Advanced Function Presentation Consortium
Data Stream and Object Architectures

Presentation Text Object Content Architecture Reference

AFPC-0009-03

AFPConsortium
Advanced Function Presentation

Note:

Before using this information, read the information in "Notices" on page 171.

AFPC-0009-03
Fourth Edition (March 2016)

This edition applies to the Presentation Text Object Content Architecture (PTOCA). It is the first edition produced by the AFP Consortium™(AFPC™) and replaces and makes obsolete the previous edition, SC31-6803-02, published by the IBM® Corporation. This edition remains current until a new edition is published.

Specific changes are indicated by a vertical bar to the left of the change. For a detailed list of the changes, see "Summary of Changes" on page ix.

Internet

Visit our home page: www.afpcinc.org

Preface

This book describes the functions and services associated with the Presentation Text Object Content Architecture (PTOCA) 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.

AFP Consortium (AFPC)

The Advanced Function Presentation™(AFP™) architectures began as the strategic, general purpose document and information presentation architecture for the IBM Corporation. The first specifications and products go back to 1984. Although all of the components of the architecture have grown over the years, the major concepts of object-driven structures, print integrity, resource management, and support for high print speeds were built in from the start.

In the early twenty-first century, IBM saw the need to enable applications to create color output that is independent from the device used for printing and to preserve color consistency, quality, and fidelity of the printed material. This need resulted in the formation, in October 2004, of the AFP Color Consortium™ (AFPCC™). The goal was to extend the object architectures with support for full-color devices including support for comprehensive color management. The idea of doing this via a consortium consisting of the primary AFP architecture users was to build synergism with partners from across the relevant industries, such as hardware manufacturers that produce printers as well as software vendors of composition, work flow, viewer, and transform tools. Quickly more than 30 members came together in regular meetings and work group sessions to create the AFP Color Management Object Content Architecture™ (CMOCA™). A major milestone was reached by the AFP Color Consortium with the initial official release of the CMOCA specification in May 2006.

Since the cooperation between the members of the AFP Color Consortium turned out to be very effective and valuable, it was decided to broaden the scope of the consortium efforts and IBM soon announced its plans to open up the complete scope of the AFP architecture to the consortium. In June 2007, IBM's role as founding member of the consortium was transferred to the InfoPrint® Solutions Company, an IBM/Ricoh® joint venture. In February 2009, the consortium was incorporated under a new set of bylaws with tiered membership and shared governance resulting in the creation of a formal open standards body called the AFP Consortium (AFPC). Ownership of and responsibility for the AFP architectures was transferred at that time to the AFP Consortium.

How to Use This Book

This book is divided into six chapters, three appendixes, and a glossary.

- Chapter 1, “Overview of Presentation Architecture” introduces the AFP presentation architectures and positions Presentation Text Object Content Architecture as a strategic object content architecture.
- Chapter 2, “Introduction to PTOCA” briefly states the purpose and function of PTOCA.
- Chapter 3, “Overview of PTOCA” introduces the concepts that form the basis of PTOCA and provides a brief description of the data structures.
- Chapter 4, “Data Structures in PTOCA” provides the detailed syntax, semantics, and pragmatics of the data structures found in PTOCA.
- Chapter 5, “Exception Handling in PTOCA” describes how exceptions are handled in PTOCA and lists the exception codes.
- Chapter 6, “Compliance with PTOCA” describes how products may be valid generators or receivers of PTOCA.
- Appendix A, “MO:DCA Environment” describes the Presentation Text object in the context of a MO:DCA™ data stream.
- Appendix B, “IPDS Environment” describes the Presentation Text object in the context of an IPDS™ data stream.
- Appendix C, “PTOCA Retired Functions” describes the retired PTOCA functions.
- The “Glossary” on page 173 defines some of the terms used within this book.

How to Read the Syntax Diagrams

Throughout this book, syntax is described using the structure defined below. 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

Syntax for PTOCA is shown in tables like the following:

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
The field's offset, data type, or both		Name of field if applicable	Range of valid values if applicable	Meaning or purpose of the data element			

M/O *M* means mandatory. *O* means optional.

Def *Y* means that a default value is defined for the field. *N* means that there is no default value defined for the field.

Ind *Y* means that the field defaults to a hierarchical default value when the default indicator (X'F..F') is present. *N* means that the default indicator semantic is not valid for the field.

The following is an example of PTOCA syntax for the Begin Line (BLN) control sequence as it appears in this book:

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	2	Control sequence length	M	N	N
3	CODE	TYPE	X'D8' – X'D9'	Control sequence function type	M	N	N

Please refer to "Control Sequence Detailed Descriptions" on page 45 for a more detailed description of PTOCA syntax.

Related Publications

Several other publications can help you understand the architecture concepts described in this book. AFP Consortium publications and a few other AFP publications are available on the AFP Consortium web site, `http://www.afpcinc.org`.

Table 1. AFP Consortium Architecture References

AFP Architecture Publication	Book Identification
AFP Programming Guide and Line Data Reference	S544-3884 (IBM)
Bar Code Object Content Architecture™ Reference	AFPC-0005
Color Management Object Content Architecture Reference	AFPC-0006
Font Object Content Architecture Reference	AFPC-0007
Graphics Object Content Architecture for Advanced Function Presentation Reference	AFPC-0008
Image Object Content Architecture Reference	AFPC-0003
Intelligent Printer Data Stream™ (IPDS) Reference	AFPC-0001
Metadata Object Content Architecture Reference	AFPC-0013
Mixed Object Document Content Architecture™ (MO:DCA) Reference	AFPC-0004
Presentation Text Object Content Architecture Reference	AFPC-0009

Table 2. Additional AFP Consortium Documentation

AFPC Publication	Book Identification
AFP Color Management Architecture™ (ACMA™)	AFPC-0015
AFPC Company Abbreviation Registry	AFPC-0012
AFPC Font Typeface Registry	AFPC-0016
BCOCA™ Frequently Asked Questions	AFPC-0011
MO:DCA-L:The OS/2 PM Metafile (.met) Format	AFPC-0014
Presentation Object Subsets for AFP	AFPC-0002
Recommended IPDS Values for Object Container Versions	AFPC-0017

Table 3. AFP Font-Related Documentation

Publication	Book Identification
Character Data Representation Architecture Reference and Registry; please refer to the online version for the most current information: `http://www-01.ibm.com/software/globalization/cdra/index.html`	SC09-2190 (IBM)
Font Summary for AFP Font Collection	S544-5633 (IBM)
How To Use TrueType and OpenType Fonts in an AFP System	G544-5876 (IBM)
Technical Reference for Code Pages	S544-3802 (IBM)

