name	Range	Meaning	M/O
Repeating Group Structure					
0–1	UBIN	RGLngth	22–65,535	Repeating group length	M
2				Reserved; must be zero	M
3	CODE	RGTpe	X'10'	Repeating group type: **X'10'** Resource access table repeating group **All others** Reserved	M
4–5	BITS	RGFlgs		Repeating group flags; semantics defined by resource object-type	M
6–21	CODE	ObjTpe		Object-type OID for resource being accessed	M
Zero or more variable-length table vectors in fixed order. The table vector semantics and their order in the repeating group are defined by the resource object type					

Offset	Type	Name	Range	Meaning	M/O
			Table Vector Structure		
0	UBIN	TVLngth	2–252	Table vector length; a length of 2 indicates the table vector data is not specified	M
1	CODE	TVid		Table vector identifier	M
2–251		TVData		Table vector data	O

Resource Access Table Semantics

TBLlngth Contains the length of the table, including this length field, in bytes.

TBLid Contains the identifier for the table.

TBLtpe Is a code that defines the type of table.

Value	Description
X'02'	Resource Access Table. The table specifies information needed to access and process a resource.

UTStmp Contains the time stamp that specifies when the table was created or when it was last updated. The time stamp is specified with 10 bytes using the syntax specified in bytes 3-12 of the Universal Date and Time Stamp (X'72') triplet, see “Universal Date and Time Stamp Triplet X'72'” on page 394.

InstInf Is a code that defines whether the table header contains information about the Font Install application that generated this RAT.

Value	Description
X'00'	No additional Font Installer information is specified. This parameter terminates the table header. No additional RAT header bytes are allowed and will cause a RAT processing error if specified.
X'01'	40 additional bytes of Font Installer information are specified in bytes 18 - 57 of the RAT header.

InstNme Is a character string that identifies the Font Installer application, encoded in UTF-16BE. The name is left-justified and padded with blanks (space character = X'0020').

Architecture Note: The IBM Font Installer Application is identified as “IBM FI”.

InstVrs Version number of the Font Installer application. For example, version 1 is identified with InstVrs = X'01'.

InstRel Release level of the Font Installer application. For example, release level 2 is identified with InstRel = X'02'.

InstMod Modification level of the Font Installer application. For example, modification level 3 is identified with InstMod = X'03'.

InstSrv Service level of the Font Installer application. For example, service level 4 is identified with InstSrv = X'04'.

RGlngth Contains the length of the repeating group, including this length field, in bytes.

RGtpe Is a code that defines the type of repeating group.

Value	Description
X'10'	Resource Access Table repeating group. The repeating group specifies information needed to access and process a resource.

RGFlgs Specifies processing flags for the resource. The flag semantics are defined by the resource object type.

ObjTpe Specifies the object-type OID for the resource that is accessed and processed with this repeating group. The object-type OID for resource objects supported in MO:DCA environments is registered in "Object Type Identifiers" on page 535. The OID is left-justified and padded with zeros. For example, the object-type OID for TrueType font objects is X'06072B120004010133'. This OID is specified in the ObjTpe parameter as X'06072B12000401013300000000000000'.

Resource Access Table Exception Condition Summary

An exception condition exists when the following is detected:
- The RAT header does not specify TBLtpe = X'02'
- A RAT repeating group header does not specify RGTpe = X'10'
- The ObjTpe parameter does not specify a supported object-type OID
- The table contains invalid data.

Repeating Group Definition for TrueType and OpenType Font Resources

TrueType and OpenType font resources are identified by object-type OID = X'06072B120004010133'. They are referenced in the MO:DCA data stream using Map Data Resource (MDR) structured fields. The reference specifies a full font name that is also specified by the font manufacturer in the font naming table. The full font name in the font may be specified in multiple languages; the supported encoding is UTF-16. The full font name from the font reference is used to index the RAT repeating groups, which specify the full font name of a TrueType/OpenType font in all supported languages using the UTF-16 encoding. Within a repeating group the full font names in all languages must be sorted so that the UTF-16 code point sequences for the names are in ascending numerical order. The repeating groups are then sorted so that the UTF-16 code point sequences for the first full font names in each repeating group are in ascending order. Note that in MO:DCA environments, all UTF-16 data is considered to be in big-endian format (UTF-16BE).

Repeating Group Flag Definitions for TrueType and OpenType Font Resources

Following are the flag definitions for TrueType and OpenType font resources.

RGFlgs Provide additional information for accessing and processing the TrueType/OpenType font resource. RGFlgs bits have the following descriptions:

Bit **Description**

0 TrueType Collection (TTC)

B'0' The font is not packaged in a TTC. If this bit is set to B'0', the TTF/TTC File Name vector (TVid = X'04') references a TrueType/OpenType font (TTF/OTF), and the TTF/TTC Object OID vector (TVid = X'08'), if not empty, specifies an object OID for the font. The TTC Font Index vector (TVid = X'1A') should be empty and is ignored.

B'1' The font is packaged in a TTC. If this bit is set to B'1', the TTF/TTC File Name vector (TVid = X'04') references a TrueType Collection (TTC), and the TTF/TTC Object OID vector (TVid = X'08'), if not empty, specifies an object OID for the collection. The TTC Font Index vector (TVid = X'1A') must specify a valid index, and the collection must contain and index a version of the referenced font that is logically equivalent to the font.

1 Linked Fonts

B'0' The font does not have any linked fonts. If this bit is set to B'0', the Linked TTF/OTF Full Font Name vector (TVid = X'24') should be empty and is ignored.

B'1' The font has linked fonts. A linked font is a complete TTF/OTF that is processed as a logical extension of the base font. If this bit is set to B'1', the Linked TTF/OTF Full Font Name vector (TVid = X'24') and any additional Linked TTF/OTF Full Font Name vectors must specify valid full font names for TTFs/OTFs. Note that linked fonts can be packaged in a TTC. Note also that only one level of linking is supported. That is, if a linked font specifies its own linked fonts, these 'secondary' linked fonts are not processed as linked fonts for the original base font.

2 Private

B'0' The installer considers this font or the TTC that contains this font to be a public resource. A public resource is a candidate for resource capture by a printer. A public resource may also be resident in the printer, and this version can be used if the object OID matches the object OID associated with the resource reference.

B'1' The installer considers this font or the TTC that contains this font to be a private resource. A private resource is not a candidate for resource capture by printers. A private resource is always downloaded to the printer; if an object OID has been generated for the resource, it is ignored.

3 Embed

B'0' The installer does not allow this font or the TTC that contains this font to be embedded inline into a printfile–level resource group.

B'1' The installer allows this font or the TTC that contains this font to be embedded inline into a printfile–level resource group.

4 Capture

B'0' The installer does not allow this font or the TTC that contains this font to be captured.

B'1' The installer allows this font or the TTC that contains this font to be captured. A number of requirements must be met before the presentation system will actually let resource capture take place:

- The font or collection must be identified as ‘public’ (RGFlgs bit 2 set to B'0') by the installer
- The font or collection must have an object OID associated with it
- The font or collection must be in a location that the presentation system considers secure.

5–15 Reserved; all bits must be B'0'.

Architecture Note:

1. The setting of RGFlgs bits 2-4 reflect not only the intent of the person running the install process, but also the processing of the font permission bits (fsType parameter in the OS/2 Table of the TTF file) by the install program. For example, if RGFlgs bit 2 = B'0' (font is public), this means (i) the intent of the person running the install process is to install the font as a public font, and (ii) the font permission bits allow the font to be treated as a public font.
2. If the RAT repeating group maps a full font name to the file name of a collection, the installer needs to ensure that RGFlgs bits 2-4 apply to all fonts in the collection. For example, if RGFlgs bit 4 = B'1' (capture allowed), then this needs to reflect all fonts in the collection, since the complete collection may end up being captured.

Table Vector Definitions for TrueType and OpenType Font Resources

Following are the table vectors defined for TrueType and OpenType font resources. The table vectors must appear in the order shown. Unless indicated otherwise, each table vector must occur once, regardless of whether its data parameter is specified or not. If a table vector contains no data, its length must be set to X'02' to indicate that the table vector data is not specified. This is also referred to as an empty table vector. Table vectors within a RAT repeating group must be compact. This means that for a table vector that can be repeated, all occurrences of the vector must specify valid content, that is, the vectors cannot be empty unless there is only one occurrence of that vector.

Offset	Type	Name	Range	Meaning	M/O
TrueType/OpenType Font (TTF/OTF) Full Font Name; table vector may be repeated to specify the full font name in all supported languages					
0	UBIN	TVLngth	4–252; even values only	Table vector length	M
1	CODE	TVid	X'01'	Table vector identifier	M
2–251	CHAR	FFName		Full font name of the base font. This parameter must be specified.	M
TrueType/OpenType Font or TrueType/OpenType Collection (TTC) File Name; table vector must be specified only once					

Offset	Type	Name	Range	Meaning	M/O
0	UBIN	TVLngth	4–252; even values only	Table vector length	M
1	CODE	TVid	X'04'	Table vector identifier	M
2–251	CHAR	FileNme		File name with which the font or the collection that contains the font has been stored in the presentation system's resource library. RGFlgs bit 0 = B'0' indicates that the file name references a TrueType/OpenType font (TTF/OTF). RGFlgs bit 0 = B'1' indicates that the file name references a TrueType Collection (TTC). The file name does not include path information. This parameter must be specified.	M
TrueType/OpenType Font or TrueType/OpenType Collection Object OID; table vector must be specified only once					
0	UBIN	TVLngth	2–131	Table vector length; a length of 2 indicates the table vector data is not specified	M
1	CODE	TVid	X'08'	Table vector identifier	M
2–130	CODE	ObjOID		The object OID that is assigned to the font or to the collection that contains the font. RGFlgs bit 0 = B'0' indicates that the object OID is associated with a TrueType/OpenType font (TTF/OTF). RGFlgs bit 0 = B'1' indicates that the object OID is associated with a TrueType Collection (TTC). The length of this parameter must reflect the length of the actual OID; padding bytes are not allowed. The object OID enables the font or the collection to be captured and made resident in the printer.	O
TrueType/OpenType Collection Font Index; table vector must be specified only once					
0	UBIN	TVLngth	2, 4	Table vector length; a length of 2 indicates the table vector data is not specified	M
1	CODE	TVid	X'1A'	Table vector identifier	M

Offset	Type	Name	Range	Meaning	M/O
2–3	UBIN	FntIndx		The index used to locate the TTF/OTF in the TTC. This is an index into the array of offsets that comprise the 4th parameter in the TTC Header. Each offset points to the directory of a specific TTF/OTF in the TTC. An index value of X'0000' selects the first offset, a value of X'0001' selects the second offset, a value of (n-1) selects the nth offset. This index must be specified if RGFlgs bit 0 = B'1'. This vector should be empty and is ignored if RGFlgs bit 0 = B'0'.	O
Linked TrueType/OpenType Font Full Font Name; table vector may be repeated to specify multiple linked fonts					
0	UBIN	TVLngth	2–252; even values only	Table vector length; a length of 2 indicates the table vector data is not specified	M
1	CODE	TVid	X'24'	Table vector identifier	M
2–251	CHAR	LFFName		Full font name of the linked font. This parameter must be specified if RGFlgs bit 1 = B'1'.	O

Table Notes:

1. All character data in the table vectors is encoded in UTF-16BE. This encoding is characterized by the following parameters:

   Encoding scheme ID - as carried in the Encoding Scheme ID (X'50') triplet: X'7200'

   CCSID - as carried in the Coded Graphic Character Set Global Identifier (X'01') triplet (CCSID form) - 1200 (X'04B0'

   Note that in MO:DCA environments, all UTF-16 data is considered to be in big-endian format (UTF-16BE).
2. If multiple TrueType/OpenType Font (TTF/OTF) Full Font Name table vectors are specified, each vector must specify a valid full font name.
3. If multiple Linked TrueType/OpenType Font (TTF/OTF) Full Font Name table vectors are specified, each vector must specify a valid full font name.
4. The order in which multiple Linked TrueType/OpenType Font (TTF/OTF) Full Font Name table vectors are specified in the repeating group determines the order in which the linked fonts are processed by the presentation system:
   - The base font is processed first, followed by the first linked font in the repeating group, followed by the next linked font in the repeating group, and so on; the last linked font in the repeating group is processed last.
   - If an external (printfile-level) resource group is specified for the printfile, this resource group is searched first for a specified linked font. If the specified linked font is not found in the resource group, the RAT is accessed to locate the linked font in a library. Note that linked fonts can be packaged in a TTC.
   - Only one level of linking is supported. That is, if a linked font specifies its own linked fonts, these ’secondary’ linked fonts are not processed as linked fonts for the original base font.
5. A specific linked font should only be specified once in a given repeating group.

Appendix C. MO:DCA Migration Functions

This appendix:
- Describes obsolete structured fields and triplets that may occur in a MO:DCA data stream
- Describes retired structured fields and triplets that may occur in a MO:DCA data stream
- Describes coexistence functions that may occur in a MO:DCA data stream

The objective in defining obsolete, retired, and coexistence functions is twofold:
- To allow existing IBM applications to run unchanged
- To provide a clear growth direction for future applications

Migration Functions

The migration functions are divided into three different categories:

- *Obsolete functions.* These are objects, structured fields, triplets, and parameters that will be accepted but ignored. New products must not generate these functions.
- *Retired functions.* These are objects, structured fields, triplets, and parameters whose use has been retired except for specific products. Only these specific products may use these functions. All other products must not use these functions, that is, generators must not generate these functions and receivers must ignore them.
- *Coexistence functions.* These are objects, structured fields, triplets, and parameters whose function has been enhanced or superseded by newer functions. In this case, the old and new functions can *coexist*. New generators must generate the new functions. New receivers must process the new functions, but may also continue to process the old functions.

Obsolete Functions

Obsolete functions are objects, structured fields, triplets, and parameters that will be accepted but ignored. New products must not generate these functions.

Obsolete Structured Fields

The following four structured fields are obsolete in the current data stream, but are still allowed to be present as constant data. AFP servers recognize these fields and ignore them:

- Composed-Text Control (CTC)
- Begin Form Environment Group (BFG)
- End Form Environment Group (EFG)
- Form Environment Group Descriptor (FGD)

The CTC can appear as a constant in the Active Environment Group of a page. The BFG, EFG, and FGD can appear optionally in the Medium Map object of a Form Map.

New applications must not generate these structured fields.

Composed Text Control (CTC)

CTC (X'D3A79B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A79B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–9		ConData		Constant data	M	X'06'

CTC Semantics:

ConData Constant data. Must be set to X'0000 0000 0000 0000 2D00'.

Begin Form Environment Group (BFG)

BFG (X'D3A8C5') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A8C5'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	**Type**	**Name**	**Range**	**Meaning**	**M/O**	**Exc**
0–7	CHAR	FEGName		Name of the Form Environment Group	O	X'02'

BFG Semantics:

FEGName Is the name of the form environment group.

End Form Environment Group (EFG)

EFG (X'D3A9C5') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A9C5'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	FEGName		Name of the Form Environment Group	O	X'02'

EFG Semantics:

FEGName Is the name of the form environment group being terminated. If a name is specified, it must match the name in the most recent Begin Form Environment Group structured field in the Form Map. If the first two bytes in FEGName contain the value X'FFFF', the name matches any name specified on the Begin Form Environment Group structured field that initiated the current definition.

Form Environment Group Descriptor (FGD)

FGD (X'D3A6C5') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A6C5'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–3		ConData		Constant data	M	X'06'

FGD Semantics:

Constant data Must be set to X'0001 00FF'.

Obsolete Structured Field Names

The following structured fields are still in use, but have been renamed:

- Composed Text Data (CTX)
- Composed Text Descriptor (CTD)
- Begin Composed Text (BCT)
- End Composed Text (ECT)

Composed Text Data (CTX) Structured Field (X'D3EE9B')

This structured field has been renamed Presentation Text Data (PTX).

Composed Text Descriptor (CTD) Structured Field (X'D3A69B')

This structured field has been renamed Presentation Text Data Descriptor Format 1 (PTD-1).

Begin Composed Text (BCT) Structured Field (X'D3A89B')

This structured field has been renamed Begin Presentation Text (BPT).

End Composed Text (ECT) Structured Field (X'D3A99B')

This structured field has been renamed End Presentation Text (EPT).

Retired Functions

Retired functions are objects, structured fields, triplets, and parameters whose use has been retired except for specific products. Only these specific products may use these functions. All other products must not use these functions, that is, generators must not generate these functions and receivers must ignore them.

Retired Structured Fields

The following structured fields were previously retired but are now valid MO:DCA structured fields:
- Begin Resource (BR), see “Begin Resource (BRS)” on page 143.
- End Resource (ER), see “End Resource (ERS)” on page 171.

Retired Triplets

The following triplets have been retired:
- Text Orientation Triplet X'1D'
- Resource Object Type Triplet X'21'
- Line Data Object Position Migration Triplet X'27'
- Object Checksum Triplet X'63'
- Object Origin Identifier Triplet X'64'
- IMM Insertion Triplet X'73'

Text Orientation Triplet X'1D'

The use of this triplet is restricted to the MCF-2 structured field for 3800 compatibility for the following products:
- PSF/MVS
- PSF/VM
- PSF/VSE
- PSF/400
- PSF/2
- Infoprint Manager (IPM)
- 3800 printer
- Applications that generate MCF-2s in documents to be printed on the 3800 printer

The Text Orientation triplet is used to specify the text orientation for a coded font.

When the MCF-2 structured field is used to reference different sections of the same double-byte font, a Text Orientation (X’1D’) triplet may be specified in *any* of the repeating groups associated with the font and *need only* be specified in *one* of the repeating groups. However, if specified in more than one of the associated repeating groups, the value of all Text Orientation (X’1D’) triplets must be identical.

Triplet X'1D' Syntax:

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'1D'	Identifies the Text Orientation triplet	M	X'00'

Offset	Type	Name	Range	Meaning	M/O	Exc
2–3	CODE	IAxis	X'0000', X'2D00', X'5A00', X'8700'	Specifies the orientation of the Inline axis: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	M	X'06'
4–5	CODE	BAxis	X'0000', X'2D00', X'5A00', X'8700'	Specifies the orientation of the Baseline axis: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	M	X'06'

Triplet X'1D' Semantics:

Tlength Contains the length of the triplet.

Tid Identifies the Text Orientation triplet.

IAxis Specifies the orientation of the I-axis with respect to the X axis of the page or overlay. Valid values are the following:

Value	**I-Axis Orientation**
X'0000'	0 degrees
X'2D00'	90 degrees
X'5A00'	180 degrees
X'8700'	270 degrees
All others	Reserved

BAxis Specifies the orientation of the B-axis with respect to the X axis of the page or overlay. Valid values are the following:

Value	**B-Axis Orientation**
X'0000'	0 degrees
X'2D00'	90 degrees
X'5A00'	180 degrees
X'8700'	270 degrees
All others	Reserved

Structured Fields Using Triplet X'1D':

- “Map Coded Font (MCF) Format 2” on page 213

Resource Object Type Triplet X'21'

The use of this triplet is restricted to the BRS structured field in external (printfile-level) resource groups in AFP environments.

The Resource Object Type triplet identifies the type of object enveloped by the Begin Resource (BRS) and End Resource (ERS) structured fields.

Triplet X'21' Syntax:

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	10	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'21'	Identifies the Resource Object Type triplet	M	X'00'

Offset	Type	Name	Range	Meaning	M/O	Exc
2	CODE	ObjType	X'03', X'05'–X'06', X'40'–X'42', X'92'–X'A8', X'FB'–X'FE'	Specifies the object type: **X'03'** Graphics (GOCA) object **X'05'** Bar Code (BCOCA) object **X'06'** Image (IOCA) object **X'40'** Font Character Set object **X'41'** Code Page object **X'42'** Coded Font object **X'92'** Object Container **X'A8'** Document object **X'FB'** Page Segment object **X'FC'** Overlay object **X'FD'** Pagedef object **X'FE'** Formdef object	M	X'06'
3–9	CODE	ConData		Constant data	M	X'06'

Triplet X'21' Semantics:

Tlength Contains the length of the triplet.

Tid Identifies the Resource Object Type triplet.

ObjType Specifies the object type.

Value	**Description**
X'03'	Graphics (GOCA) object
X'05'	Bar Code (BCOCA) object
X'06'	Image (IOCA) object
X'40'	Font Character Set object
X'41'	Code Page object
X'42'	Coded Font object
X'92'	Object Container
X'A8'	Document object
X'FB'	Page Segment object
X'FC'	Overlay object
X'FD'	Pagedef object
X'FE'	Formdef object
All others	Reserved

ConData Constant data. Must be set to X'0000 0000 0000 00'.

Structured Fields Using Triplet (X'21'):
- “Begin Resource (BRS)” on page 143

Line Data Object Position Migration Triplet X'27'

The use of this triplet is restricted to the BBC, BGR, BII, BIM, IPS structured fields for the migration of line-data containing bar code objects, graphic objects, image objects, and page segments to MO:DCA document format. This triplet may be specified on these structured fields only for objects that occur directly in a page. The triplet may not be specified on objects in a resource group or in a resource library; if it is specified, it is ignored.

Triplet X'27' Syntax: Use of this triplet is restricted to the following products:
- ACIF
- PSF/MVS

- PSF/VM
- PSF/VSE
- PSF/2
- Infoprint Manager (IPM)
- PSF/400
- AFP Workbench

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	3	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'27'	Identifies the Line Data Object Position Migration triplet	M	X'00'
2	CODE	TempOrient	X'00'–X'03'	Location and orientation of coordinate system for object position and rotation: **X'00'** Standard page origin, 0° rotation **X'01'** Lower left origin, 270° rotation **X'02'** Lower right origin, 180° rotation **X'03'** Upper right origin, 90° rotation	M	X'06'

Triplet X'27' Semantics:

Tlength Contains the length of the triplet.

Tid Identifies the Line Data Object Position Migration triplet.

TempOrient Specifies a temporary page coordinate system (X,Y) that matches the text coordinate (I,B) system that was defined when the objects that specify this triplet were included in line data. The origin of the temporary coordinate system is specified as one of the four corners of the page presentation space. The orientation of the temporary coordinate system is specified as a rotation of the X axis with respect to the page presentation space X_p axis. The temporary coordinate system uses the same units of measure as the page coordinate system. The temporary coordinate system is used as follows:

- For objects in a page segment, the X'27' triplet may be specified on the IPS and has the following effect on object offset and orientation:
  - *IM image objects.* The image origin offset from the page segment origin is measured using the temporary (X,Y) coordinate system. If the image is celled, cell offsets from the image origin are also measured using the temporary (X,Y) coordinate system. The image rotation is measured using the page (X_p,Y_p) coordinate system.
  - *OCA objects (bar code, graphics, image).* If OBP byte 23 = X'00', the object area offset from the page segment origin and the object area rotation are measured using the temporary (X,Y) coordinate system. If OBP byte 23 = X'01', the object area offset from the page segment origin and the object area rotation are measured using the page (X_p,Y_p) coordinate system.

If specified on the IPS, the X'27' triplet overrides any X'27' triplet that is specified on the Begin structured field of an object in the page segment.

- For stand-alone objects, the X'27' triplet may be specified on the object Begin structured field and has the following effect on object offset and orientation:
  - *IM image objects.* The image origin offset is measured from the temporary (X,Y) coordinate system origin (X=0,Y=0) using the temporary (X,Y) coordinate system. If the image is celled, cell offsets from the image origin are also measured using the temporary (X,Y) coordinate system. The image rotation is measured using the page (X_p,Y_p) coordinate system.
  - *OCA objects (bar code, graphics, image).* If OBP byte 23 = X'00', the object area offset is measured from the temporary (X,Y) coordinate system origin (X=0,Y=0) using the temporary (X,Y) coordinate system. The object area rotation is also measured using the temporary (X,Y) coordinate system. If OBP byte 23 = X'01', the object area offset is measured from the page origin (X_p=0,Y_p=0) using the page (X_p,Y_p) coordinate system. Object area rotation is also measured using the page (X_p,Y_p) coordinate system.

The following values are defined:

Value	Description
X'00'	The temporary (X,Y) coordinate system is the page (X_p,Y_p) coordinate system. This is the standard MO:DCA page coordinate system that is used for object positioning and rotation. This coordinate system is used if this triplet is omitted.
X'01'	The temporary coordinate system origin is the lower-left corner of the page presentation space (X_p=0, Y_p=Y_{extent}). Its axes are rotated 270° from the axes of the page presentation space, so that the X axis increases from bottom to top and the Y axis increases from left to right.
X'02'	The temporary coordinate system origin is the lower-right corner of the page presentation space (X_p=X_{extent}, Y_p=Y_{extent}). Its axes are rotated 180° from the axes of the page presentation space, so that the X axis increases from right to left and the Y axis increases from bottom to top.
X'03'	The temporary coordinate system origin is the upper-right corner of the page presentation space (X_p=X_{extent}, Y_p=0). Its axes are rotated 90° from the axes of the page presentation space, so that the X axis increases from top to bottom and the Y axis increases from right to left.

Table 28 on page 510 provides a comparison of object position and rotation in line data and object position and rotation in MO:DCA data transformed from line data.

Table 28. 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 (X_p=0,Y_p=0). The offset is measured using the page (X_p,Y_p) coordinate system. The offset was adjusted to include the LND 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—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—Byte 23=X'01'	
(XoaOset,YoaOset) in OBP specify an offset from the page origin (X_p=0,Y_p=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 (X_p=0,Y_p=0). The offset is measured using the page (X_p,Y_p) coordinate system.
OCA Object Rotation—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—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 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 position.
IM Image Object Rotation	

Table 28. 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**
(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 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 position.
OCA Object Origin—OBP Byte 23= X'01'	
(XoaOset,YoaOset) in OBP specify an offset from the page origin (X_p=0,Y_p=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 (X_p=0,Y_p=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.

Structured Fields Using Triplet X'27':
- "Begin Bar Code Object (BBC)" on page 105
- "Begin Graphics Object (BGR)" on page 118
- "Begin IM Image Object (BII)" on page 526
- "Begin Image Object (BIM)" on page 120
- "Include Page Segment (IPS)" on page 197

Object Checksum Triplet X'63'

The use of this triplet is restricted to the BMO and BPS structured fields in external (printfile-level) AFP resource groups for the following products:
- PSF/MVS
- PSF/VSE
- RPM 2.0
- RPM 3.0
- PSF/2 (DPF)
- RMARK

The Object Checksum specifies a qualifier that can be used to identify or fingerprint an object.

Triplet X'63' Syntax:

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	6	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'63'	Identifies the Object Checksum	M	X'00'
2	CODE	Format	X'01'–X'02'	Specifies the format of the checksum: **X'01'** Object Cyclic Redundancy Check (CRC) **X'02'** Retired for private use	M	X'06'
3–4	UBIN	Qualifier	X'0000'–X'FFFF'	Object CRC check sum	M	X'06'
5	BITS	ClassFlgs		Object class flags. See “Triplet X'63' Semantics” for ClassFlgs bit definitions.	M	X'06'

Triplet X'63' Semantics:

Tlength
Contains the length of the triplet.

Tid Identifies the Object Checksum.

Format
Specifies the format of the checksum.

Value	Description
X'01'	Cyclic Redundancy Code (CRC) check sum
X'02'	Retired for private use
All others	Reserved

Application Note: Format X'02' is used in AFP environments for font resource management. For a description, see the *Font Object Content Architecture Reference*.

Qualifier
A two-byte value that may be used to support object identification based on the bit-content of the object. This value is the Cyclic Redundancy Check (CRC) check sum and is generated as follows:

1. All bits in the object, from the first bit in the Begin structured field to the last bit in the End structured field, are treated as coefficients of an nth order polynomial.
2. A second bit string is formed based on the coefficients of a generator polynomial, which is the CCITT V.41 polynomial defined as $X^{16} + X^{12} + X^{5} + 1$.
3. The object polynomial is divided by the generator polynomial using binary division on the bit strings that represent the coefficients of the two polynomials.
4. The remainder of this division is a polynomial of order less than 16. The coefficients of this polynomial are the CRC check sum.

ClassFlgs
Classifies objects for resource management. ClassFlgs bits have the following descriptions:

Bit	Description
0	Usage scope: **B'0'** Public resource object, unlimited usage **B'1'** Private resource object, limited usage
1	Resource retention indicator: **B'0'** Save resource **B'1'** Do not save resource
2–7	Reserved; all bits must be B'0'

Structured Fields Using Triplet X'63':
- "Begin Overlay (BMO)" on page 124
- "Begin Page Segment (BPS)" on page 137

Application Note: This triplet is also used on the following private font object structured fields in AFP environments:
- Begin Code Page (BCP)
- Begin Font Character Set (BFN)

Object Origin Identifier Triplet X'64'

The use of this triplet is restricted to the BMO and BPS structured fields in external (printfile-level) AFP resource groups for the following products:
- PSF/MVS
- PSF/VSE
- RPM 2.0
- PSF/2
- RMARK

The Object Origin Identifier triplet is used to identify the system on which an object originated.

Triplet X'64' Syntax:

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	61	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'64'	Identifies the Object Origin Identifier triplet	M	X'00'
2	CODE	System	X'01'–X'04'	Identifies originating system: **X'01'** MVS **X'02'** VM **X'03'** PC-DOS **X'04'** VSE	M	X'06'
3–10	CHAR	SysID		System ID and serial number	M	X'06'
11–16	CHAR	MedID		Storage media ID	M	X'06'
17–60	CHAR	DSID		Data set ID	M	X'06'

Triplet X'64' Semantics:

Tlength Contains the length of the triplet.

Tid Identifies the Object Origin Identifier triplet.

System Specifies the type of system on which the object originated:

Value	Description
X'01'	MVS
X'02'	VM
X'03'	PC-DOS
X'04'	VSE
All others	Reserved

SysID Specifies the ID and serial number of the processor on which the object originated

MedID Identifies the storage media that contains the object (for example, the Volume Serial Number on an MVS system)

DSID Identifies the data set on the storage media that contains the object

Structured Fields Using Triplet (X'64'):

- "Begin Overlay (BMO)" on page 124
- "Begin Page Segment (BPS)" on page 137

Application Note: This triplet is also used on the following private font object structured fields in AFP environments:

- Begin Code Page (BCP)
- Begin Font Character Set (BFN)

IMM Insertion Triplet X'73'

The use of this triplet is restricted to the IMM structured field for the following products:

- AFP OnDemand
- AFP Workbench

The IMM Insertion triplet is used to indicate that the Invoke Medium Map (IMM) structured field on which it is specified was inserted at the beginning of a page group by a filtering application. The IMM was inserted between the BNG and the first BPG in the group, but only if an IMM was not already specified there. The purpose of the inserted IMM is to allow the page group to be processed in stand-alone fashion. This triplet is ignored by presentation servers, and the IMM on which it is specified is processed as if the triplet were absent. The presence of this triplet on an IMM may be used by an inverse filtering application to remove the IMM when it is desired to present the complete document as it appeared before the IMM was inserted.

Triplet X'73' Syntax:

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	Tlength	4	Length of the triplet, including Tlength	M	X'02'
1	CODE	Tid	X'73'	Identifies the IMM Insertion triplet	M	X'00'
2–3				Reserved; must be zero	M	X'06'

Triplet X'73' Semantics:

Tlength Contains the length of the triplet.

Tid Identifies the IMM Insertion triplet.

Structured Fields Using Triplet X'73':

- "Invoke Medium Map (IMM)" on page 178

Retired Parameters

The following parameters have been retired:

- MMC Keyword X'0E*nn*'
- MMC Keyword X'F1*nn*'
- MMO Flag Byte Bit 0
- Triplet X'21' Object Type X'02', OCA Function Set X'4000'
- Triplet X'21' Object Type X'05'
- Triplet X'62' StampType X'01'
- OBP RefCSys (Byte 23) = X'05'
- IPO value of X'FFFFFF' for XolOset, YolOset
- IPS value of X'FFFFFF' for XpsOset, YpsOset
- CDD Bytes 0–11
- GRID Font Width value of X'FFFF'
- MGO Mapping Option X'50': Replicate-and-Trim
- IOB RefCSys = X'00'
- Triplet X'22' ResType = X'30'

MMC Keyword X'0Enn'

The use of this keyword is restricted to products that generate and process Form Maps for the 3800 printer.

The maximum horizontal adjustment, in pels, that a 3800 operator can make to position the printing on each form in this subgroup. This modification can occur only in the first repeating group. If X'0E' is not specified, the previous horizontal adjustment value remains in effect.

If more than one MMC contains an adjustment value, the maximum value is specified to the operator. The operator can make an adjustment from 0 to twice the value of this parameter.

At the start of a data stream, this value defaults to 0. Once a value is set, it remains in effect for the entire print job unless it is changed in another subgroup.

The value of *nn* must be from 0 through 20 or X'FF'. X'FF' indicates that the maximum horizontal adjustment is unchanged.

MMC Keyword X'F1nn'

The use of this keyword is restricted to products that generate and process Form Maps for the 3800 printer.

Shows whether forms flash is active. This value is not used by printers that do not support forms flash. This modification can occur only once in the structured field. If this keyword is not present, forms flash is not active.

The value of *nn* can be:

Value	Description
X'00'	Forms flash is not active
X'01'	Forms flash is active

MMO Flag Byte Bit 0

The use of this flag bit is restricted to products that generate and process Form Maps for the 3800 printer.