Table 4. UP³I™ Architecture Documentation

UP³I Publication	Book Identification
Universal Printer Pre- and Post-Processing Interface (UP³I) Specification; please refer to the online version: `http://www.afpcinc.org`	Version 1.20

Summary of Changes

This fourth edition of the *PTOCA Reference* contains the following changes:

- The description of font usage in PTOCA has been expanded to include TrueType/Open Type fonts.
- The definition of control sequences to support the processing of complex scripts:
  - Glyph Advance Run (GAR)
  - Glyph ID Run (GIR)
  - Glyph Layout Control (GLC)
  - Glyph Offset Run (GOR)
  - Unicode Complex Text (UCT)
- The definition of a new PTOCA subset - PT4 - that consists of the PT3 subset plus the new control sequences listed above
- The Precision byte in the Set Text Color control sequence has been retired; this is documented in a new PTOCA Migration Functions appendix
- Editorial updates and clarifications
- An updated glossary
- Information about the AFP Consortium
- IBM product-specific information has been removed (CCS, CPI, CUA, MO:DCA-P, SAA, etc.)
- Style changes to make this book more consistent with other AFPC publications

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.

Contents

Figures

Tables

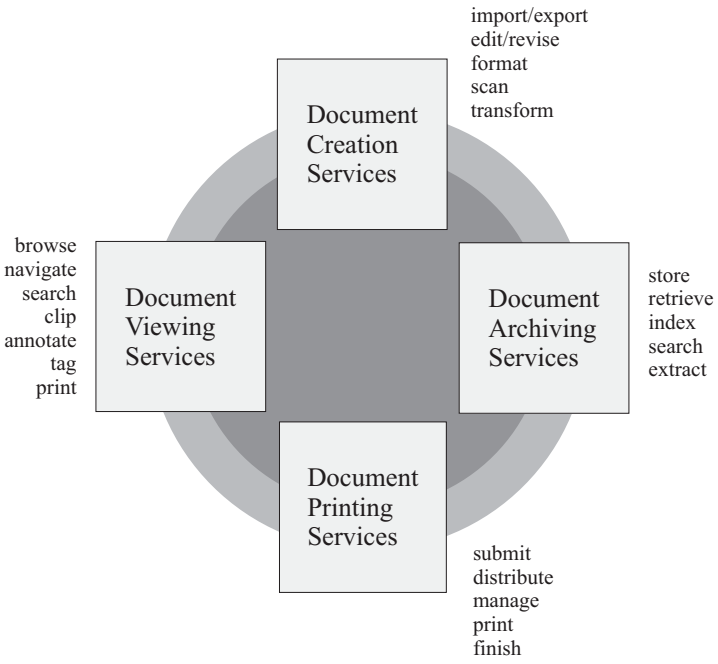
Chapter 1. Overview of Presentation Architecture

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. AFP 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. AFP architectures provide structures that support object-oriented models and client/server environments.

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

Architecture Components

AFP 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 the AFP Architecture, 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:

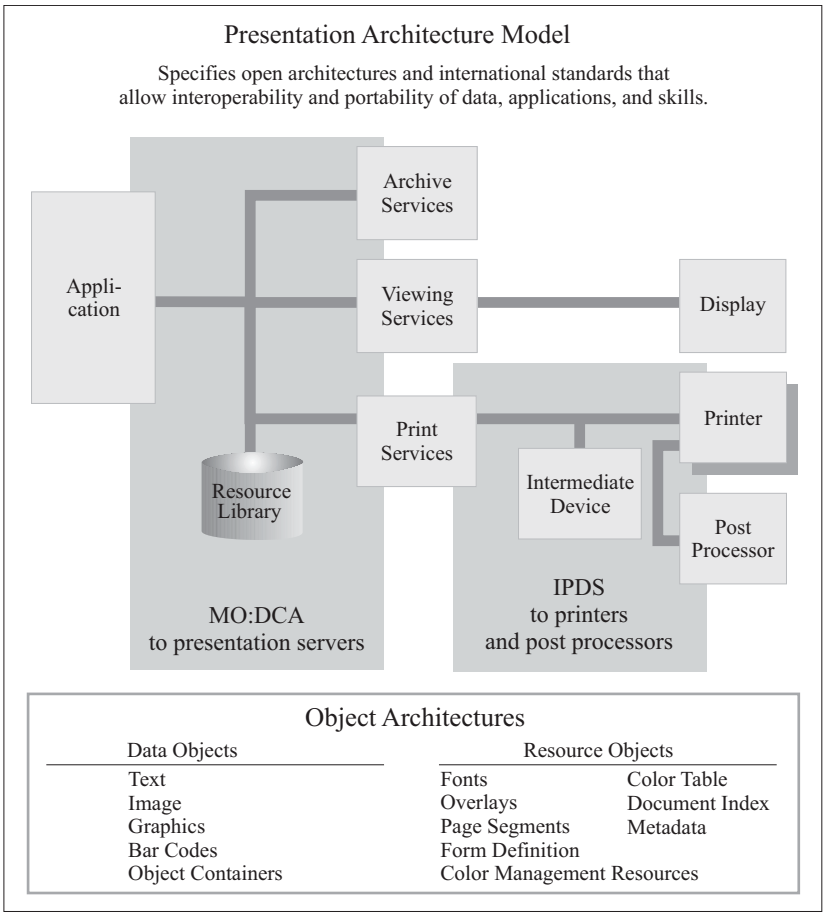
- *Mixed Object Document Content Architecture (MO:DCA)*
- *Intelligent Printer Data Stream (IPDS) Architecture*

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 can 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.

Figure 2 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 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.