Bit	Description
0	Raster Indicator

Shows whether the overlay is to be loaded into the printer as a raster pattern overlay or as a coded overlay:

B'0'	Coded overlay
B'1'	Raster overlay

If this bit is B'1' and a raster overlay is already loaded, the overlay is processed as a coded overlay.

Triplet X'21' Object Type X'02', OCA Function Set X'4000'

Use of this parameter value is restricted to the following products:

- PSF/MVS
- PSF/VSE
- PSF/VM
- PSF/400
- AFPU/400

When the object type is set to ObjType = X'02'—Presentation Text, the following Object Function Set value is used:

OCAFnSet	Description
X'4000'	PTOCA PT2

Triplet X'21' Object Type X'05'

Use of this parameter value is restricted to AFP environments.

Value	Description
X'05'	Bar Code

When this object type is specified, the OCAFnSet parameter in the triplet must be set to the following:

OCAFnSet	Description
X'0000'	BCOCA BCD1

Triplet X'62' StampType X'01'

Use of this parameter value is restricted to RMARK.

Value	Description
X'01'	Date and time stamp indicates when the resource object was marked by the RMARK utility program.

OBP RefCSys (Byte 23) = X'05'

Use of this parameter value is restricted to the following products:

- PSF/MVS
- PSF/VSE
- PSF/VM
- PSF/400
- PSF/2
- Infoprint Manager (IPM)

This value is used to specify the current text (I,B) coordinate system as the reference coordinate system. The products that use this value also use three additional bytes in the Object Area Position (OBP) structured field to identify which text coordinate system (absolute I,B or relative I,B) is specified.

IPO value of X'FFFFFF' for XolOset, YolOset

Use of this parameter value is restricted to the following products:

- ACIF
- PSF/MVS
- PSF/VSE
- PSF/VM
- PSF/400
- Infoprint Manager (IPM)

When specified for XolOset or YolOset, this value indicates that the X_p or Y_p value, respectively, of the current text print position should be used for the origin of the overlay.

IPS value of X'FFFFFF' for XpsOset, YpsOset

Use of this parameter value is restricted to the following products:

- ACIF
- PSF/MVS
- PSF/VSE
- PSF/VM
- PSF/400
- Infoprint Manager (IPM)

When specified for XpsOset or YpsOset, this value indicates that the X_p or Y_p value, respectively, of the current text print position should be used for the “origin” of the page segment.

CDD Bytes 0–11

Use of this parameter is restricted to the following products:

- Pre-year 2000 AFP applications

These parameters define the unit base, units per unit base, and extents for the object presentation space:

XocBase (byte 0)
Specifies the unit base for the X axis of the object presentation space coordinate system. The range is X'00', X'01' (10 inches, 10 centimeters).

YocBase (byte 1)
Specifies the unit base for the Y axis of the object presentation space coordinate system. The range is X'00', X'01' (10 inches, 10 centimeters).

XocUnits (bytes 2–3)
Specifies the number of units per unit base for the X axis of the object presentation space coordinate system. The range is 1–32767.

YocUnits (bytes 4–5)
Specifies the number of units per unit base for the Y axis of the object presentation space coordinate system. The range is 1–32767.

XocSize (bytes 6–8)
Specifies the extent of the X axis of the object presentation space coordinate system. This is also known as the object presentation space’s X axis size. The range is 1–32767; a value of X'000000' indicates that the presentation space X axis extent is not specified.

YocSize (bytes 9–11)	Specifies the extent of the Y axis of the object presentation space coordinate system. This is also known as the object presentation space's Y axis size. The range is 1–32767; a value of X'000000' indicates that the presentation space Y axis extent is not specified.

GRID Font Width value of X'FFFF'

Use of this parameter value is restricted to the following products:
- OS/400 print applications

When specified for the GRID font width on an FQN type X'84' triplet, this value indicates that the device default font width should be used.

MGO Mapping Option X'50': Replicate-and-Trim

Use of this parameter is restricted to the following products:
- PSF/390
- PSF/400
- Infoprint Manager for AIX
- Infoprint Manager for Windows

This parameter defines the following mapping option.

The Graphics Presentation Space Window is positioned so that the top left corner of the window is coincident with the origin of the object area and the window size is unchanged. The Graphics Presentation Space Window is then replicated in the X and Y directions of the object area until the object area is filled. Each new replicate of the window in the X direction is precisely aligned with the window previously placed in the X direction. Each new replicate of the window in the Y direction is precisely aligned with the window previously placed in the Y direction. If the last Graphics Presentation Space Window in either the X or Y direction fits only partially into the object area, the portion of the window that falls outside the object area is trimmed. All data that falls within the object area extents is presented, but data that falls outside of the object area is not presented. When this option is specified, the data object's content origin specified in the XocaOset and YocaOset parameters in the Object Area Position structured field is ignored.

IOB RefCSys = X'00'

This parameter value is retired for private use in AFP line-data environments. It is used in AFP line-data environments to position and rotate the object area with respect to the current text (I,B) coordinate system. For more information, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

Triplet X'22' ResType = X'30'

This parameter value is retired for private use in AFP line-data environments. It is used in AFP line-data environments in a PageDef object to denote an IOB Reference. It matches an Include Object (IOB) structured field to a Descriptor. For more information, see *Advanced Function Presentation: Programming Guide and Line Data Reference*.

Coexistence Functions

Coexistence functions are objects, structured fields, triplets, and parameters whose function has been enhanced or superseded by newer functions. In this case, the old and new functions can *coexist*. New generators must generate the new functions. New receivers must process the new functions, but may also continue to process the old functions.

Coexistence Objects

The following objects are coexistence objects:
- AFP page segment
- IM image

AFP Page Segment

The AFP page segment is a coexistence resource object that is being superseded by the MO:DCA page segment. The AFP page segment has the following structure:

```
Begin Page Segment  (BPS, D3A85F)
   +  [  (      D3..FB)     Image Object                        (S)  ]
   +  [  (      D3..7B)     IM Image Object                     (S)  ]
      [  (      D3..BB)     Graphics Object                     (S)  ]
      [  (      D3..9B)     Presentation Text Object                 ]
End Page Segment  (EPS, D3A95F)
```

Figure 105. AFP Page Segment Structure

Positioning of IM Image Objects in an AFP Page Segment: When an IM image object is included in an AFP page segment, it is always positioned relative to the reference point defined in the Include Page Segment (IPS) structured field using the offset, in *image points*, specified in the Image Output Control (IOC) structured field. This offset is resolved using the units of measure specified in the Image Input Descriptor (IID) structured field.

Orientation of Objects in an AFP Page Segment: Unless a Line Data Object Position Migration (X'27') triplet is specified for the AFP page segment or for objects in the page segment, the orientation of the objects in an AFP page segment is always measured with respect to the including page (X_p,Y_p) or overlay (X_{ol},Y_{ol}) coordinate system. For a description of object orientation when the X'27' triplet is specified, see Table 28 on page 510.

Positioning of IO Image and Graphics Objects in an AFP Page Segment: When an IO image object or a graphics object is included in an AFP page segment, it is positioned relative to the page or overlay coordinate system reference point defined in the IPS or relative to the page or overlay coordinate system origin. This is determined by the Reference Coordinate System parameter in the object's OBP structured field. The OBP also specifies the offset with respect to either reference point. This offset is specified in logical units, and if non-zero, must be resolved using the including page or overlay's units of measure. *Because these units of measure are, in general, not known when the page segment is created, using non-zero offsets can lead to unpredictable object positioning and is strongly discouraged*. A MO:DCA page segment or an overlay should be generated to avoid these positioning problems.

Font Mapping for Graphics Objects in an AFP Page Segment: The OEG of a graphics object may not contain any MCF structured fields.

Text Objects in an AFP Page Segment: If an AFP page segment contains text, the following rules apply:

- Text suppressions specified for the including page or overlay also apply to text in the page segment if the suppression local IDs are the same.
- The Absolute Move Baseline (AMB) and Absolute Move Inline (AMI) PTOCA control sequences are processed relative to the origin of the including page or overlay coordinate system.
- The Relative Move Baseline (RMB) and Relative Move Inline (RMI) PTOCA control sequences are processed relative to the reference point defined on the including page or overlay coordinate system by the IPS when these control sequences occur first in the text object.
- Fonts used in the text object must be mapped in the AEG of the including page or overlay. If the text object does not explicitly specify a font using the Set Coded Font Local (SCFL) control sequence, the font that is currently active on the including page or overlay is used. *Because this font is, in general, not known when the page segment is created, including a text object that does not explicitly specify a font can lead to unpredictable text presentation and is strongly discouraged*.
- 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.

Architecture Note: In non-MO:DCA data streams that contain a mixture of structured fields and line data, an IPS offset set to (X'FFFFFF') indicates that the position defined by the current Line Descriptor (LND) is to be used as the reference point for the IPS.

IM Image Object

An IM image data object specifies the contents of a raster image and its placement on a page, overlay, or page segment. An IM image can be either *simple* or *complex*. A simple image is composed of one or more Image Raster Data (IRD) structured fields that define the raster pattern for the entire image. A complex image is divided into regions called *image cells*. Each image cell is composed of one or more IRD structured fields that define the raster pattern for the image cell, and one Image Cell Position (ICP) structured field that defines the position of the image cell relative to the origin of the entire image. Each ICP also specifies the size of the image cell and a fill rectangle into which the cell is replicated. An example of a simple image and a complex image is shown in Figure 106 on page 521.

The IM image object is a valid MO:DCA-P object, but has been superseded by the IOCA image object. This object may appear in MO:DCA-P structures wherever the IOCA image object may appear. New MO:DCA-P generators must generate IO image objects instead of IM image objects. New MO:DCA-P receivers can continue to receive and process IM image objects. The same MO:DCA document can contain both types of objects. This provides upward compatible growth for applications to take advantage of the expanded functions offered by IO Image objects: data compression, image scaling, and resolution-independent output mappings.

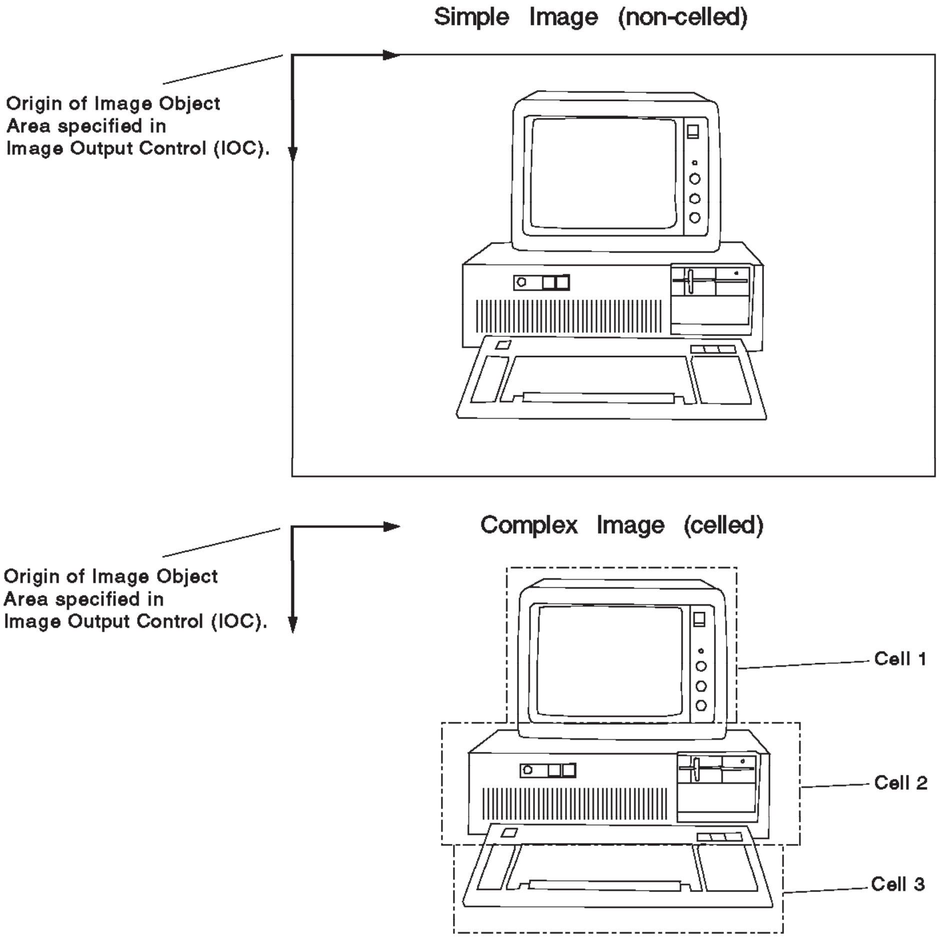

Simple Image (non-celled)
Origin of Image Object Area specified in Image Output Control (IOC).
Complex Image (celled)
Origin of Image Object Area specified in Image Output Control (IOC).
Cell 1
Cell 2
Cell 3

Figure 106. Two Forms of IM Image

In the description of the IM image structured fields that follow, the X-direction, unless otherwise qualified, is the direction in which image points are added to a scan line. The image width is measured in the X-direction. The Y-direction, unless otherwise qualified, is the direction in which scan lines are added to the image. The image height is measured in the Y-direction.

IM Image Object Structure: The structure of an IM image data object is defined as follows using the notation conventions defined in Chapter 4, “MO:DCA-P Objects,” on page 67.

```
Begin IM Image Object  (BII, D3A87B)
        (IOC,  D3A77B)     IM Image Output Control
        (IID,  D3A67B)     IM Image Input Descriptor
        (IRD,  D3EE7B)     IM Image Raster Data                    (S)
End IM Image Object  (EII, D3A97B)
```

Figure 107. IM Image Object Structure: Simple (non-celled) Image

```
Begin IM Image Object  (BII, D3A87B)
        (IOC,  D3A77B)     IM Image Output Control
        (IID,  D3A67B)     IM Image Input Descriptor
        (      D3..7B)     IM Image Cell                         (S)
End IM Image Object  (EII, D3A97B)

IM Image Cell
        (ICP,  D3AC7B)     IM Image Cell Position
        (IRD,  D3EE7B)     IM Image Raster Data                  (S)
```

Figure 108. IM Image Object Structure: Complex (celled) Image

IM Image Structured Fields: The following IM Image structured fields are described under “Coexistence Structured Fields”:
- Begin IM Image Object
- End IM Image Object
- IM Image Cell Position
- IM Image Input Descriptor
- IM Image Output Control
- IM Image Raster Data

Coexistence Structured Fields

The following structured fields are provided in two formats:
- Map Coded Font (MCF)
- Page Position (PGP)
- Presentation Text Descriptor (PTD)

MCF structured fields are called MCF Format 1 and MCF Format 2. PGP structured fields are called PGP Format 1 and PGP Format 2. PTD structured fields are called PTD Format 1 and PTD Format 2. An obsolete name for the PTD Format 1 is Composed Text Descriptor (CTD).

MO:DCA-P receivers may continue to receive and process format-1 structured fields. New MO:DCA-P generators must generate only format-2 versions of these structured fields.

Application Note: The Format 1 version of these structured fields is supported by current AFP data stream applications; but Format 2 is the designated format that is to be used by new AFP applications. PSF servers accept both Format 1 and format 2 structured fields. If both MCF Format 1 and MCF Format 2 structured fields are present in the same environment group, PSF servers require that the MCF Format 1 structured fields precede the MCF Format 2 structured fields.

The following structured fields are described in this section because they are used by a coexistence object, the IM Image object:
- Begin IM Image Object (BII)
- End IM Image Object (EII)
- IM Image Cell Position (ICP)
- IM Image Input Descriptor (IID)
- IM Image Output Control (IOC)
- IM Image Raster Data (IRD)

Map Coded Font (MCF-1) Format 1

The Map Coded Font Format 1 structured field identifies the correspondence between external font names and resource local identifiers.

A font is specified either with the name for a coded font or with a pair of names for the code page and font character set. For a double-byte font, a coded font name is specified, or each coded font section is specified by a code page and font character set pair.

MCF-1 (X'D3B18A') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3B18A'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	UBIN	RGLength	X'1C', X'1E'	Length of each repeating group	M	X'06'
1–3				Reserved; must be zero	M	X'04'
Zero or more repeating groups in the following format:						
0	UBIN	CFLid	X'01'–X'7F', X'FE'	Coded font local ID	M	X'06'
1				Reserved; must be zero	M	X'04'
2	CODE	Sectid	X'00', X'41'–X'FE'	Coded font section ID: **X'00'** Single-byte coded font **X'41'–X'FE'** Double-byte coded font	M	X'04'
3				Reserved; must be zero	M	X'04'
4–11	CHAR	CFName		Coded font name	M	X'04'
12–19	CHAR	CPName		Code page name	M	X'06'
20–27	CHAR	FCSName		Font character set name	M	X'06'
28–29	CODE	CharRot	X'0000', X'2D00', X'5A00', X'8700'	Character rotation for font: **X'0000'** 0 degrees **X'2D00'** 90 degrees **X'5A00'** 180 degrees **X'8700'** 270 degrees	O	X'02'

MCF-1 Semantics:

RGLength Length of each repeating group. Set to 28 if no character rotation is specified; set to 30 if character rotation is specified.

CFLid Coded font local ID. The value must be from 1 to 127. A value of 254 may be used when the MCF-1 structured field is included in the Active Environment Group of a page or overlay for resource management purposes. When a local ID is mapped to a single-byte coded font, or when it is mapped to a double-byte coded font identified with a coded font name, the local ID must be unique across all repeating groups. When a local ID is mapped to a double-byte coded font section, the same local ID must be used to map all sections of the double-byte coded font, and the repeating groups must be contiguous and in ascending order by section number.

Architecture Note: A unique local ID must be mapped for each character rotation of a font.

Sectid
Coded font section ID. For a single-byte coded font, only one section ID can be specified and must be X'00'. For a double-byte coded font that is identified using a coded font name, the sections are specified in the font resource object, and the section ID in the MCF-1 repeating group should be set to X'00'. For a double-byte coded font that is identified using code page and font character set pairs for each section, this value specifies the coded font section number (the first byte of each two-byte code point). The value must be from X'41' to X'FE' for bounded box coded fonts and from X'41' to X'7F' for unbounded box fonts. Each repeating group with the same font local ID must have a unique coded font section ID, and the section ID must be greater than the section ID of the previous repeating group.

CFName
Coded font name. Specifies the name of the coded font. If the name contains a value of X'FFFF' in the first two bytes, it is considered to be a null name, and the coded font must be identified using a code page name and a font character set name. Multiple font local IDs may be mapped to the same coded font name.

CPName
Code page name. Specifies the name of the code page for the single-byte coded font or double-byte coded font section. If the name contains a value of X'FFFF' in the first two bytes, it is considered to be a null name, and the coded font must be identified using a coded font name. In this case, the font character set name must also be specified with a null name. A code page name can appear in multiple repeating groups coupled with the same font character set or with a different font character set.

FCSName
Font character set name. Specifies the name of the font character set for the single-byte coded font or double-byte coded font section. If the name contains a value of X'FFFF' in the first two bytes, it is considered to be a null name, and the coded font must be identified using a coded font name. In this case, the code page name must also be specified with a null name. A font character set name can appear in multiple repeating groups coupled with the same code page or with a different code page.

CharRot
Character rotation (optional). Specifies the character rotation of a font relative to the character baseline. It must be one of the following:

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

Application Note: The character rotation parameter does not exist for 3800 fonts.

Application Note: In AFP environments, the names specified in this structured field must be encoded using the conventions defined in "External Resource Naming Conventions" on page 80.

Page Position (PGP-1) Format 1

The Page Position Format 1 structured field specifies the position of a page's presentation space on the medium presentation space of the physical medium. The page presentation space is oriented so that its X axis, X_{pg} is oriented at zero degrees relative to the X_m axis of the medium presentation space.

PGP-1 (X'D3ACAF') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3ACAF'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–2	UBIN	X_mOset	X'0000'–X'7FFF'	X_m coordinate of page presentation space origin	M	X'06'
3–5	UBIN	Y_mOset	X'0000'–X'7FFF'	Y_m coordinate of page presentation space origin	M	X'06'

PGP-1 Semantics:

X_mOset Offset of the page's presentation space origin along the X_m axis of the medium presentation space using the measurement units specified in the Medium Descriptor structured field.

Y_mOset Offset of the page's presentation space origin along the Y_m axis of the medium presentation space using the measurement units specified in the Medium Descriptor structured field.

Application Note: In AFP environments, the offset range for X_mOset and Y_mOset is 0 to 5461 when the medium coordinate system units of measure are 240 units per inch, and 0 to 32767 when they are 1440 units per inch.

Presentation Text Data Descriptor (PTD-1) Format 1

The Presentation Text Data Descriptor Format 1 structured field specifies the size of a text object presentation space and the measurement units used for the size and for all linear measurements within the text object.

PTD-1 (X'D3A69B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A69B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0	CODE	XptBase	X'00'	Text presentation space unit base for the X axis: **X'00'** 10 inches	M	X'06'
1	CODE	YptBase	X'00'	Text presentation space unit base for the Y axis: **X'00'** 10 inches	M	X'06'
2–3	UBIN	XptUnits	2400, 14400	Text presentation space units per unit base for the X axis	M	X'06'
4–5	UBIN	YptUnits	2400, 14400	Text presentation space units per unit base for the Y axis	M	X'06'
6–7	UBIN	XptSize	X'0001'–X'7FFF'	Text presentation space extent for the X axis	M	X'06'
8–9	UBIN	YptSize	X'0001'–X'7FFF'	Text presentation space extent for the Y axis	M	X'06'
10–11				Reserved; must be binary zero	O	X'00'

PTD-1 Semantics:

XptBase Specifies the unit base for the X axis of the text presentation space.

YptBase Specifies the unit base for the Y axis of the text presentation space.

XptUnits Specifies the number of units per unit base for the X axis of the text presentation space.

YptUnits Specifies the number of units per unit base for the Y axis of the text presentation space.

XptSize Specifies the extent along the X axis of the text presentation space. This must be equal to the extent along the X axis of the including page or overlay presentation space.

YptSize Specifies the extent along the Y axis of the text presentation space. This must be equal to the extent along the Y axis of the including page or overlay presentation space.

Begin IM Image Object (BII)

The Begin IM Image Object structured field begins an IM image data object, which becomes the current data object.

BII (X'D3A87B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A87B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	ImoName		Name of the IM image data object	O	X'02'

BII Semantics:

ImoName Is the name of the IM image data object.

The page, overlay, or resource group containing the Begin IM Image Object structured field must also contain a subsequent matching End IM Image Object structured field, or a X’08’ exception condition exists.

End IM Image Object (EII)

The End IM Image Object structured field terminates the current IM image object initiated by a Begin IM Image Object structured field.

EII (X'D3A97B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A97B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–7	CHAR	ImoName		Name of the IM image data object	O	X'02'

EII Semantics:

ImoName Is the name of the IM image data object being terminated. If a name is specified, it must match the name in the most recent Begin IM Image Object structured field in the containing page, overlay, or resource group or a X’01’ exception condition exists. If the first two bytes of ImoName contain the value X'FFFF', the name matches any name specified on the Begin IM Image Object structured field that initiated the current definition.

A matching Begin IM Image Object structured field must appear at some location preceding the End Image Object structured field, or a X’20’ exception condition exists.

IM Image Cell Position (ICP)

The IM Image Cell Position structured field specifies the placement, size, and replication of IM image cells.

ICP (X'D3AC7B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3AC7B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–1	UBIN	XCOset	X'0000'–X'7FFF'	Offset of image cell in X direction	M	X'06'
2–3	UBIN	YCOset	X'0000'–X'7FFF'	Offset of image cell in Y direction	M	X'06'
4–5	UBIN	XCSize	X'0001'–X'7FFF'	Size of image cell in X direction	M	X'06'
			X'FFFF'	Use default X-extent in IID		
6–7	UBIN	YCSize	X'0001'–X'7FFF'	Size of image cell in Y direction	M	X'06'
			X'FFFF'	Use default Y-extent in IID		
8–9	UBIN	XFilSize	X'0001'–X'7FFF'	Size of fill rectangle in X direction	M	X'06'
			X'FFFF'	Use image cell X-extent		
10–11	UBIN	YFilSize	X'0001'–X'7FFF'	Size of fill rectangle in Y direction	M	X'06'
			X'FFFF'	Use image cell Y-extent		

ICP Semantics:

XCOset Specifies the offset along the X_p direction, in image points, of this image cell from the IM image object area origin.

YCOset Specifies the offset along the Y_p direction, in image points, of this image cell from the IM image object area origin.

XCSize Specifies the extent in the X direction, in image points, of this image cell. A value of X'FFFF' indicates that the default extent specified in bytes 28–29 of the Image Input Descriptor (IID) is to be used.

YCSize Specifies the extent in the Y direction, in image points, of this image cell. A value of X'FFFF' indicates that the default extent specified in bytes 30–31 of the Image Input Descriptor (IID) is to be used.

XFilSize Specifies the extent of the fill rectangle in the X direction, in image points. This value can be smaller than, equal to, or larger than the image cell extent in the X direction (XCSize). A value of X'FFFF' indicates that the image cell X-extent should be used as the fill rectangle X-extent. The fill rectangle is filled in the X direction by repeating the image cell in the X direction. The image cell can be truncated to fit the rectangle.

YFilSize Specifies the extent of the fill rectangle in the Y direction, in image points. This value can be smaller than, equal to, or larger than the image cell extent in the Y direction (YCSize). A value of X'FFFF' indicates that the image cell Y-extent should be used as the fill rectangle Y-extent. The fill rectangle is filled in the Y direction by repeating the image cell in the Y direction. The image cell can be truncated to fit the rectangle.

IM Image Input Descriptor (IID)

The IM Image Input Descriptor structured field contains the descriptor data for an IM image data object. This data specifies the resolution, size, and color of the IM image.

IID (X'D3A67B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A67B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–11	CODE	ConData1		Constant data	M	X'06'
12	CODE	XBase	X'00'	Unit base for the image X axis: **X'00'** 10 inches	M	X'06'
13	CODE	YBase	X'00'	Unit base for the image Y axis: **X'00'** 10 inches	M	X'06'
14–15	UBIN	XUnits	1–32767	Image points per unit base for the image X axis	M	X'06'
16–17	UBIN	YUnits	1–32767	Image points per unit base for the image Y axis	M	X'06'
18–19	UBIN	XSize	X'0001'–X'7FFF'	Size of image in X direction	M	X'06'
20–21	UBIN	YSize	X'0001'–X'7FFF'	Size of image in Y direction	M	X'06'
22–27	CODE	ConData2		Constant data	M	X'06'
28–29	UBIN	XCSizeD	X'0000'–X'7FFF'	Default size of image cell in X direction	M	X'06'
30–31	UBIN	YCSizeD	X'0000'–X'7FFF'	Default size of image cell in Y direction	M	X'06'
32–33	CODE	ConData3		Constant data	M	X'06'
34–35	CODE	Color	See “IID Semantics” for details	Image color	M	X'06'

IID Semantics:

ConData1 Constant data. Must be set to X'0000 0960 0960 0000 0000 0000'.

XBase Specifies the unit base for the X axis of the image.

YBase Specifies the unit base for the Y axis of the image.

XUnits Specifies the number of image points per unit base for the X axis of the image. This value is ten times the resolution of the image in the X direction.

YUnits Specifies the number of image points per unit base for the Y axis of the image. This value is ten times the resolution of the image in the Y direction.

XSize Specifies the extent in the X direction, in image points, of an non-celled (simple) image.

YSize Specifies the extent in the Y direction, in image points, of an non-celled (simple) image.

ConData2 Constant data. Must be set to X'0000 0000 2D00'.

XCSizeD Specifies the default extent in the X direction, in image points, of the image cell. This value is used if the IM Image Cell Position (ICP) structured field does not specify the image cell X extent in bytes 4–5. This value must be set to X'0000' for non-celled images.

YCSizeD Specifies the default extent in the Y direction, in image points, of the image cell. This value is used if the IM Image Cell Position (ICP) structured field does not specify the image cell Y extent in bytes 6–7. This value must be set to X'0000' for non-celled images.

ConData3 Constant data. Must be set to X'0001'.

Color Specifies the color of the image. Syntactically valid values for specifying colors are X'0000' through X'0010' and X'FF00' through X'FF08', which is the range of values defined in the Standard OCA Color Value Table. For a complete description of this table, see “Standard OCA Color Value Table” on page 473. An additional valid value for IM image is X'FFFF'—presentation process default.

Architecture Note: The value X'FFFF' is not a valid color value for IM image in IPDS environments.

IM Image Output Control (IOC)

The IM Image Output Control structured field specifies the position and orientation of the IM image object area and the mapping of the image points to presentation device pels.

IOC (X'D3A77B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3A77B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–2	UBIN	XoaOset	0–32767	X-axis origin of the object area	M	X'06'
3–5	UBIN	YoaOset	0–32767	Y-axis origin of the object area	M	X'06'
6–7	CODE	XoaOrent	X'0000', X'2D00', X'5A00', 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'06'

Offset	Type	Name	Range	Meaning	M/O	Exc
8–9	CODE	YoaOrent	X'0000', X'2D00', X'5A00', 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'06'
Note: See "IOC Semantics" for valid combinations of the XoaOrent and YoaOrent values.						
10–17	CODE	ConData1		Constant data	M	X'06'
18–19	CODE	XMap	X'03E8', X'07D0'	Image mapping in X direction: **X'03E8'** Image point-to-pel **X'07D0'** Image point-to-two pel (double-dot)	M	X'06'
20–21	CODE	YMap	X'03E8', X'07D0'	Image mapping in Y direction: **X'03E8'** Image point-to-pel **X'07D0'** Image point-to-two pel (double-dot)	M	X'06'
22–23	CODE	ConData2		Constant data	M	X'06'

IOC Semantics:

XoaOset Specifies the offset, along the X-axis, of the IM image object area origin to the origin of the including page or overlay coordinate system. If the IM image object is contained in a page segment, specifies the offset, along the X-axis, of the IM image object area origin to the reference point on the including page or overlay coordinate system defined by the Include Page Segment (IPS) structured field. The offset is specified in image points and is resolved using the units of measure specified for the image in the IID structured field.

YoaOset Specifies the offset, along the Y axis, of the IM image object area origin to the origin of the including page or overlay coordinate system. If the IM image object is contained in a page segment, specifies the offset, along the Y-axis, of the IM image object area origin to the reference point on the including page or overlay coordinate system defined by the Include Page Segment (IPS) structured field. The offset is specified in image points and is resolved using the units of measure specified for the image in the IID structured field.

XoaOrent Specifies the amount of clockwise rotation of the IM image object area's X axis about its defined origin relative to the X axis of the page or overlay coordinate system.

YoaOrent Specifies the amount of clockwise rotation of the IM image object area's Y axis about its defined origin relative to the Y 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.

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

Table 29. IOC: Valid Values for XoaOrent and YoaOrent

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

Note: When a complex image is rotated, each cell must be repositioned and rotated.

Application Note: The XoaOrent and YoaOrent values do not affect the placement of image cell origins. Image cell origins can be expressed only in the Xp, Yp coordinate system. When the orientation of a complex (celled) image is changed, the image cell origins must be recalculated so that the appearance of the image is preserved. To simplify the processing of image rotation, it is recommended that the orientation of complex images always be (0, 90).

ConData1 Constant data. Must be set to X'0000 0000 0000 0000'.

XMap Specifies mapping of image points to presentation device pels in the X direction. This value must match the value for YMap.

Value	Description
X'03E8'	Map an image point to a single presentation device pel in the X direction of the IM image object area
X'07D0'	Map an image point to two presentation device pels in the X direction of the IM image object area (double-dot)

YMap Specifies mapping of image points to presentation device pels in the Y direction. This value must match the value for XMap.

Value	Description
X'03E8'	Map an image point to a single presentation device pel in the Y direction of the IM image object area
X'07D0'	Map an image point to two presentation device pels in the Y direction of the IM image object area (double-dot)

Note: If the double-dot function is specified for a complex (celled) image, this function is performed before the cells are used to populate the fill rectangle and before any truncation occurs to fit the cell into the rectangle.

ConData2 Constant data. Must be set to X'FFFF'.

IM Image Raster Data (IRD)

The IM Image Raster Data structured field contains the image points that define the raster pattern for an IM image data object.

IRD (X'D3EE7B') Syntax:

Structured Field Introducer				
SF Length (2B)	ID = **X'D3EE7B'**	Flags (1B)	Reserved X'0000'	Structured Field Data

Offset	Type	Name	Range	Meaning	M/O	Exc
0–*n*	UNDF	IMdata		Up to 32,759 bytes of IM image raster data	O	X'00'

IRD Semantics:

IMdata Contains the image points that define the IM image raster pattern. A *raster pattern* is the array of presentation device pels that forms the image. The image data is uncompressed. Bits are grouped into bytes and are ordered from left to right within each byte. Each bit in the image data represents an image point and is mapped to presentation device pels as specified in the IOC structured field. A bit with value B'1' indicates a significant image point; a bit with value B'0' indicates an insignificant image point.

Image points are recorded from left to right in rows that represents scan lines (X direction), and rows representing scan lines are recorded from top to bottom (Y direction). When the image is presented, all image points in a row are presented before any image points in the next sequential row are presented, and all rows have the same number of image points. If the total number of image points is not a multiple of 8, the last byte of the image data is padded to a byte boundary. The padding bits do not represent image points and are ignored by presentation devices.

Architecture Note: The presentation environment determines how to map significant image points and insignificant image points to presentation device pels. For example, some printers map significant image points to toned pels and insignificant image points to untoned pels.