Objects

The AFPC-defined object content architectures are:

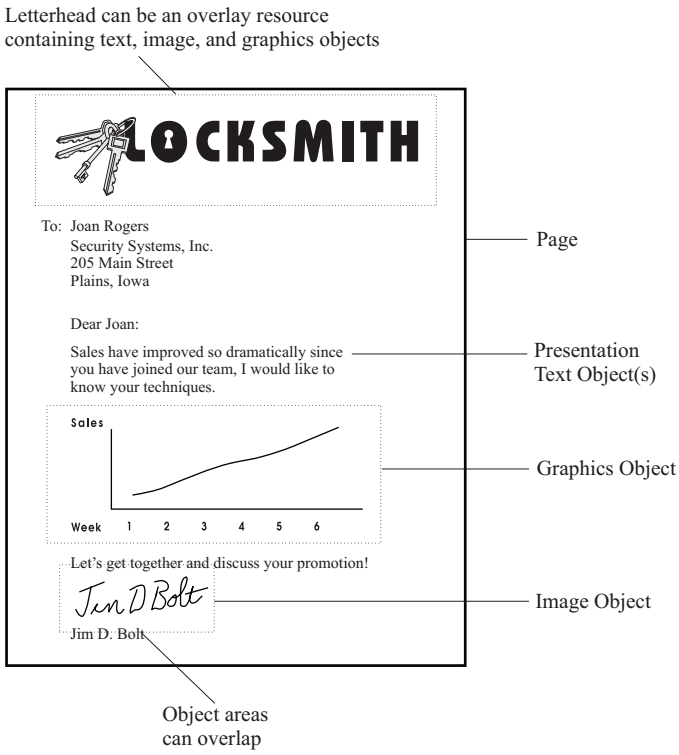
- *Presentation Text Object Content Architecture (PTOCA):* A data architecture for describing text objects that have been formatted for all-points-addressable presentations. Specifications of fonts, text color, and other visual attributes are included in the architecture definition.
- *Image Object Content 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 grayscale encoding 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.
- *Color Management Object Content Architecture (CMOCA):* A resource architecture for describing the color management information required to render presentation data.
- *Metadata Object Content Architecture (MOCA):* A resource architecture used to carry metadata in an AFP environment.

The MO:DCA and IPDS architectures also support data objects that are not defined by AFPC 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 shows an example of an all-points-addressable page composed of multiple presentation objects.

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.



Chapter 2. Introduction to PTOCA

The *Presentation Text* object is the data object used in document processing environments for representing text which has been prepared for presentation. *Text,* as used here, means an ordered string of characters, such as graphic symbols, numbers, and letters, that are suitable for the specific purpose of representing coherent information. Text which has been prepared for presentation has been reduced to a primitive form through explicit specification of the characters and their placement in the presentation space. *Control sequences* which designate specific control functions may be embedded within the text. These functions extend the primitive form by applying specific characteristics to the text when it is presented. The collection of the graphic characters and control sequences is called *Presentation Text*, and the object that contains the Presentation Text is called the Presentation Text object.

Presentation Text is associated with the output of text information. A Presentation Text object is the description of Presentation text for a portion of a document, the intended connotation being finished product or formatted output. This output version of text contained within the object is in the form specified by Presentation Text Object Content Architecture (PTOCA) and has been designed for direct output on devices, such as displays or printers.

A Presentation Text object is a device-independent, self-defining representation of a two-dimensional presentation space, called the *Presentation Text* object space, or *object space*, which contains the Presentation Text data. The rules of PTOCA specify how the object space is constituted, what the boundaries are for text positioning, what the text content is, and how the text content is to be placed within the object space, using concepts such as sequential order, orientation, and position.

Architecture Note: Note that when presentation text is processed in a MO:DCA environment where the Presentation Text Data Descriptor (PTD) is carried in the Active Environment Group (AEG) for the page, or when such text is processed in an IPDS environment, the Presentation Text object is bounded by the beginning of the page and the end of the page. This is sometimes called a *text major* environment. When the PTD is carried in the Object Environment Group (OEG) of a MO:DCA text object, the text object is bounded by the Begin Presentation Text (BPT) and End Presentation Text (EPT) structured fields. For such objects, the PTD in the AEG is ignored.

The Presentation Text object space is defined on the X_p,Y_p coordinate system, which is an orthogonal coordinate system based on the fourth quadrant of a standard Cartesian coordinate system. The object space is positioned within the data stream's object area. Coincident with the X_p,Y_p coordinate system is the I,B coordinate system, which is a translation of the X_p,Y_p coordinate system.

The position of the elements in the object space is described in terms of the I,B coordinate system. The increasing I-axis is the inline direction, which is normally the reading direction of the text. The increasing B-axis is the baseline direction, which is normally the direction for adding lines of text.

The basic elements of the object are the graphic characters which are identified as code points of a code page. The identification of graphic characters, their relationship to each other, and the relationship of the code point to the graphic character are given by the coded font selected to present the text.

The relationship of the elements to the space they occupy is described in terms of their orientation, starting location, and units of measure.

The positioning of the graphic characters on a line is accomplished by moving the presentation position. Graphic characters may be placed adjacent to one another or positioned anywhere in the object space through the use of control sequences defined by PTOCA. Control sequences have been defined to move the presentation position to another position, to move to the beginning of another line, to adjust the distance between two adjacent characters, to draw lines such as rules, to adjust the distance between lines, to change the font, to specify the color of characters and rules, to overstrike a text field with a specified character, and to underscore a text field.

National Language Support (NLS) is handled in the level of formatting above the Presentation Text object. Font NLS support is provided in the font mapping function in the controlling environment.

Chapter 3. Overview of PTOCA

This chapter:

- Summarizes the concepts that form the basis of PTOCA
- Summarizes the data structures in PTOCA

General Concepts

The Presentation Text object is a description of text for a portion of a document that has been generated from one of many possible sources. Examples of possible sources are:

- Formatting from revisable text
- Transformation from other data streams
- Editor or formatting process
- Direct generation process

Once created, the object can be presented, revised, or used in a resource such as an overlay. It occupies a given amount of space, the object space, and can be located and oriented in a physical area, the object area. The environment that carries the object is the provider of all external relationships for the object, including the object area.

The object space consists of an array or matrix of *addressable positions* which identify potential locations at which to place the basic elements of the object, *graphic characters*. Graphic characters are placed at addressable positions called the *presentation positions*, *rotated* relative to a *baseline*, and have the baseline of a group of characters undergo *orientation* to various angular positions, such as vertical presentation. These positioning functions are specified by *control sequences* which are carried along with the graphic characters.

The initial positions or beginning values for many of the control sequences are described in a descriptor.

Data Stream Environment

The Presentation Text object is designed to be carried by and become part of a data stream, called the *controlling environment.* The data stream defines rules by which the object can be carried. Further information about data streams can be found in Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161.

Data Structures

The *Presentation Text Data Descriptor* carries the size, shape, and other special information about the object. The data stream is responsible for providing the proper information to the receiver, but PTOCA specifies a hierarchical method for determining the default values to be used by the receiver if the data stream does not supply the requisite information.

The *Presentation Text data* contains the code points that identify the graphic characters and the control sequences that specify where and how the graphic characters are to be positioned within the object space. The graphic character code points that represent text information can be specified in a Transparent Data (TRN), a Repeat String (RPS) , or a Unicode Complex Text (UCT) control sequence, or they can be specified as free-standing code points that appear between control sequences. Graphic character code points can also be resolved to glyph IDs in a font. These glyph IDs are carried in Glyph Layout Control (GLC) chains for presentation.