Coexistence Parameters

The following parameters are coexistence parameters:

- Triplet X'04' mapping option X'41': image point-to-pel
- Triplet X'04' mapping option X'42': image point-to-pel with double dot
- Triplet X'04' mapping option X'50': replicate and trim

Triplet X'04' Mapping Option X'41': Image Point-to-Pel

This mapping is supported for IOCA FS10 for the migration of IM image objects. It provides a mapping for the IOCA FS10 image object similar to the mapping defined for the IM image object. The origin of the IOCA FS10 presentation space is positioned at the origin of the object area. Each image point in the presentation space is mapped to a presentation device pel. Any portion of the image that falls outside the object area is trimmed.

Architecture Note: Resolution correction is not required with this mapping. Therefore, the size of the image presented in the object area is dependent on the pel resolution of the presentation device.

Triplet X'04' Mapping Option X'42': Image Point-to-Pel with Double Dot

This mapping is supported for IOCA FS10 for the migration of IM image objects. It provides a mapping for the IOCA FS10 image object similar to that defined for the IM image object. The origin of the IOCA FS10 presentation space is positioned at the origin of the object area. Each image point in the presentation space is doubled in both directions, resulting in four new image points. The four new image points are then mapped to presentation device pels. Any portion of the image that falls outside the object area is trimmed.

Architecture Note: Resolution correction is not required with this mapping; therefore the size of the image presented in the object area is dependent on the pel resolution of the presentation device.

Triplet X'04' Mapping Option X'50': Replicate and Trim

This mapping is supported for IOCA FS10 for the migration of IM image objects. It provides a function for the IOCA FS10 image object similar to that defined for the celled IM image object. The IOCA FS10 presentation space is positioned in the object area so that its origin is coincident with the origin of the object area and its size is unchanged. The presentation space is then replicated in the X and Y directions of the object area until the object area is filled. Each new replicate of the presentation space in the X direction is precisely aligned with the presentation space previously placed in the X direction. Each new replicate of the presentation space in the Y direction is precisely aligned with the presentation space previously placed in the Y direction. If the last presentation space in either the X or Y direction fits only partially into the object area, the portion of the presentation space that falls outside the object area is trimmed. All data that falls within the object area extents is presented, but data that falls outside of the object area is not presented. When this option is specified, the data object’s content origin specified in the XocaOset and YocaOset parameters in the Object Area Position structured field is ignored.

Appendix D. MO:DCA Registry

This appendix provides a registry for object type identifiers.

Object Type Identifiers

Non-OCA object types supported in MO:DCA document interchange must be identified using ASN.1 Object Identifiers (OIDs) defined in ISO/IEC 8824:1990(E), whose last component identifier is registered in this appendix. Such identifiers are referred to as *object-type OIDs*.

Architecture Note: Object-type OIDs are only assigned to objects that have a clear presentation semantic. Objects can be registered as presentation objects or as non-presentation objects. If an object can be a presentation object and a non-presentation object, a different object-type OID will be assigned to each usage.

The following ISO OID sub-tree is used for the registry:

```
ISO (1)
    Identified Organization (3)
        IBM (18)
            Objects (0)
                Print (4)
                    Document Format (1)
                        MO:DCA (1)
                            Object Type (nnnn)
```

The complete object-type OID is encoded using the Basic Encoding Rules for ASN.1 specified in ISO/IEC 8825:1990(E). The encoding is in the "definite short form" and has the following syntax:

Byte	Description
0	Identifier byte, set to X'06' to indicate an OID encoding
1	Length of content bytes that follow
2–*n*	Content bytes that encode the OID component identifiers

Application Note: The definition of an object-type OID in this registry does not guarantee that the object type identified by the OID is supported in a MO:DCA-P presentation system. To see which object-type OIDs are supported, consult the product documentation. In particular, to see which object-type OIDs are supported by AFP presentation servers, see the *Advanced Function Presentation: Programming Guide and Line Data Reference*.

Registered Object-type OIDs

- *EPS*: Encapsulated Postscript.

  Definition Encapsulated Postscript is defined in Appendix H of the *Postscript Language Reference Manual* (Second Edition, Adobe Systems Incorporated).

  Presentation Space Size Specified by the mandatory “%%BoundingBox” comment in the EPS header.

  Foreground Complete object presentation space

  Background None

  Component ID (13)

  Object-type OID X'06072B12000401010D'

- *TIFF*: Tag Image File Format. This is a raster image format for bi-level, grayscale, and color images. The object contains a single, paginated image, defined by TIFF fields.

  Definition TIFF is defined in *TIFF Revision 6.0* (Aldus Corporation, June 3, 1992).

  Presentation Space Size Specified by the ImageLength (Tag 257), ImageWidth (Tag 256), and ResolutionUnit (Tag 296) TIFF tags.

  Foreground Gray-scale & color: all image points; bi-level: all significant image points

  Background Gray-scale & color: none; bi-level: all insignificant image points

  Component ID (14)

  Object-type OID X'06072B12000401010E'

- *COM Set-up File*: This is a set-up file that contains information used to present MO:DCA data on microfiche media with Anacomp devices.

  Definition Anacomp COM Set-up files are defined in *XFP2000 Reference* (XF-07-9201 [Device Recorder Software], Anacomp Inc., July 15, 1992).

  Presentation Space Size N/A; this is not a page-level presentation object

  Foreground N/A; this is not a page-level presentation object

  Background N/A; this is not a page-level presentation object

  Component ID (15)

  Object-type OID X'06072B12000401010F'

- *Tape Label Set-up File*: This is a set-up file that contains information used to present MO:DCA documents that exists in tape libraries on microfiche media.

  Definition Tape Label Set-up files are defined in *MVS/DFP™ V3.3: Using Magnetic Tape Labels and File Structure*, SC26-4565.

  Presentation Space Size N/A; this is not a page-level presentation object

  Foreground N/A; this is not a page-level presentation object

  Background N/A; this is not a page-level presentation object

Component ID (16)

Object-type OID X'06072B120004010110'

- *Device Independent Bit Map (DIB), Windows Version*: This is an image file format used by Microsoft Windows Version 3.0 and higher for bi-level and color images.

  Definition This image file format is defined in *Microsoft Windows Software Development Kit: Reference Volume 2, Version 3.0* (Microsoft Corporation, 1990).

  Presentation Space Size Specified by the biWidth and biHeight parameters in the BITMAPINFOHEADER structure.

  Foreground Gray-scale & color: all image points; bi-level: all significant image points

  Background Gray-scale and color: none; bi-level: all insignificant image points

  Component ID (17)

  Object-type OID X'06072B120004010111'

- *Device Independent Bit Map (DIB), OS/2 PM Version*: This is an image file format used by OS/2 PM Version 1.1 and 1.2 for bi-level and color images.

  Definition This image file format is defined in *Microsoft Windows Software Development Kit: Reference Volume 2, Version 3.0* (Microsoft Corporation, 1990).

  Presentation Space Size Specified by the bcWidth and bcHeight parameters in the BITMAPCOREHEADER structure.

  Foreground Gray-scale & color: all image points; bi-level: all significant image points

  Background Gray-scale & color: none; bi-level: all insignificant image points

  Component ID (18)

  Object-type OID X'06072B120004010112'

- *Paintbrush Picture File Format (PCX)*: This is an image file format for bi-level and color images.

  Definition This image file format is defined in *Technical Documentation for PC Paintbrush & Frieze Graphics* (Z Soft Corporation, 1985).

  Presentation Space Size Header bytes 4–11 define the x,y coordinates of the upper-left and lower-right corners of the image, in pixels. The x-difference + 1 is the width of the image, the y-difference + 1 is the height of the image.

  Foreground Gray-scale and color: all image points; bi-level: all significant image points

  Background Gray-scale & color: none; bi-level: all insignificant image points

Component ID (19)

Object-type OID X'06072B120004010113'

- *Color Mapping Table (CMT)*: This is a set-up file that provides mappings for color values specified in one or more documents.

  Definition The Color Mapping Table is defined in the *Mixed Object Document Content Architecture Reference*.

  Presentation Space Size N/A; this is not a page-level presentation object

  Foreground N/A; this is not a page-level presentation object

  Background N/A; this is not a page-level presentation object

  Component ID (20)

  Object-type OID X'06072B120004010114'

- *Graphics Interchange Format (GIF)*: This is an image file format for bi-level and color images.

  Definition This image file format is defined in *Graphics Interchange Format, Version 89a Programming Reference* (CompuServe Incorporated, July 31, 1990).

  Presentation Space Size The width and height of the image, in pixels, is specified in the Image Descriptor.

  Foreground All image points

  Background None

  Component ID (22)

  Object-type OID X'06072B120004010116'

- *JPEG File Interchange Format (JFIF)*: This is an image file format for grayscale and color images.

  Definition This image file format is defined in Eric Hamilton, *JPEG File Interchange Format, Version 1.02* (C-Cube Microsystems, Inc., September 31, 1990).

  Presentation Space Size The number of rows and number of columns for the Y, C_b, and C_r components of the image are specified in the Start of Frame (SOF0) segment.

  Foreground All image points

  Background None

  Component ID (23)

  Object-type OID X'06072B120004010117'

- *Anacomp AnaStak Control Record*: This is a set-up file that contains accounting and control information to present MO:DCA documents on microfiche media using Anacomp devices via tape or data transmission.

  Definition The Anacomp AnaStak Control Record is defined in *AnaStak, The Anacomp Report-Stacking System: User's Guide and Reference* (Anast203, Anacomp Inc.).

  Presentation Space Size N/A; this is not a page-level presentation object

Foreground N/A; this is not a page-level presentation object

Background N/A; this is not a page-level presentation object

Component ID (24)

Object-type OID X'06072B120004010118'

- *Portable Document Format (PDF) Single-page Object*: This is a presentation object consisting of a PDF file that defines a single page containing text, graphics, and image using PDF operators.

  Definition The PDF file format is defined in the *Portable Document Format Reference Manual* (Adobe Systems Incorporated, 1993).

  Presentation Space Size The (x,y) coordinates of the lower-left corner and upper-right corner are specified by the required MediaBox key in the Page Object dictionary.

  Foreground Complete object presentation space

  Background None

  Component ID (25)

  Object-type OID X'06072B120004010119'

- *Portable Document Format (PDF) Resource Object*: This is a resource object that may be referenced by a PDF single-page object. Examples of PDF resource objects are fonts, font descriptors, and raster images.

  Definition The PDF file format is defined in the *Portable Document Format Reference Manual* (Adobe Systems Incorporated, 1993).

  Presentation Space Size N/A; this is not a page-level presentation object

  Foreground N/A; this is not a page-level presentation object

  Background N/A; this is not a page-level presentation object

  Component ID (26)

  Object-type OID X'06072B12000401011A'

- *PCL Page Object*: This is a paginated presentation object that is specified using the PCL language.

  Definition The PCL printer language is defined in the *PCL 5 Printer Language Technical Reference Manual* (Hewlett Packard Company).

  Presentation Space Size Specified by the E_c&l#A command.

  Foreground Complete object presentation space

  Background None

  Component ID (34)

  Object-type OID X'06072B120004010122'

- *Resident Color Profile Resource Object*: This is a device-resident resource object that defines how device-dependent colors in a data object are related to device-independent colors.

  Definition Resident Color Profile objects are device-dependent and are defined by the presentation device.

Presentation Space Size N/A; this is not a page-level presentation object

Foreground N/A; this is not a page-level presentation object

Background N/A; this is not a page-level presentation object

Component ID (46)

Object-type OID X'06072B12000401012E'

Implementation Note: If a presentation object references a color profile resource object and this resource is not supported by the presentation device, AFP print servers will issue a warning message and allow presentation to proceed without the color profile.

- *IOCA Tile Resource*: This is an IOCA FS45 tile resource.

  Definition The IOCA FS45 resource tile is defined in *Image Object Content Architecture Reference*.

  Presentation Space Size N/A; this is not a page-level presentation object

  Foreground N/A; this is not a page-level presentation object

  Background N/A; this is not a page-level presentation object

  Component ID (47)

  Object-type OID X'06072B12000401012F'

- *Encapsulated PostScript (EPS) Object with Transparency*:

  Definition Encapsulated Postscript is defined in Appendix H of the *Postscript Language Reference Manual* (Second Edition, Adobe Systems Incorporated).

  Presentation Space Size Specified by the mandatory "%%BoundingBox" comment in the EPS header.

  Foreground The painted portions of the object presentation space

  Background The unpainted portions of the object presentation space

  Component ID (48)

  Object-type OID X'06072B120004010130'

- *Portable Document Format (PDF) Single-page Object with Transparency*: This is a presentation object consisting of a PDF file that defines a single page containing text, graphics, and image using PDF operators.

  Definition The PDF file format is defined in the *Portable Document Format Reference Manual* (Adobe Systems Incorporated, 1993).

  Presentation Space Size The (x,y) coordinates of the lower-left corner and upper-right corner are specified by the required MediaBox key in the Page Object dictionary.

  Foreground The painted portions of the object presentation space

  Background The unpainted portions of the object presentation space

  Component ID (49)

Object-type OID X'06072B120004010131'

- *TrueType/OpenType Font Resource Object*: This is a font resource object that may be referenced by a data object.

Definition The TrueType Font format is defined in the *TrueType Reference Manual* (Apple Computer, Inc., 1999). It is a subset of the OpenType Font Format, which is defined in the *OpenType Specification* (Microsoft Corporation and Adobe Systems Incorporated, 2000).

Presentation Space Size N/A; this is not a page-level presentation object

Foreground N/A; this is not a page-level presentation object

Background N/A; this is not a page-level presentation object

Component ID (51)

Object-type OID X'06072B120004010133'

- *TrueType/OpenType Collection Resource Object*: This is a collection of TrueType/OpenType font resources. It is identified with a *TTC* file extension in Windows environments.

Definition The TrueType Font collection format is defined in the *TrueType Reference Manual* (Apple Computer, Inc., 1999). It is a subset of the OpenType Font Format, which is defined in the *OpenType Specification* (Microsoft Corporation and Adobe Systems Incorporated, 2000).

Presentation Space Size N/A; this is not a page-level presentation object

Foreground N/A; this is not a page-level presentation object

Background N/A; this is not a page-level presentation object

Component identifier (53)

Object-type OID X'06072B120004010135'

- *Resource Access Table (RAT)*: This is a set-up file that provides information used to access and process resources that are referenced in MO:DCA documents.

Definition The Resource Access Table is defined in the *Mixed Object Document Content Architecture Reference*.

Presentation Space Size N/A; this is not a page-level presentation object

Foreground N/A; this is not a page-level presentation object

Background N/A; this is not a page-level presentation object

Component ID (54)

Object-type OID X'06072B120004010136'

Object Type Summary

Table 30 lists the object types registered in the MO:DCA architecture along with their component identifier and their encoded object-type OID.

Table 30. Registered Object Types Sorted by Component ID

Component ID	Object Type	Encoded Object-type OID
13	EPS	X'06072B12000401010D'
14	TIFF	X'06072B12000401010E'
15	COM Set-up	X'06072B12000401010F'
16	Tape Label Set-up	X'06072B120004010110'
17	DIB, Windows Version	X'06072B120004010111'
18	DIB, OS/2 PM Version	X'06072B120004010112'
19	PCX	X'06072B120004010113'
20	Color Mapping Table (CMT)	X'06072B120004010114'
22	GIF	X'06072B120004010116'
23	JFIF	X'06072B120004010117'
24	AnaStak Control Record	X'06072B120004010118'
25	PDF Single-page Object	X'06072B120004010119'
26	PDF Resource Object	X'06072B12000401011A'
34	PCL Page Object	X'06072B120004010122'
46	Resident Color Profile	X'06072B12000401012E'
47	IOCA FS45 Tile Resource	X'06072B12000401012F'
48	EPS with Transparency	X'06072B120004010130'
49	PDF with Transparency	X'06072B120004010131'
51	TrueType/OpenType Font	X'06072B120004010133'
53	TrueType/OpenType Font Collection	X'06072B120004010135'
54	Resource Access Table	X'06072B120004010136'

Non-OCA Object Types Supported by the IOB Structured Field

Table 31 lists the object types that can be included for presentation by the Include Object (IOB) structured field with ObjType = X'92'—Other object data. All object types in this table are not supported by all presentation systems. To see which object-type OIDs are supported, consult the product documentation. In particular, to see which object-type OIDs are supported by AFP presentation servers, see the *Advanced Function Presentation: Programming Guide and Line Data Reference*.

Table 31. Non-OCA Object Types Supported by the IOB

Component ID	Object Type	Encoded Object-type OID
13	EPS	X'06072B12000401010D'
14	TIFF	X'06072B12000401010E'
17	DIB, Windows Version	X'06072B120004010111'
18	DIB, OS/2 PM Version	X'06072B120004010112'
19	PCX	X'06072B120004010113'
22	GIF	X'06072B120004010116'
23	JFIF	X'06072B120004010117'
25	PDF Single-page Object	X'06072B120004010119'
34	PCL Page Object	X'06072B120004010122'
48	EPS with Transparency	X'06072B120004010130'
49	PDF with Transparency	X'06072B120004010131'

Data Objects and Supported Secondary Resources

Table 32 lists the secondary resources that are supported by various data objects.