Further information about PTOCA data structures is found under "Presentation Text Data" on page 20 and "Presentation Text Data Descriptor" on page 28.

Subsets of Function

The control sequences represent the functional capabilities provided by the Presentation Text object. Since receivers of the object might not all have equivalent capabilities, it is convenient to create subsets, also called subset levels, of the total function that is available. The *base* is a set of functions required in any environment, including the ability to interpret and validate the control sequences and parameters, and to detect and report exception conditions that are within the PTOCA subsets. The first subset of function, *PT1,* includes a set of relatively primitive control sequences that a receiver is expected to support. A second subset, *PT2,* includes all of the PT1 subset plus new control sequences for underscore, overstrike, superscripts and subscripts. A third subset, *PT3*, includes all of the PT2 subset plus a new control sequence to enable spot (highlight) colors and process colors for text and rules. A fourth subset, PT4, includes new control sequences to support the rendering of complex text.

The intent of subsets is to reduce the number of combinations of supported controls so that interchange between host processors is manageable. For further information about subsets, see Chapter 6, "Compliance with PTOCA", on page 145, Appendix A, "MO:DCA Environment", on page 155, and Appendix B, "IPDS Environment", on page 161.

Spatial Concepts

Object Space

The Presentation Text object space defines the presentation space into which the presented text characters will fit. The object space is the matrix of addressable positions which are available to the generating process that defined it. This space has no relationship to the physical medium or printed page until it is placed in an object area by a composition process as part of the creation of a page or overlay. The Presentation Text object has no concept of pages, although the composition process may create an entire page from one object.

Positioning of the object space within the object area is accomplished by a mapping within the controlling environment. The object area is the boundary for text presentation by a receiver, and the controlling environment specifies the error recovery action that must occur if any portion of a character or rule violates the object area boundary. The object space is the boundary for the text positioned for presentation.

Coordinate Systems

The Presentation Text object uses two orthogonal coordinate systems. One, the X_p,Y_p coordinate system, simulates the reader's view of the object space. The other is the I,B coordinate system, which indicates the direction of the addition of characters to form words and lines, and the direction of the addition of subsequent lines.

The X_p,Y_p coordinate is an orthogonal coordinate system based on the fourth quadrant of a standard Cartesian coordinate system. Both the X_p axis and the Y_p axis specify positive values, which is a difference from the Cartesian system where the Y axis in the fourth quadrant specifies negative values. The origin of the coordinate system is in the upper left corner; the positive X_p-axis is from left-to-right, and the positive Y_p-axis is from top-to-bottom. The frame of reference for the X_p,Y_p coordinate system is provided by the environment's coordinate system for the object area into which the object space is placed. The location of the X_p,Y_p coordinate system origin is specified as an offset from the object area's coordinate system origin.

The X_p,Y_p coordinate system describes the boundary of the object space, which is a rectangle with sides equal to the extent along each axis. That is, the X_p-extent is the length along the X_p-axis, and the Y_p-extent is the

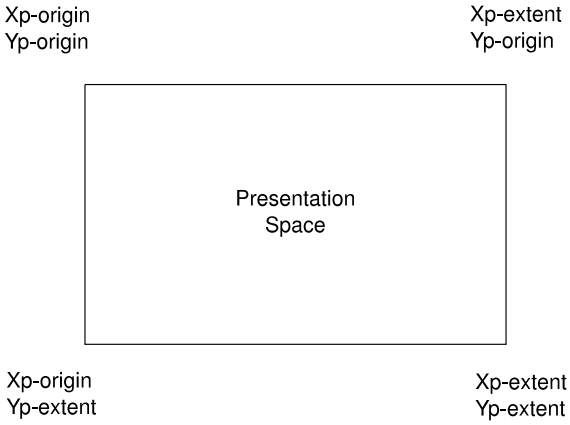
length along the Y_p-axis. Thus the object space is bounded by a rectangle described by the following four coordinate pairs:

- X_p-origin,Y_p-origin
- X_p-extent,Y_p-origin
- X_p-extent,Y_p-extent
- X_p-origin,Y_p-extent

Please see Figure 4.

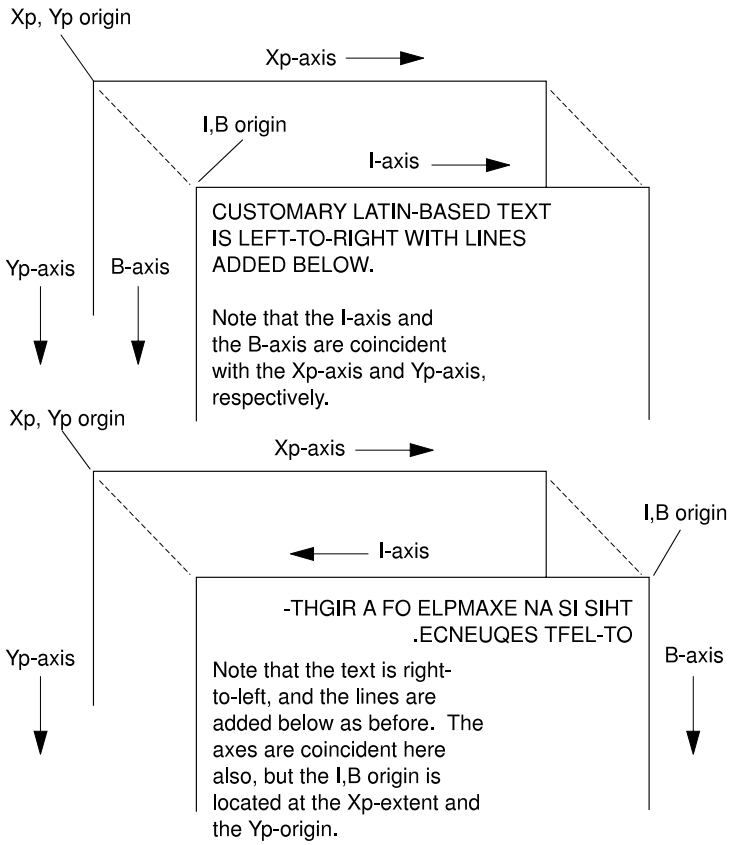
The X_p,Y_p coordinate system and the I,B coordinate system are closely related, as indicated in Figure 5 on page 12. In fact, the X_p-extent is equal to one of the I,B coordinate extents, either the I-extent or the B-extent, and the Y_p-extent is equal to the other. Therefore, the angle between the I-axis and B-axis will be identical to the angle between the X_p-axis and the Y_p-axis. The X_p,Y_p coordinate system describes the spatial viewport for the reader, while the I,B coordinate system describes the directions to be used for presentation and for interpretation by the reader of the graphic characters being presented.

Figure 4. Presentation Space Definition



The I,B coordinate system adds a concept of direction to the object space definition. The reader of text comprehends the text by assembling the characters into words or phrases. The direction in which the reader normally constructs the words or phrases is called the inline direction or I-direction. The inline direction for typical Latin-based text is left-to-right, but for languages such as Japanese, or tasks such as labeling graphs, the inline direction may be top-to-bottom or one of the other possible directions. Please see Figure 5.

Figure 5. I,B Coordinate System Examples



The inline direction is also the direction of increasing positive values of *i* along the I-axis, and prescribes the order in which succeeding characters are processed by a receiver. The maximum value of *i* is the I-extent.

All of the graphic characters placed in the inline direction for a given value of *b* constitute a line. The direction in which successive lines are placed for continued reading of the text is the baseline direction or B-direction. The baseline direction for typical Latin-based text is top-to-bottom, but for other languages, such as Japanese vertical writing, the baseline direction is from right-to-left. The baseline direction is also the direction of increasing positive values of *b* along the B-axis. The maximum value of *b* is the B-extent.

Measurement

Although the controlling environment, as a carrier of the Presentation Text object, specifies the layout characteristics of the object presentation, the object, as a self-defining portion, provides the measurement units used by the generator in formatting the data. The Presentation Text object provides for both the English and metric systems of measurement. The measurement units for the object are specified in the Presentation Text Data Descriptor or determined by defaults. Measurement units can be specified so that the X_p-axis and the Y_p-axis have different resolutions.

Measurement units are used throughout PTOCA 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 32,767

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

For example, if the measurement base is 10 inches and the units per unit base value is 5,000, the units of measure are 10 inches / 5000 or one five-hundredth of an inch. Here are some additional examples:

Units/inch	Comments
1,440 X 1,440 units/inch	14,400 divisions in 10 inches on both axes
80 X 77 units/centimeter	800 divisions in 10 centimeters on X_p and 770 divisions in 10 centimeters on Y_p

The size of the object space is specified in measurement units. Each addressable position is one measurement unit away from another addressable position in any direction. That is, a specified measurement unit along the X_p-axis separates the addressable positions in the direction parallel to the X_p-axis, and another specified measurement unit along the Y_p-axis separates the addressable positions in the direction parallel to the Y_p-axis. This creates an array of addressable positions, each of which has the potential of being designated as the position of a graphic character.

The measurement units thus defined become the measurement units for all linear measurements within the object. The receivers must convert from these measurement units to measurement units for their environment as required, and keep track of rounding errors, making appropriate adjustments as needed to ensure presentation fidelity at a given level of capability.

The measurement units for angular dimensions are degrees.

Font Concepts

When a PTOCA receiver detects a graphic character code point, the code point must be translated into a pattern of marks on some medium. A single-byte or multi-byte code point is used to identify the graphic character which is to be presented. Before presentation can take place, several attributes of the graphic character must be determined, such as the following:

- What character is represented by the code point?
- Is the character valid?
- What is the shape of the character?

The assignment of code points to characters is done by means of a *code page* or similar encoding structure such as a *character map* . A code page or character map can be envisioned as a table which contains pairs of values, where the first element of each pair is the code point and the second element is the identifier of the graphic character. The code page also defines the number of bytes required to represent a character, that is, bytes per code point.

For some font technologies such as the FOCA font technology, the validity of a character may be verified by referring to a *graphic character set*. A graphic character set is a set of letters, digits, punctuation marks, arithmetic operators, chemical symbols, or other symbols. If the character represented by the code point is not

contained in the graphic character set, then that character is invalid, and another graphic character must be substituted for it. The active coded font designates what graphic character should be substituted in its place.

The shape or graphic pattern of the character is determined from the related font. A font consists of an algorithm for presenting all the graphic characters that have a given style, size, weight and certain other characteristics. Here are examples:

Style: Bodoni
Size: 10-point
Weight: bold
Other characteristics: italic

This algorithm could consist of a style manual, raster patterns, vector graphic command lists, stroke generation programs, engraved type, or other means of specifying the necessary attributes. The font also specifies the character increment or escapement, that is, the width of the character, and the character reference point or character origin, that is, the point within the graphic pattern which is to be aligned with the presentation position. Within a Presentation Text object, the desired characteristics are specified through a reference to a specific font. The coded font contains the encoding and the shape and metric information which are assigned to each graphic character. The presentation process applies the graphic character code points found within the Presentation Text object to the active coded font in order to determine the presentation characteristics of the characters. The font is managed as a *font resource* in the controlling environment. A Presentation Text object uses this resource by making reference to the coded font.

The structure and content of FOCA-based fonts is defined by the Font Object Content Architecture (FOCA), which is described in *Font Object Content Architecture Reference,* AFPC-0007.

The structure and content of TrueType and OpenType fonts are described in the following documents available from the Microsoft® and Apple® web sites:

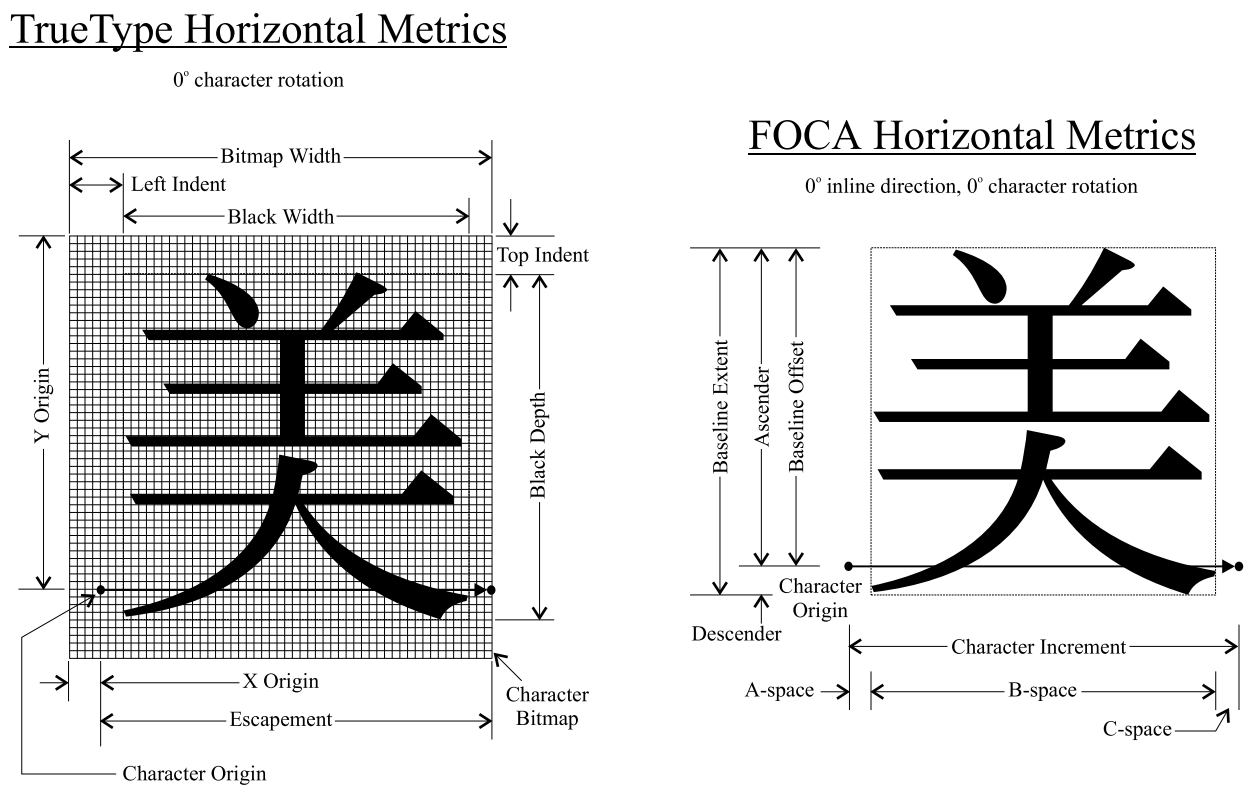
- *OpenType Specification Version 1.4* (Microsoft Corporation: October 11, 2002), at `http://www.microsoft.com/typography/otspec/default.htm`
- *TrueType Reference Manual* (Apple Computer, Inc.: December 18, 2002), at `http://developer.apple.com/fonts/TrueType-Reference-Manual`

Graphic Character Placement Concepts

Graphic characters are the basic elements of the Presentation Text object. The control sequences defined by PTOCA deal with the presentation of these graphic characters regarding either their positioning within the object, or some attribute of their presentation, such as color.

PTOCA assumes that the graphic characters are identified by one-byte or multi-byte code points that are defined within the encoding structure for a font. Each graphic character thus identified has a defined *character reference point* or *character origin*, a *character increment* or *character escapement*, and a *character baseline* that allows them to be correctly positioned along the *baseline* in the I-direction of the Presentation Text object. Please see Figure 6.

Figure 6. Horizontal Metrics: TrueType/OpenType Fonts and FOCA Fonts



The presentation of a graphic character is accomplished by placing the character reference point or character origin of the graphic character at the *presentation position*. The presentation position is an I,B coordinate pair, that is, an addressable position in the object space. The *b* value is fixed for the current baseline, B_c. The current *i* value, the new presentation position, is calculated from the previous *i* value by adding the character increment or character escapement of the graphic character being presented to the previous value of *i,* that is, the previous presentation position.

The presentation position in PTOCA designates a *between-the-pels* position on a presentation surface, not a pel centerline intersection position. The concept of between-the-pels positioning is especially important for the presentation of rules. Please see "Graphic Character Processing" on page 38 for more information.

Object generators will determine which characters are to be placed on each line of the object. This does not require that the font be known at generation time in all cases. For fixed pitch fonts where the character increment is a constant value and for fonts utilizing standard metrics, it is possible for any font with the same metrics to be specified without modification to the relative positioning of the graphic characters.

Spacing between the characters can be modified by an adjustment, which is either an increment or a decrement on the character increment values provided for the graphic characters. In addition, the character increment specified for the space code point may be changed to a different value at any time to provide variation in the spacing between words.

Lines of graphic characters are ended by moving the presentation position to the beginning of the next line. This may be done using the positioning control sequences or through the use of a control sequence that causes the baseline increment value and the inline margin to set the presentation position to the next line.

PTOCA is intended to be precise enough to permit multiple products to reproduce the Presentation Text object faithfully. Faithful reproduction includes such aspects as the size and relative positions of graphic characters and strings of graphic characters. The responsibility for faithful reproduction belongs to the process that presents the object. PTOCA is also designed to permit less than faithful reproduction. It is possible to specify *exception conditions* for which continuation of processing is acceptable. This permits a process that cannot faithfully reproduce the object to continue with its best approximation. If less than faithful reproduction is acceptable for an application, interchange among a larger set of receivers is possible.

Chaining Concepts

The Presentation Text object uses a control sequence to indicate that a function is to be performed. The control sequence consists of the *Control Sequence Introducer* and a list of parameters.

A Control Sequence Introducer contains the following fields:
- A one-byte prefix, X'2B'
- A one-byte class, X'D3'
- A one-byte length
- A one-byte function type

Control sequences can be *chained* together using a chaining convention. Although the first control sequence in a chain has the prefix and class, the remaining chained control sequences do not. Chaining reduces the number of bytes to be handled and removes the need to determine whether the next character is a control sequence or not. Please see Table 5 on page 24 for a list of PTOCA control sequences, showing both unchained and chained function types. Please see “Control Sequence Chaining” on page 34 for more information about chaining.

Character String Concepts

Graphic characters may be grouped together as character strings to eliminate the necessity of checking for the Control Sequence Prefix. This capability is useful for creating strings of repeated characters. An example is the leader dots in a table of contents. The leader dot graphic character occurs only once per line in the object although it is repeated many times at presentation.

In addition this capability, when used in conjunction with chaining, allows the object to be described in terms of two parsing modes: control sequences and graphic characters. These two basic modes can then be optimized separately in an implementation.

Ruled Line Concepts

Simple line graphic functions have been incorporated to satisfy requirements for figure enclosures, tables, boxes, line drawings, and so on. The capability includes vertical and horizontal rules which may have both the length and the width of the rules specified.