Table 32. Data Objects and Secondary Resources

Data Object	Secondary Resource	Internal Resource Identifier
IOCA Image	IOCA Tile Resource	4-byte local ID
Encapsulated PostScript (EPS) (with or without transparency)	Resident Color Profile	None
PDF Single-Page Object (with or without transparency)	Resident Color Profile	None
PDF Single-Page Object (with or without transparency)	PDF Resource Object	Identifier with syntax defined by PDF
PTOCA Text; see Note	TrueType/OpenType Font	1-byte local ID
AFP GOCA Text; see Note	TrueType/OpenType Font	1-byte local ID
BCOCA Text; see Note	TrueType/OpenType Font	1-byte local ID
Note: These table entries are not formally primary resource/ secondary resource pairs since PTOCA, AFP GOCA, and BCOCA objects currently cannot be processed as resource objects. However, the TrueType/OpenType fonts for these objects are processed like other secondary resources.		

Media Type Identifiers

Media types supported in MO:DCA document interchange may be identified using ASN.1 Object Identifiers (OIDs) defined in ISO/IEC 8824:1990(E), whose last component identifier is registered in this appendix. Such identifiers are referred to as *media-type OIDs*.

The following ISO OID sub-tree is used for the registry:

ISO(1)
  Identified Organization (3)
    IBM (18)
      Objects (0)
        Print (4)
          Print Attributes (3)
            Media Types (1)
              Media (*nnnn*)

Architecture Note: The Document Printing Application (DPA) ISO/IEC DIS 10175:1991 draft standard has also registered media types with OIDs using a DPA ISO OID sub-tree. Wherever media types in the MO:DCA registry are also registered in the DPA registry, the last leaf in the MO:DCA OID, also called the MO:DCA media type component ID, has been chosen to match the last leaf in the DPA OID.

The complete media-type OID is encoded using the Basic Encoding Rules for ASN.1 specified in ISO/IEC 8825:1990(E). The encoding is in the “definite short form” and has the following syntax:

Byte	**Description**
0	Identifier byte, set to X'06' to indicate an OID encoding
1	Length of content bytes that follow
2–*n*	Content bytes that encode the OID component identifiers

Media Type Summary

Table 33 and Table 34 on page 547 list the media types registered in the MO:DCA architecture along with their component identifier and their encoded media-type OID.

Table 33. Registered Media Types Sorted by Component ID

Component ID	Media Name	Media Type	Encoded Media-type OID
0	ISO A4	ISO A4 white (210 × 297 mm)	X'06072B120004030100'
1	ISO A4 CO	ISO A4 colored	X'06072B120004030101'
2	ISO A4 TR	ISO A4 transparent	X'06072B120004030102'
5	ISO A4 THD	ISO 1/3 A4	X'06072B120004030105'
7	ISO A4 TAB	ISO A4 tab (225 × 297 mm)	X'06072B120004030107'
10	ISO A3	ISO A3 white (297 × 420 mm)	X'06072B12000403010A'
11	ISO A3 CO	ISO A3 colored	X'06072B12000403010B'
20	ISO A5	ISO A5 white (148.5 × 210 mm)	X'06072B120004030114'
21	ISO A5 CO	ISO A5 colored	X'06072B120004030115'
30	ISO B4	ISO B4 white (257 × 364 mm)	X'06072B12000403011E'
31	ISO B4 CO	ISO B4 colored	X'06072B12000403011F'
40	ISO B5	ISO B5 white (176 × 250 mm)	X'06072B120004030128'
41	ISO B5 CO	ISO B5 colored	X'06072B120004030129'
42	JIS B4	JIS B4 (257 × 364 mm)	X'06072B12000403012A'
43	JIS B5	JIS B5 (182 × 257 mm)	X'06072B12000403012B'
50	LETTER	North American letter white (8.5 × 11 in.)	X'06072B120004030132'
51	LETTER CO	North American letter colored	X'06072B120004030133'
52	LETTER TR	North American letter transparent	X'06072B120004030134'
60	LEGAL	North American legal white (8.5 × 14 in.)	X'06072B12000403013C'
61	LEGAL CO	North American legal colored	X'06072B12000403013D'
63	LEGAL 13	North American legal 13 (Folio) (8.5 × 13 in.)	X'06072B12000403013F'
65	EXEC	North American executive (7.25 × 10.5 in.)	X'06072B120004030141'
67	LEDGER	North American ledger (11 × 17 in.)	X'06072B120004030143'
69	STATEMNT	North American statement (5.5 × 8.5 in.)	X'06072B120004030145'
73	ISO B5 ENV	ISO B5 envelope (176 × 250 mm)	X'06072B120004030149'
75	COM 10 ENV	Com10 envelope (9.5 × 4.125 in.)	X'06072B12000403014B'
76	MON ENV	Monarch envelope (7.5 × 3.875 in.)	X'06072B12000403014C'
77	DL ENV	DL envelope (220 × 110 mm)	X'06072B12000403014D'
79	C5 ENV	C5 envelope (229 × 162 mm)	X'06072B12000403014F'
80	JP PC ENV	Japan postcard envelope (200 × 150 mm)	X'06072B120004030150'
81	JP PC	Japan postcard (Hagaki) (100 × 148 mm)	X'06072B120004030151'

Table 33. Registered Media Types Sorted by Component ID (continued)

Component ID	Media Name	Media Type	Encoded Media-type OID
83	ISO B4 ENV	ISO B4 envelope (250 × 353 mm)	X'06072B120004030153'
93	ISO C4 ENV	ISO C4 envelope (229 × 324 mm)	X'06072B12000403015D'
103	ISO C5 ENV	ISO C5 envelope (162 × 229 mm)	X'06072B120004030167'
113	ISO LNG ENV	ISO long envelope	X'06072B120004030171'
123	10×13 ENV	North American 10×13 envelope	X'06072B12000403017B'
133	9×12 ENV	North American 9×12 envelope	X'06082B12000403018105'
143	BSNS ENV	North American business envelope	X'06082B1200040301810F'
145	LETTER TAB	Letter tab (9 × 11 in.)	X'06082B12000403018111'
146	LEGAL TAB	Legal tab (9 × 14 in.)	X'06082B12000403018112'
147	9×12 MAN	Manual (9 × 12 in.)	X'06082B12000403018113'
148	8×10.5 MED	Media (8 × 10.5 in.)	X'06082B12000403018114'
149	9×14 MED	Media (9 × 14 in.)	X'06082B12000403018115'
150	INDEX CD	Index Card	X'06082B12000403018116'
151	US PC	US Postcard	X'06082B12000403018117'
152	ISO A6 PC	ISO A6 Postcard (105 × 148 mm)	X'06082B12000403018118'
153	RA3	Oversize A3 (16.923 × 12.007 in.)	X'06082B12000403018119'
154	14×17 MED	Media (14 × 17 in.)	X'06082B1200040301811A'
155	12×18 MED	Media (12 × 18 in.)	X'06082B1200040301811B'
156	14×18 MED	Media (14 × 18 in.)	X'06082B1200040301811C'
157	8.5×10 MED	Media (8.5 × 10 in.)	X'06082B1200040301811D'
160	8×10 MED	Media (8 × 10 in.)	X'06082B12000403018120'
162	RA4	Oversize A4 (8.465 × 12.007 in.)	X'06082B12000403018122'
163	8×13 MED	Media (8 × 13 in)	X'06082B12000403018123'
164	8.25×13 MED	Media (8.25 × 13 in)	X'06082B12000403018124'
165	8.25×14 MED	Media (8.25 × 14 in)	X'06082B12000403018125'
166	8.5×12.4 MED	Media (8.5 × 12.4 in)	X'06082B12000403018126'
167	10×14 MED	Media (10 × 14 in)	X'06082B12000403018127'
168	10×15 MED	Media (10 × 15 in)	X'06082B12000403018128'
169	11×14 MED	Media (11 × 14 in)	X'06082B12000403018129'
170	11×15 MED	Media (11 × 15 in)	X'06082B1200040301812A'
171	ISO B6	ISO B6 (128 × 182 mm)	X'06082B1200040301812B'
172	REP PD PC	Reply-paid PC (148 × 200 mm)	X'06082B1200040301812C'
173	170×210 MED	Media (170 × 210 in)	X'06082B1200040301812D'
174	182×210 MED	Media (182 × 210 in)	X'06082B1200040301812E'
175	210×340 MED	Media (210 × 340 in)	X'06082B1200040301812F'
176	8KAI	8KAI Media (267 × 290 mm)	X'06082B12000403018130'
177	16KAI	16KAI Media (195 × 267 mm)	X'06082B12000403018131'

Table 34. Registered Media Types Sorted by Media Names

Media Name	Media Type	Component ID	Encoded Media-type OID
BSNS ENV	North American business envelope	143	X'06082B1200040301810F'
COM 10 ENV	Com10 envelope (9.5 × 4.125 in.)	75	X'06072B12000403014B'
C5 ENV	C5 envelope (229 × 162 mm)	79	X'06072B12000403014F'
DL ENV	DL envelope (220 × 110 mm)	77	X'06072B12000403014D'
EXEC	North American executive (7.25 × 10.5 in.)	65	X'06072B120004030141'
INDEX CD	Index Card	150	X'06082B12000403018116'
ISO A4	ISO A4 white (210 × 297 mm)	0	X'06072B120004030100'
ISO A4 CO	ISO A4 colored	1	X'06072B120004030101'
ISO A4 TAB	ISO A4 tab (225 × 297 mm)	7	X'06072B120004030107'
ISO A4 THD	ISO 1/3 A4	5	X'06072B120004030105'
ISO A4 TR	ISO A4 Transparent	2	X'06072B120004030102'
ISO A3	ISO A3 white (297 × 420 mm)	10	X'06072B12000403010A'
ISO A3 CO	ISO A3 colored	11	X'06072B12000403010B'
ISO A5	ISO A5 white (148.5 × 210 mm)	20	X'06072B120004030114'
ISO A5 CO	ISO A5 colored	21	X'06072B120004030115'
ISO A6 PC	ISO A6 Postcard (105 × 148 mm)	152	X'06082B12000403018118'
ISO B4	ISO B4 white (257 × 364 mm)	30	X'06072B12000403011E'
ISO B4 CO	ISO B4 colored	31	X'06072B12000403011F'
ISO B5	ISO B5 white (176 × 250 mm)	40	X'06072B120004030128'
ISO B5 CO	ISO B5 colored	41	X'06072B120004030129'
ISO B4 ENV	ISO B4 envelope (250 × 353 mm)	83	X'06072B120004030153'
ISO B5 ENV	ISO B5 envelope (176 × 250 mm)	73	X'06072B120004030149'
ISO B6	ISO B6 (128 × 182 mm)	171	X'06082B1200040301812B'
ISO C4 ENV	ISO C4 envelope (229 × 324 mm)	93	X'06072B12000403015D'
ISO C5 ENV	ISO C5 envelope (162 × 229 mm)	103	X'06072B120004030167'
ISO LNG ENV	ISO long envelope	113	X'06072B120004030171'
JIS B4	JIS B4 (257 × 364 mm)	42	X'06072B12000403012A'
JIS B5	JIS B5 (182 × 257 mm)	43	X'06072B12000403012B'
JP PC	Japan postcard (Hagaki) (100 × 148 mm)	81	X'06072B120004030151'
JP PC ENV	Japan postcard envelope (200 × 150 mm)	80	X'06072B120004030150'
LEDGER	North American ledger (11 × 17 in.)	67	X'06072B120004030143'
LEGAL	North American legal white (8.5 × 14 in.)	60	X'06072B12000403013C'
LEGAL CO	North American legal colored	61	X'06072B12000403013D'
LEGAL TAB	Legal tab (9 × 14 in.)	146	X'06082B12000403018112'
LEGAL 13	North American legal 13 (Folio) (8.5 × 13 in.)	63	X'06072B12000403013F'
LETTER	North American letter white (8.5 × 11 in.)	50	X'06072B120004030132'

Table 34. Registered Media Types Sorted by Media Names (continued)

Media Name	Media Type	Component ID	Encoded Media-type OID
LETTER CO	North American letter colored	51	X'06072B120004030133'
LETTER TAB	Letter tab (9 × 11 in.)	145	X'06082B12000403018111'
LETTER TR	North American letter transparent	52	X'06072B120004030134'
MON ENV	Monarch envelope (7.5 × 3.875 in.)	76	X'06072B12000403014C'
RA3	Oversize A3 (16.923 × 12.007 in.)	153	X'06082B12000403018119'
RA4	Oversize A4 (8.465 × 12.007 in.)	162	X'06082B12000403018122'
REP PD PC	Reply-paid PC (148 × 200 mm)	172	X'06082B1200040301812C'
STATEMNT	North American statement (5.5 × 8.5 in.)	69	X'06072B120004030145'
US PC	US Postcard	151	X'06082B12000403018117'
8×10 MED	Media (8 × 10 in.)	160	X'06082B12000403018120'
8×10.5 MED	Media (8 × 10.5 in.)	148	X'06082B12000403018114'
8×13 MED	Media (8 × 13 in.)	163	X'06082B12000403018123'
8.25×13 MED	Media (8.25 × 13 in.)	164	X'06082B12000403018124'
8.25×14 MED	Media (8.25 × 14 in.)	165	X'06082B12000403018125'
8.5×10 MED	Media (8.5 × 10 in.)	157	X'06082B1200040301811D'
8.5×12.4 MED	Media (8.5 × 12.4 in.)	166	X'06082B12000403018126'
9×12 ENV	North American 9×12 envelope	133	X'06082B12000403018105'
9×12 MAN	Manual (9 × 12 in.)	147	X'06082B12000403018113'
9×14 MED	Media (9 × 14 in.)	149	X'06082B12000403018115'
10×13 ENV	North American 10×13 envelope	123	X'06072B12000403017B'
10×14 MED	Media (10 × 14 in)	167	X'06082B12000403018127'
10×15 MED	Media (10 × 15 in)	168	X'06082B12000403018128'
11×14 MED	Media (11 × 14 in)	169	X'06082B12000403018129'
11×15 MED	Media (11 × 15 in)	170	X'06082B1200040301812A'
12×18 MED	Media (12 × 18 in.)	155	X'06082B1200040301811B'
14×17 MED	Media (14 × 17 in.)	154	X'06082B1200040301811A'
14×18 MED	Media (14 × 18 in.)	156	X'06082B1200040301811C'
170×210 MED	Media (170 × 210 mm)	173	X'06082B1200040301812D'
182×210 MED	Media (182 × 210 mm)	174	X'06082B1200040301812E'
210×340 MED	Media (210 × 340 mm)	175	X'06082B1200040301812F'
8KAI	8KAI Media (267 × 290 mm)	176	X'06082B12000403018130'
16KAI	16KAI Media (195 × 267 mm)	177	X'06082B12000403018131'

Architecture Notes:

1. A total of 2^7 = 128 media types can be registered using one byte to encode the component ID, as, for example, in the encoding for component IDs 0–123. A total of 2^{14} = 16,384 media types can be registered using two bytes to encode the component ID, as, for example, in the encoding for component IDs 133 and 143. A total of 2^{21} = 2,097,152 media types can be registered using three bytes to encode the component ID. A total of 2^{28} = 268,435,456 media types can be

registered using four bytes to encode the component ID. This registry will support a maximum of 4 bytes for the encoding of the component ID.

2. The range from media type OID X'06082B1200040301E000' (component ID 12,288) to X'060A2B1200040301FFFFFF7F' (component ID 268,435,455) is reserved for user-defined media types.

Color Profile Identifiers

Color profiles supported in MO:DCA document interchange may be identified using ASN.1 Object Identifiers (OIDs) defined in ISO/IEC 8824:1990(E), whose last component identifier is registered in this appendix. Such identifiers are referred to as *object OIDs*.

The following ISO OID sub-tree is used for the registry:

ISO (1)
  Identified Organization (3)
    IBM (18)
      Objects (0)
        Print (4)
          Print Attributes (3)
            Color Profiles (3)
              Profiles (*nnnn*)

The complete OID is encoded using the Basic Encoding Rules for ASN.1 specified in ISO/IEC 8825:1990(E). The encoding is in the “definite short form” and has the following syntax:

Byte	**Description**
0	Identifier byte, set to X'06' to indicate an OID encoding
1	Length of content bytes that follow
2–*n*	Content bytes that encode the OID component identifiers

Color Profile Summary

Table 35 lists the color profiles registered in the MO:DCA architecture along with their component identifier and their object OID.