Suppression Concepts

An ability to restrict the presentation of the graphic characters in a controlled way is provided by the *suppression* function. Suppression is accomplished by marking the text data to be suppressed and specifying

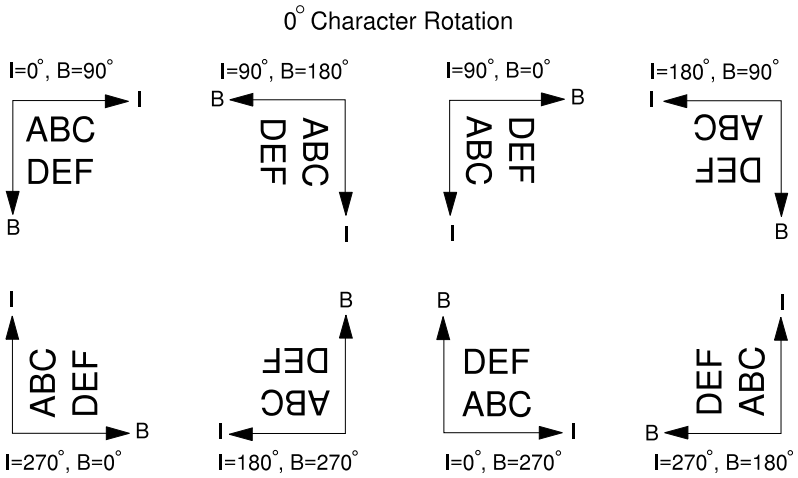
an identifier to allow grouping of the marked text data. All data marked with an active suppression identifier is prevented from being presented when the object is processed. The controlling environment specifies which suppression identifiers are active for the object.

Suppression can be used to create form letters that have portions of the form left blank, or filled in differently, depending on the intended audience of each instance of the letter.

Orientation and Rotation Concepts

There are times when it is desirable to place graphic characters in other than the customary upright reading position. For example, when labeling a graph, the graphic characters would be placed upright, but the line would be vertical; that is, the I-direction would be top-to-bottom. The I-direction and B-direction determine the *orientation* of the text, and an I-direction change is called a change of orientation. However, since the upright position is with respect to the I-axis, when reorientation occurs the characters appear to *rotate* at the same time. To create a vertical effect, such as in a graph, the graphic characters must also be rotated. Please see Figure 7. This figure illustrates changes in orientation with no change in character rotation.

Figure 7. Orientation Examples



Orientation is specified in degrees in a clockwise direction from the zero-degree starting point. The zero-degree starting point is the I-axis when the I-direction is left-to-right. A change in text orientation may also move the I,B origin to a different corner of the text object space. Figure 7 shows the location of the I,B origin for the 8 text orientations. The rotation of the characters is described in terms of angular movement of the character shape with respect to the character baseline, and is specified as part of the selection criteria for fonts.

Extended Functions Concepts

Controls are provided in PTOCA to accomplish specialized functions. These functions include underscore, overstrike, superscript, and subscript.

This group of control sequences follows a modal concept in that, once started, the function does not terminate until stopped. Each control sequence marks the beginning or the ending of a *text field* for which the function is invoked. The same control sequence syntax with a non-zero parameter value begins the text field, and with a zero parameter value indicates the end of the field. All other control sequences are valid within these text fields without causing termination of the field.

Underscore is the capability of drawing a line under individual characters or groups of characters. Overstrike is the capability of filling a field with a specific character to provide a marked-out appearance.

The superscript and subscript functions require the ability to move temporarily from the designated baseline by small amounts. The superscript function requires movement in the negative B-direction, that is, above the baseline. The subscript function requires movement in the positive B-direction, that is, below the baseline. The amount of the incremental moves about the baseline is also variable. This allows a sophisticated implementation to provide a wide range of superscript and subscript capability, to be used, for example, when positioning the various parts of mathematical equations.

Complex Text Processing Overview

The Unicode standard recommends that text for all languages be stored in the order that it would be read or spoken, without regard to presentation order. With few exceptions, Latin, Cyrillic, and Greek scripts present text in the same order that data processing systems store the text. These exceptions are ligatures which are combinations of characters and accented characters. Traditionally, computer applications encode these combined characters using one encoding point and one graphic character. As an example, most systems encode Latin small ligature ff as one character.

Complex text languages provide different layouts for the presentation of text and its storage. Bi-directional (BIDI) languages present text normally from right to left; however, some text such as numbers and embedded Latin, Cyrillic, and Greek scripts, are written from left to right. These languages include Arabic, Urdu, Farsi, and Hebrew.

The following example nests Western and bi-directional scripts. Lowercase characters represent Western Script and uppercase characters represent a bi-directional script:

- Data Storage order: my address is SUITE B 100 YORK BLVD richmond hill
- Presentation order: my address is DVLB KROY 100 B ETIUS richmond hill

Some languages require that characters be presented with different shapes or in a different order than their storage order. Arabic characters can be represented by one of four shapes depending on their position in a word. Arabic characters can also combine to form ligatures. In many south Asian languages, characters may need to be repositioned, reordered, or split, depending on adjacent characters.

Composition applications that need to present Complex Text will use layout engines such as International Components for Unicode (ICU), Windows® Uniscribe, Apple Advanced Typography, Pango, Scribe, or Graphite, to present text. Each engine has a different implementation. Outputs from the engines will differ somewhat. Some engines have better support for some language scripts than others.

PTOCA supports consistent text presentation through the Unicode Complex Text (UCT) control sequence and its complementary supporting glyph run control sequences. PTOCA presents text on a line-by-line basis. This means that applications must provide text boundary analysis. ICU provides iterators that support this type of analysis. These break iterators support determining character, word, line-break, and sentence boundaries. The Unicode Standard Annex #29 provides definitions for these boundaries. The ICU User Guide provides examples of boundary analysis. The Unicode BIDI algorithm works best on paragraphs, so the composition application should apply the algorithm before breaking the text into individual lines.

The ICU Layout Engine supports the presentation of complex text through a common client interface. This uniform base interface models a TrueType/OpenType font at a particular point size and device resolution. TrueType/OpenType fonts have the following characteristics:

- A font is a collection of images called glyphs. Each glyph in the font has a unique 16-bit id.
- There is a mapping from Unicode code points to glyph ids. Some glyphs may not have a mapping.
- There is a method to get the width of each glyph.
- There is a method to get the position of a control point for each glyph

Client applications perform boundary analysis and determine text direction runs as necessary. They then call the layout engine to produce an array of glyph indices in display order, an array of x, y position pairs for each glyph, and optionally an array that maps each glyph back to the input text array.

The MO:DCA Presentation Text Data Descriptor and Presentation Text Data structured fields carry Presentation Text Objects in MO:DCA documents. The Presentation Text object space provides the coordinate system that allows object generators to position graphic characters and glyphs without error. It is the responsibility of the generator to ensure that it positions the graphic characters and glyphs correctly so that they do not exceed the object space.