Table 35. Color Profile Registry

Component ID	Profile Name	Object OID
0	CMYK SWOP	X'06072B120004030300'
1	CMYK Euroscale	X'06072B120004030301'

Architecture Notes:

1. A total of 2^7 = 128 color profiles can be registered using one byte to encode the component ID. A total of 2^{28} = 268,435,456 color profiles can be registered using four bytes to encode the component ID. This registry will support a maximum of 4 bytes for the encoding of the component ID.
2. Many PostScript level 1 files contain color specified in the CMYK color space but tuned to one of a number of offset press standards that are geography-based. Two such standards are CMYK SWOP (US), and CMYK Euroscale (Europe). The standards essentially define the color rendering of hypothetical presses. For example, a specific color $C_1M_1Y_1K_1$ defined as SWOP CMYK has a specific colorimetric representation that is normally defined by a color swatch. The CMYK SWOP and CMYK Euroscale color profiles are supported in AFP environments for EPS objects and PDF single-page objects.

Appendix E. Cross-References

This appendix provides tables that list:

- MO:DCA structured fields sorted by identifier
- MO:DCA structured fields sorted by acronym
- MO:DCA triplets sorted by identifier
- MO:DCA triplets sorted by name

MO:DCA Structured Fields Sorted by Identifier

Table 36. Structured Fields Sorted by ID

Identifier	Acronym	Structured Field Name	Page
X'D3A088'	MFC	Medium Finishing Control	236
X'D3A090'	TLE	Tag Logical Element	310
X'D3A288'	MCC	Medium Copy Count	209
X'D3A66B'	OBD	Object Area Descriptor	270
X'D3A67B'	IID	IM Image Input Descriptor (**C**)	529
X'D3A688'	MDD	Medium Descriptor	220
X'D3A692'	CDD	Container Data Descriptor	150
X'D3A69B'	PTD-1	Presentation Text Descriptor Format-1 (**C**)	526
X'D3A6AF'	PGD	Page Descriptor	279
X'D3A6BB'	GDD	Graphics Data Descriptor	174
X'D3A6C5'	FGD	Form Environment Group Descriptor (**O**)	503
X'D3A6EB'	BDD	Bar Code Data Descriptor	110
X'D3A6FB'	IDD	Image Data Descriptor	175
X'D3A77B'	IOC	IM Image Output Control (**C**)	530
X'D3A788'	MMC	Medium Modification Control	248
X'D3A79B'	CTC	Composed Text Control (**O**)	502
X'D3A7AF'	PMC	Page Modification Control	297
X'D3A85F'	BPS	Begin Page Segment	137
X'D3A877'	BCA	Begin Color Attribute Table	107
X'D3A87B'	BII	Begin IM Image (**C**)	526
X'D3A892'	BOC	Begin Object Container	129
X'D3A89B'	BPT	Begin Presentation Text Object	139
X'D3A8A7'	BDI	Begin Document Index	112
X'D3A8A8'	BDT	Begin Document	114
X'D3A8AD'	BNG	Begin Named Page Group	126
X'D3A8AF'	BPG	Begin Page	134
X'D3A8BB'	BGR	Begin Graphics Object	118
X'D3A8C4'	BDG	Begin Document Environment Group	111
X'D3A8C5'	BFG	Begin Form Environment Group (**O**)	502
X'D3A8C6'	BRG	Begin Resource Group	141

Table 36. Structured Fields Sorted by ID (continued)

Identifier	Acronym	Structured Field Name	Page
X'D3A8C7'	BOG	Begin Object Environment Group	133
X'D3A8C9'	BAG	Begin Active Environment Group	104
X'D3A8CC'	BMM	Begin Medium Map	122
X'D3A8CD'	BFM	Begin Form Map	116
X'D3A8CE'	BRS	Begin Resource	143
X'D3A8D9'	BSG	Begin Resource Environment Group	148
X'D3A8DF'	BMO	Begin Overlay	124
X'D3A8EB'	BBC	Begin Bar Code Object	105
X'D3A8FB'	BIM	Begin Image Object	120
X'D3A95F'	EPS	End Page Segment	167
X'D3A977'	ECA	End Color Attribute Table	153
X'D3A97B'	EII	End IM Image (**C**)	527
X'D3A992'	EOC	End Object Container	164
X'D3A99B'	EPT	End Presentation Text Object	168
X'D3A9A7'	EDI	End Document Index	156
X'D3A9A8'	EDT	End Document	157
X'D3A9AD'	ENG	End Named Page Group	163
X'D3A9AF'	EPG	End Page	166
X'D3A9BB'	EGR	End Graphics Object	159
X'D3A9C4'	EDG	End Document Environment Group	155
X'D3A9C5'	EFG	End Form Environment Group (**O**)	503
X'D3A9C6'	ERG	End Resource Group	170
X'D3A9C7'	EOG	End Object Environment Group	165
X'D3A9C9'	EAG	End Active Environment Group	151
X'D3A9CC'	EMM	End Medium Map	161
X'D3A9CD'	EFM	End Form Map	158
X'D3A9CE'	ERS	End Resource	171
X'D3A9D9'	ESG	End Resource Environment Group	172
X'D3A9DF'	EMO	End Overlay	162
X'D3A9EB'	EBC	End Bar Code Object	152
X'D3A9FB'	EIM	End Image Object	160
X'D3AB77'	MCA	Map Color Attribute Table	207
X'D3AB88'	MMT	Map Media Type	260
X'D3AB8A'	MCF	Map Coded Font	213
X'D3AB92'	MCD	Map Container Data	211
X'D3ABAF'	MPG	Map Page	263
X'D3ABBB'	MGO	Map Graphics Object	245
X'D3ABC3'	MDR	Map Data Resource	223
X'D3ABCC'	IMM	Invoke Medium Map	178
X'D3ABD8'	MPO	Map Page Overlay	265

Table 36. Structured Fields Sorted by ID (continued)

Identifier	Acronym	Structured Field Name	Page
X'D3ABEA'	MSU	Map Suppression	268
X'D3ABEB'	MBC	Map Bar Code Object	206
X'D3ABFB'	MIO	Map Image Object	246
X'D3AC6B'	OBP	Object Area Position	272
X'D3AC7B'	ICP	IM Image Cell Position (**C**)	527
X'D3ACAF'	PGP-1	Page Position Format-1 (**C**)	525
X'D3ADC3'	PPO	Preprocess Presentation Object	299
X'D3AF5F'	IPS	Include Page Segment	197
X'D3AFAF'	IPG	Include Page	191
X'D3AFC3'	IOB	Include Object	180
X'D3AFD8'	IPO	Include Page Overlay	194
X'D3B077'	CAT	Color Attribute Table	149
X'D3B15F'	MPS	Map Page Segment	267
X'D3B18A'	MCF-1	Map Coded Font Format-1 (**C**)	522
X'D3B19B'	PTD	Presentation Text Data Descriptor	308
X'D3B1AF'	PGP	Page Position	282
X'D3B1DF'	MMO	Map Medium Overlay	259
X'D3B288'	PFC	Presentation Fidelity Control	277
X'D3B2A7'	IEL	Index Element	176
X'D3B490'	LLE	Link Logical Element	199
X'D3EE7B'	IRD	IM Image Raster Data (**C**)	533
X'D3EE92'	OCD	Object Container Data	276
X'D3EE9B'	PTX	Presentation Text Data	309
X'D3EEBB'	GAD	Graphics Data	173
X'D3EEEB'	BDA	Bar Code Data	109
X'D3EEEE'	NOP	No Operation	269
X'D3EEFB'	IPD	Image Picture Data	190

Key:

O	Obsolete
R	Retired
C	Coexistence

MO:DCA Structured Fields Sorted by Acronym

Table 37. Structured Fields Sorted by Acronym

Acronym	Identifier	Structured Field Name	Page
BAG	X'D3A8C9'	Begin Active Environment Group	104
BBC	X'D3A8EB'	Begin Bar Code Object	105
BCA	X'D3A877'	Begin Color Attribute Table	107
BDA	X'D3EEEB'	Bar Code Data	109
BDD	X'D3A6EB'	Bar Code Data Descriptor	110
BDG	X'D3A8C4'	Begin Document Environment Group	111
BDI	X'D3A8A7'	Begin Document Index	112
BDT	X'D3A8A8'	Begin Document	114
BFG	X'D3A8C5'	Begin Form Environment Group (**O**)	502
BFM	X'D3A8CD'	Begin Form Map	116
BGR	X'D3A8BB'	Begin Graphics Object	118
BII	X'D3A87B'	Begin IM Image (**C**)	526
BIM	X'D3A8FB'	Begin Image Object	120
BMM	X'D3A8CC'	Begin Medium Map	122
BMO	X'D3A8DF'	Begin Overlay	124
BNG	X'D3A8AD'	Begin Named Page Group	126
BOC	X'D3A892'	Begin Object Container	129
BOG	X'D3A8C7'	Begin Object Environment Group	133
BPG	X'D3A8AF'	Begin Page	134
BPS	X'D3A85F'	Begin Page Segment	137
BPT	X'D3A89B'	Begin Presentation Text Object	139
BRG	X'D3A8C6'	Begin Resource Group	141
BRS	X'D3A8CE'	Begin Resource	143
BSG	X'D3A8D9'	Begin Resource Environment Group	148
CAT	X'D3B077'	Color Attribute Table	149
CDD	X'D3A692'	Container Data Descriptor	150
CTC	X'D3A79B'	Composed Text Control (**O**)	502
EAG	X'D3A9C9'	End Active Environment Group	151
EBC	X'D3A9EB'	End Bar Code Object	152
ECA	X'D3A977'	End Color Attribute Table	153
EDG	X'D3A9C4'	End Document Environment Group	155
EDI	X'D3A9A7'	End Document Index	156
EDT	X'D3A9A8'	End Document	157
EFG	X'D3A9C5'	End Form Environment Group (**O**)	503
EFM	X'D3A9CD'	End Form Map	158
EGR	X'D3A9BB'	End Graphics Object	159
EII	X'D3A97B'	End IM Image (**C**)	527
EIM	X'D3A9FB'	End Image Object	160

Table 37. Structured Fields Sorted by Acronym (continued)

Acronym	Identifier	Structured Field Name	Page
EMM	X'D3A9CC'	End Medium Map	161
EMO	X'D3A9DF'	End Overlay	162
ENG	X'D3A9AD'	End Named Page Group	163
EOC	X'D3A992'	End Object Container	164
EOG	X'D3A9C7'	End Object Environment Group	165
EPG	X'D3A9AF'	End Page	166
EPS	X'D3A95F'	End Page Segment	167
EPT	X'D3A99B'	End Presentation Text Object	168
ERG	X'D3A9C6'	End Resource Group	170
ERS	X'D3A9CE'	End Resource	171
ESG	X'D3A9D9'	End Resource Environment Group	172
FGD	X'D3A6C5'	Form Environment Group Descriptor (**O**)	503
GAD	X'D3EEBB'	Graphics Data	173
GDD	X'D3A6BB'	Graphics Data Descriptor	174
ICP	X'D3AC7B'	IM Image Cell Position (**C**)	527
IDD	X'D3A6FB'	Image Data Descriptor	175
IEL	X'D3B2A7'	Index Element	176
IID	X'D3A67B'	Image Input Descriptor (**C**)	529
IMM	X'D3ABCC'	Invoke Medium Map	178
IOB	X'D3AFC3'	Include Object	180
IOC	X'D3A77B'	IM Image Output Control (**C**)	530
IPD	X'D3EEFB'	Image Picture Data	190
IPG	X'D3AFAF'	Include Page	191
IPO	X'D3AFD8'	Include Page Overlay	194
IPS	X'D3AF5F'	Include Page Segment	197
IRD	X'D3EE7B'	IM Image Raster Data (**C**)	533
LLE	X'D3B490'	Link Logical Element	199
MBC	X'D3ABEB'	Map Bar Code Object	206
MCA	X'D3AB77'	Map Color Attribute Table	207
MCC	X'D3A288'	Medium Copy Count	209
MCD	X'D3AB92'	Map Container Data	211
MCF	X'D3AB8A'	Map Coded Font	213
MCF-1	X'D3B18A'	Map Coded Font Format-1 (**C**)	522
MDD	X'D3A688'	Medium Descriptor	220
MDR	X'D3ABC3'	Map Data Resource	223
MFC	X'D3A088'	Medium Finishing Control	236
MGO	X'D3ABBB'	Map Graphics Object	245
MIO	X'D3ABFB'	Map Image Object	246
MMC	X'D3A788'	Medium Modification Control	248
MMO	X'D3B1DF'	Map Medium Overlay	259

Table 37. Structured Fields Sorted by Acronym (continued)

Acronym	Identifier	Structured Field Name	Page
MMT	X'D3AB88'	Map Media Type	260
MPG	X'D3ABAF'	Map Page	263
MPO	X'D3ABD8'	Map Page Overlay	265
MPS	X'D3B15F'	Map Page Segment	267
MSU	X'D3ABEA'	Map Suppression	268
NOP	X'D3EEEE'	No Operation	269
OBD	X'D3A66B'	Object Area Descriptor	270
OBP	X'D3AC6B'	Object Area Position	272
OCD	X'D3EE92'	Object Container Data	276
PFC	X'D3B288'	Presentation Fidelity Control	277
PGD	X'D3A6AF'	Page Descriptor	279
PGP	X'D3B1AF'	Page Position	282
PGP-1	X'D3ACAF'	Page Position Format-1 (**C**)	525
PMC	X'D3A7AF'	Page Modification Control	297
PPO	X'D3ADC3'	Preprocess Presentation Object	299
PTD	X'D3B19B'	Presentation Text Data Descriptor	308
PTD-1	X'D3A69B'	Presentation Text Descriptor Format-1 (**C**)	526
PTX	X'D3EE9B'	Presentation Text Data	309
TLE	X'D3A090'	Tag Logical Element	310

Key:

O	Obsolete
R	Retired
C	Coexistence

MO:DCA Triplets Sorted by Identifier

Table 38. Triplets Sorted by ID

Triplet ID	Triplet Name	Page
X'01'	Coded Graphic Character Set Global ID	317
X'02'	Fully Qualified Name	320
X'04'	Mapping Option	332
X'10'	Object Classification	335
X'18'	MO:DCA Interchange Set	339
X'1D'	Text Orientation (**R**)	505
X'1F'	Font Descriptor Specification	341
X'20'	Font Coded Graphic Character Set Global Identifier	345
X'21'	Object Function Set Specification	346
X'21'	Resource Object Type (**R**)	506
X'22'	Extended Resource Local ID	348
X'24'	Resource Local ID	350
X'25'	Resource Section Number	351
X'26'	Character Rotation	352
X'27'	Line Data Object Position Migration (**R**)	507
X'2D'	Object Byte Offset	353
X'36'	Attribute Value	354
X'43'	Descriptor Position	355
X'45'	Media Eject Control	356
X'46'	Page Overlay Conditional Processing	361
X'47'	Resource Usage Attribute	363
X'4B'	Object Area Measurement Units	364
X'4C'	Object Area Size	365
X'4D'	Area Definition	366
X'4E'	Color Specification	367
X'50'	Encoding Scheme ID	371
X'56'	Medium Map Page Number	374
X'57'	Object Byte Extent	375
X'58'	Object Structured Field Offset	376
X'59'	Object Structured Field Extent	377
X'5A'	Object Offset	378
X'5D'	Font Horizontal Scale Factor	380
X'5E'	Object Count	381
X'62'	Local Date and Time Stamp	383
X'63'	Object Checksum (**R**)	511
X'64'	Object Origin Identifier (**R**)	513
X'65'	Comment	385
X'68'	Medium Orientation	386

Table 38. Triplets Sorted by ID (continued)

Triplet ID	Triplet Name	Page
X'6C'	Resource Object Include	388
X'70'	Presentation Space Reset Mixing	390
X'71'	Presentation Space Mixing Rules	392
X'72'	Universal Date and Time Stamp	394
X'73'	IMM Insertion (**R**)	514
X'74'	Toner Saver	397
X'75'	Color Fidelity	399
X'78'	Font Fidelity	401
X'80'	Attribute Qualifier	402
X'81'	Page Position Information	403
X'82'	Parameter Value	404
X'83'	Presentation Control	405
X'84'	Font Resolution and Metric Technology	406
X'85'	Finishing Operation	407
X'86'	Text Fidelity	414
X'87'	Media Fidelity	416
X'88'	Finishing Fidelity	418
X'8B'	Data-Object Font Descriptor	420
X'8E'	UP3i Finishing Operation	424

Key:

O	Obsolete
R	Retired
C	Coexistence

MO:DCA Triplets Sorted by Name

Table 39. Triplets Sorted by Name

Triplet Name	Triplet ID	Page
Area Definition	X'4D'	366
Attribute Qualifier	X'80'	402
Attribute Value	X'36'	354
Character Rotation	X'26'	352
Coded Graphic Character Set Global ID	X'01'	317
Color Fidelity	X'75'	399
Color Specification	X'4E'	367
Comment	X'65'	385
Data-Object Font Descriptor	X'8B'	420
Descriptor Position	X'43'	355
Encoding Scheme ID	X'50'	371
Extended Resource Local ID	X'22'	348
Finishing Fidelity	X'88'	418
Finishing Operation	X'85'	407
Font Coded Graphic Character Set Global Identifier	X'20'	345
Font Fidelity	X'78'	401
Font Descriptor Specification	X'1F'	341
Font Horizontal Scale Factor	X'5D'	380
Font Resolution and Metric Technology	X'84'	406
Fully Qualified Name	X'02'	320
IMM Insertion (**R**)	X'73'	514
Line Data Object Position Migration (**R**)	X'27'	507
Local Date and Time Stamp	X'62'	383
Mapping Option	X'04'	332
Media Eject Control	X'45'	356
Media Fidelity	X'87'	416
Medium Map Page Number	X'56'	374
Medium Orientation	X'68'	386
MO:DCA Interchange Set	X'18'	339
Object Area Measurement Units	X'4B'	364
Object Area Size	X'4C'	365
Object Byte Extent	X'57'	375
Object Byte Offset	X'2D'	353
Object Checksum (**R**)	X'63'	511
Object Classification	X'10'	335
Object Count	X'5E'	381
Object Function Set Specification	X'21'	346
Object Offset	X'5A'	378

Table 39. Triplets Sorted by Name (continued)

Triplet Name	Triplet ID	Page
Object Origin Identifier (**R**)	X'64'	513
Object Structured Field Extent	X'59'	377
Object Structured Field Offset	X'58'	376
Page Overlay Conditional Processing	X'46'	361
Page Position Information	X'81'	403
Parameter Value	X'82'	404
Presentation Control	X'83'	405
Presentation Space Mixing Rules	X'71'	392
Presentation Space Reset Mixing	X'70'	390
Resource Local ID	X'24'	350
Resource Object Include	X'6C'	388
Resource Object Type (**R**)	X'21'	506
Resource Section Number	X'25'	351
Resource Usage Attribute	X'47'	363
Text Fidelity	X'86'	414
Text Orientation (**R**)	X'1D'	505
Toner Saver	X'74'	397
Universal Date and Time Stamp	X'72'	394
UP3i Finishing Operation	X'8E'	424

Key:

O	Obsolete
R	Retired
C	Coexistence

Summary of Changes

This seventh edition of the *Mixed Object Document Content Architecture Reference* contains the following significant architecture extensions:

- Support for rendering text using TrueType and OpenType fonts:
  - The text can be encoded in Unicode (UTF-16 or UTF-8) or in any of the currently-used EBCDIC or ASCII encoding schemes
  - The text can be in PTOCA, AFP GOCA, or BCOCA objects
  - The fonts are installed and used in their unaltered, native file format
  - The fonts are referenced with an MDR structured field using their *full font name*, and the references are processed consistently on all supported platforms using an architected Resource Access Table (RAT)
  - The fonts may be captured in the printer across job boundaries to avoid repeated downloads
  - The TrueType and OpenType MDR font references can be mixed on a page in any combination with FOCA MCF font references
- Major extensions for document finishing:
  - Support for finishing operations that are specified using the UP3i protocol and passed unaltered from the Form Definition and from Medium Maps to the printer
  - Allowing finishing operations to be changed for groups of pages within the document by changing Medium Maps
  - Support for a Finishing Fidelity control that allows the job submitter to instruct the presentation system how to continue if a specified finishing operation cannot be satisfied
  - Support for additional finishing operations:
    - Punch
    - Saddle-stitch In
    - Center-fold In
- The ability to identify objects to be pre-processed before printing of the document is started using the new Preprocess Presentation Object (PPO) structured field
- Allowing the Link Logical Element (LLE) structured field to reference a web-based target using a *url*
- Support for a new bi-level tiled image object: IOCA FS40
- Negative offsets for overlays and page segments
- Allowing a page group to be empty
- Support for two new medium orientations: *reverse portrait* and *reverse landscape*
- Changing the definition of the Begin Resource (BRS) and End Resource (ERS) structured fields to be valid MO:DCA-P structured fields that can be specified in printfile-level resource groups
- Numerous corrections and clarifications.

As stated in the edition notice, the additions are marked in this publication using revision bars located on the left-hand side 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.

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
AFP
AIX®
AS/400®
Bar Code Object Content Architecture
BCOCA
Common User Access®
CUA®
EDMSuite
GDDM®
IBM®
ImagePlus®
Infoprint®
Intelligent Printer Data Stream
IPDS
Mixed Object Document Content Architecture
MO:DCA
MVS
MVS/DFP
MVS/ESA
OS/2®
OS/390®
OS/400®
Print Services Facility
SAA®
Systems Application Architecture®
VisualInfo
WIN/OS2®

Microsoft, Windows, and Windows NT are 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*, ZC20-1699.

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*.

Abstract Syntax Notation One (ASN.1). A notation for defining data structures and data types. The notation is defined in international standard ISO/IEC 8824:1990(E). See also *object identifier*.

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. (1) 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. (2) A comment or explanation associated with the contents of a document component. An example of an annotation is a string of text that represents a comment on an image object on a page.

annotation link. A link type that specifies the linkage from a source document component to a target document component that contains an annotation.

annotation object. An object that contains an annotation. Objects that are targets of annotation links are annotation objects.

APA. See *all points addressable*.

append. An addition to or continuation of the contents of a document component. An example of an append is a string of text that is a continuation of an existing string of text on a page.

append link. A link type that specifies the linkage from the end of a source document component to a target document component that contains an append.

append object. An object that contains an append. Objects that are targets of append links are append objects.

application. (1) The use to which an information system is put. (2) A collection of software components used to perform specific types of work on a computer.

application program. A program written for or by a user that applies to the user's work.

architected. Identifies data that is defined and controlled by an architecture. Contrast with *unarchitected*.

ASN.1. See *Abstract Syntax Notation One*.

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. Contrast with *foreground*.

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.

baseline. A conceptual line with respect to which successive characters are aligned.

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*.

big-endian. A bit or byte ordering where the leftmost bits or bytes (those with a lower address) are most significant. Contrast with *little-endian*.

BITS. A data type for architecture syntax, indicating one or more bytes to be interpreted as bit string information.

blend. A mixing rule in which the intersection of part of a new presentation space P_{new} with part of an existing presentation space $P_{existing}$ changes to a new color attribute that represents a color-mixing of the color attributes of P:sub.new:esub. with the color attributes of $P_{existing}$. For example, if P_{new} has foreground color attribute blue and $P_{existing}$ has foreground color attribute yellow, the area where the two foregrounds intersect assumes a color attribute of green if the mixing rule is blend.

C

CCS. See *Common Communications Support*.

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 that 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. (1) 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*. (2) In IPDS, a similar concept is *font inline sequence*, which specifies a counter-clockwise character 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*.

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 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. (1) A resource object containing descriptive information, graphic character identifiers, and code points corresponding to a coded graphic character set. Graphic characters can be added over time; therefore, to specifically identify a code page, both a GCSGID and a CPGID should be used. See also *coded graphic character set*. (2) 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*.

code table. A table showing the character allocated to each code point in a code. See also *code page* and *code point*.

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 model. See *color space*.

color of medium. The color of a presentation space before any data is added to it. Synonymous with *reset color*.

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 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*.

command. (1) In the IPDS architecture, a structured field sent from a host to a printer. (2) A request for system action.

Common Communications Support (CCS). Protocols and conventions for connecting systems and software. One of the three SAA architectural areas (the other two being Common Programming Interface and Common User Access®).

CPI. See *Common Programming Interface*.

Common Programming Interface (CPI). Definitions of those application development languages and services that have (or are intended to have) implementations on and a high degree of commonality across the SAA environments. One of the three SAA architectural areas (the other two being Common Programming Interface and Common User Access).

Common User Access (CUA®). Guidelines for the dialog between a person and the workstation or terminal. One of the three SAA architectural areas (the other two being Common Communications Support and Common Programming Interface).

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*.

CPI. See *Common User Access*.

D

data block. A deprecated term for object area.

data element. A unit of data that is considered indivisible.

data frame. A rectangular division of computer output on microfilm.

data-object font. In the MO:DCA architecture, a complete non-FOCA font resource object that is analogous to a coded font. Examples of data-object fonts are TrueType fonts and OpenType fonts.

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. An architected part of a document data stream. Examples of document components are documents, pages, page groups, indexes, resource groups, objects, and process elements.

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 hierarchy. An ordering of the document in terms of its lower-level components. The components are ordered by decreasing level as follows:

- Document (highest level)
- Page group
- Page
- Data object (lowest level)

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*.

escapement direction. In FOCA, the direction from a character reference point to the character escapement point, that is, the font designer's intended direction for successive character shapes.

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 local identifier. A binary identifier that is mapped by the environment to a named resource to identify a font. See also *local identifier*.

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 by object data. Contrast with *background*.

form. A division of the physical medium; multiple forms can exist on a physical medium. For example, a roll of paper might be divided by a printer into rectangular pieces of paper, each representing a form. Envelopes are an example of a physical medium that comprises only one form. A form has two sides, a front side and a back side. 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. See *Graphic Character Set Global Identifier*.

GID. See *global identifier*.

Global Identifier (GID). One of the following:
- A Coded Graphic Character Set Global Identifier (CGCSGID)
- 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)
- A Global Resource Identifier (GRID)
- An encoded graphic character string that, when qualified by the associated CGCSGID, specifies a reference name
- An object identifier (OID), as defined in ISO/IEC 8824:1990(E)
- A Uniform Resource Locator (URL), as defined in RFC 1738, Internet Engineering Task Force (IETF), December, 1994
- An identifier used by a data object to reference a resource

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.

- For fixed-pitch, uniform character increment fonts: the fixed character increment, which is also the space character increment.
- For PSM fonts: the width of the space character.
- 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. (3) 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. (1) 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*. (2) In FOCA, the numerator of a scaling ratio, determined by dividing the horizontal scale factor by the vertical font size. If the value specified is greater or less than the specified vertical font size, the graphic characters and their corresponding metric values are stretched or compressed in the horizontal direction relative to the vertical direction by the scaling ratio indicated.

hypermedia. Interlinked pieces of information consisting of a variety of data types such as text, graphics, image, audio, video.

hypertext. Interlinked pieces of information consisting primarily of text.

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*.

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.

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*.

| **little-endian.** A bit or byte ordering where the rightmost bits or bytes (those with a higher address) are most significant. Contrast with *big-endian*.

local identifier (LID). An identifier that is mapped by the environment to a named resource.

link. A logical connection from a source document component to a target document component.

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 overlays can be mapped to a logical page. A logical page has specifiable characteristics, such as size, shape, orientation, and offset. The shape of a logical 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, collated copies without having to resend the document to the printer multiple times, 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.

mixing. (1) Combining foreground and background of one presentation space with foreground and background of another presentation space in areas where the presentation spaces intersect. (2) Combining foreground and background of multiple intersecting object data elements in the object presentation space.

mixing rule. A method for specifying the color attributes of the resultant foreground and background in areas where two presentation spaces intersect.

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 the MO:DCA architecture 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".

navigation. The traversing of a document based on links between contextually-related document components.

navigation link. A link type that specifies the linkage from a source document component to a contextually-related target document component. Navigation links may be used to support applications such as hypertext and hypermedia.

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*.

non-presentation object. An object that is not a presentation object. Contrast with *presentation object*.

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 partitioning of a side of a sheet into a fixed number of equal size partitions. For example, 4-up divides each side of a sheet into four equal partitions.

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.

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.

object identifier (OID). A notation for assigning globally-unambiguous names. The notation is defined in international standard ISO/IEC 8824:1990(E).

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.

OID. See *object identifier*.

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 intersection 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*.

overpaint. A mixing rule in which the intersection of part of a new presentation space P_{new} with an existing presentation space $P_{existing}$ keeps the color attribute of P_{new}. This is also referred to as “opaque” or “knock-out” mixing. See also *mixing rule*. Contrast with *blend* and *underpaint*.

P

page. (1) A data stream object delimited by a Begin Page structured field and an End Page structured field. A page can contain presentation data such as text, image, graphics, and bar code data. (2) The final representation of such an object on a physical medium.

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 the MO:DCA architecture, 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. (1) A variable that is given a constant value for a specified application. (2) A variable used in conjunction with a command to affect its result.

partition. Dividing the medium presentation space into a specified number of equal-sized areas in a manner determined by the current physical media.

pattern. An array of symbols used to fill an area.

pattern symbol. The geometric construct that is used repetitively to generate a pattern. Examples of symbols are dots, squares, and triangles.

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, a roll of paper, an envelope, and a display screen. See also *medium presentation space* and *sheet*.

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 media are a display screen and a sheet of paper.

presentation object. An object that describes presentation data such as text, image, and graphics, in a paginated, final-form format suitable for presentation on a page. Contrast with *non-presentation object*.

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. 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*.

process element. A document component that is defined by a structured field and that facilitates a form of document processing that does not affect the presentation of the document. Examples of process elements are Tag Logical Elements (TLEs) that specify document attributes and Link Logical Elements (LLEs) that specify linkages between document components.

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.

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 the MO:DCA architecture, 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.

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*.

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.

scaling ratio. In FOCA, the ratio of horizontal to vertical scaling of the graphic characters. See also *horizontal scale factor*.

secondary resource. A resource for an object that may itself be a resource.

section. A portion of a double-byte code page that consists of 256 consecutive entries. The first byte of a two-byte code point is the section identifier. A code-page section is also called a code-page ward in some environments. 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*.

shade. Variation of a color produced by mixing it with black.

sheet. A division of the physical medium; multiple sheets can exist on a physical medium. For example, a roll of paper might be divided by a printer into rectangular pieces of paper, each representing a sheet. Envelopes are an example of a physical medium that comprises only one sheet. A sheet has two sides, a front side and a back side. 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, type, and category. Examples of structured field types are begin, end, data, and descriptor. Examples of structured field categories are presentation text, image, graphics, and page.

subordinate object. An object that is lower in the document hierarchy than a given object. For example, a page is a subordinate object to a page group, and a page group is a subordinate object to a document.

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.

surrogates. Pairs of Unicode code points that allow for the encoding of as many as 1 million additional characters without any use of escape codes.

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.

tint. Variation of a color produced by mixing it with white.

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 data 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, weight class, width class, and posture, regardless of size. For example, Helvetica Bold Condensed Italic, 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*.

underpaint. A mixing rule in which the intersection of part of a new presentation space P_{new} with part of an existing presentation space $P_{existing}$ keeps the color attribute of $P_{existing}$. This is also referred to as “transparent” or “leave alone” mixing. See also *mixing rule*. Contrast with *blend* and *overpaint*.

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*.

Unicode. A character encoding standard for information processing that includes all major scripts of the world. Unicode defines a consistent way of encoding multilingual text. Unicode specifies a numeric value, a name, and other attributes - such as directionality - for each of its characters; for example, the name for $ is “dollar sign” and its numeric value is X'0024'. This Unicode value is called a *Unicode code point* and is represented as U+*nnnn*. Unicode provides for three encoding forms (UTF-8, UTF-16, and UTF-32), described as follows:

UTF-8 A byte-oriented form that is designed for ease of use in traditional ASCII environments. Each UTF-8 code point contains from one to four bytes. All Unicode code points can be encoded in UTF-8 and all 7-bit ASCII characters can be encoded in one byte.

UTF-16 The default Unicode encoding. A fixed, two-byte Unicode encoding form that can contain surrogates and identifies the byte order of each UTF-16 code point via a Byte Order Mark in the first 2 bytes of the data.

UTF-16BE UTF-16 that uses big endian byte order; this is the byte order for all multi-byte data within AFP data streams. The Byte Order Mark is not necessary when the data is externally identified as UTF-16BE (or UTF-16LE).

UTF-16LE UTF-16 that uses little endian byte order.

UTF-32 A fixed, four-byte Unicode encoding form in which each UTF-32 code point is precisely identical to the Unicode code point.

UTF-32BE UTF-32 serialized as bytes in most significant byte first order (big endian). UTF-32BE is structurally the same as UCS-4.

UTF-32LE UTF-32 serialized as bytes in least significant byte first order (little endian).

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 prior to 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*.

writing mode. An identified mode for the setting of text in a writing system, usually corresponding to a nominal escapement direction of the graphic characters in that mode; for example, left-to-right, right-to-left, top-to-bottom.

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

Readers' Comments — We'd Like to Hear from You

Data Stream and Object Architectures
Mixed Object Document Content Architecture Reference

Publication No. SC31-6802-06

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.

Name

Company or Organization

Phone No.

Address

Readers' Comments — We'd Like to Hear from You
SC31-6802-06

Cut or Fold
Along Line

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

Cut or Fold
Along Line

File Number: S370-40

Printed in USA

SC31-6802-06