The general approach composition applications will take to present complex text data is:

- If the data contains bi-directional scripts, use the Unicode BIDI algorithm to break the text into directional sequences. The Unicode BIDI algorithm works best with paragraphs, so it must be applied before text is broken into separate lines.
- If multiple TrueType/OpenType fonts are used to present the text, composition applications must identify the individual font that will be used for each substring of text to which the layout engine is applied. This step should be performed prior to, or concurrent with, the script analysis that identifies the appropriate layout engine. The ICU Paragraph Layout API provides class interfaces to represent linked fonts with methods to request the individual font and the extent of the text string to be composed. If the entire Unicode text is not rendered with a single font, the subsequent steps must be repeated for each font and substring.
- The application will need to determine where line breaks occur because PTOCA constrains text output sequences to individual lines. The appropriate position to break text can vary by language or script. The Unicode Standard Annex #29 provides definitions for character, word, line-break, and sentence boundaries. International Components for Unicode provides break iterators that support this type of analysis. The ICU User Guide provides examples of boundary analysis.
- The application will use a layout engine to format text sequences. A text sequence is a run of text that use the same font (e.g. typeface with the same typographic attributes including weight, width, height, posture) where the text accumulates in the same direction. Layout engines normally return:
  - Arrays of glyph indices
  - Arrays of glyph positions

  These arrays provide the information required to present the glyphs.
- The application will then normalize the positions with respect to the origin established for the current text object.
- The application will obtain or calculate the Object Identifier (OID) value of the TrueType/OpenType font used to generate the glyph ID values. This value allows presentation devices to verify that they retrieve the glyphs from the exact same font that the application used. See the *Mixed Object Document Content Architecture Reference,* AFPC-0004 for the definition of the algorithm used to calculate the OID of a TrueType/OpenType font.
- The application will provide the full font name (FFN) of the TrueType/OpenType font used to generate the glyph ID values. This name provides a human-readable identification of the font and is also used to select a specific font in a font collection when the font OID identifies a collection.

Successful completion of these tasks will result in the application having the presentation data normalized so that it can create the GIR/GAR[/GOR] sequences and the preceding GLC control (the notation “[/GOR]” will be

used to indicate that the GOR is optional). This set of controls is called a GLC chain. The generator creates the following control sequences:

- A Glyph Layout Control (GLC) which identifies the start of the complex text position requirements sequence for this text run. The GLC specifies:
  1. the advance along the current baseline caused by processing this GLC chain
  2. the font OID to identify and validate the font used for the subsequent glyph run control sequences
  3. the full font name of the font, which is used to select a specific font from a font collection when the font OID identifies a collection
- One or more groups of:
  1. a Glyph ID Run (GIR) which contains the glyph IDs for this text run
  2. a Glyph Advance Run (GAR) which contains the advances in the inline direction for each glyph
  3. an optional Glyph Offset Run (GOR) that provides the offsets of each glyph from the baseline. This control provides the ability to position glyphs such as diacritical marks and accents
- An optional Unicode Complex Text (UCT) control sequence which contains the text encoded in UTF16-BE in data storage order. Use of the UCT is recommended as it provides applications the ability to interpret the sequence of glyphs rendered by the printer.

The maximum length of a PTOCA control sequence limits a single GIR to no more than 125 glyphs. This means that print applications must be prepared to generate multiple GIR/GAR[/GOR] groupings to support long Unicode encoded text strings.

If multiple fonts linked to the currently active font are used to render the text, a GLC chain must be generated for each substring that uses a different linked font. The presentation device will use the FONTOID parameter of the GLC to identify and validate the linked font used for the subsequent glyph run control sequences. If the FONTOID parameter identifies a font collection, the presentation device uses the FFONTNME parameter of the GLC to select the specific font from the collection.

Exception Handling Concepts

PTOCA defines *exception condition codes* that identify the various exception conditions that can arise during the processing of a Presentation Text object. These codes are provided for reference purposes only. PTOCA also specifies a *standard action* for each exception condition that indicates the recommended action a processor should take when it encounters the exception condition. The manner in which a PTOCA receiver processes exception conditions depends upon the controlling environment. For any PTOCA exception condition the controlling environment may provide its own identifier, which overrides the PTOCA exception condition code. The controlling environment also may provide its own action, which overrides the PTOCA standard action.

Presentation Text Data

Presentation Text data contains the graphic characters and the control sequences necessary to position the characters within the object space. The data consists of:

- graphic characters to be presented
- control sequences that position them
- modal control sequences that adjust the positions by small amounts
- other functions which cause the text output to be presented with differences in appearance

The graphic characters are expected to conform to a font representation so that they can be translated from the code point in the object data to the character in the font. The units of measure for linear displacements are derived from the Presentation Text Data Descriptor or from the hierarchical defaults.

Control Sequence Summary Descriptions

The following pages contain summary descriptions of the PTOCA control sequences. Summary tables are provided following the descriptions. Please see "Control Sequence Detailed Descriptions" on page 45 for detailed descriptions of syntax, semantics, and pragmatics.

Absolute Move Baseline (AMB)

Establishes the baseline and the current presentation position at a new B-axis coordinate, B_{cnew}, which is a specified number of measurement units from the I-axis. There is no change to the current I-axis coordinate, I_c.

Absolute Move Inline (AMI)

Establishes the current presentation position on the baseline at a new I-axis coordinate, I_{cnew}, which is a specified number of measurement units from the B-axis. There is no change to the current B-axis coordinate, B_c.

Begin Line (BLN)

Establishes the current presentation position on the baseline with the new I-axis coordinate, I_{cnew}, equal to the inline margin, and the new B-axis coordinate, B_{cnew}, increased by the amount of the baseline increment from B_c. The baseline increment is established by the Set Baseline Increment control sequence.

Begin Suppression (BSU)

Marks the beginning of a field of presentation text, identified by a local identifier (LID), which is not to be presented when the LID is activated in the controlling environment. This control sequence does not alter the effects of other control sequences within it, except that graphic characters and rules are not presented. Suppression of presentation text by more than one control sequence at a time is not allowed; that is, nesting of suppression control sequences is not allowed.

Draw B-axis Rule (DBR)

Draws a line of specified length and specified width in the B-direction from the current presentation position. The location of the current presentation position is unchanged.

Draw I-axis Rule (DIR)

Draws a line of specified length and specified width in the I-direction from the current presentation position. The location of the current presentation position is unchanged.

End Suppression (ESU)

Marks the end of a field of presentation text, identified by a LID, which is not to be presented when the LID is activated by the controlling environment.

Glyph Advance Run (GAR)

Specifies the displacement of glyphs along the current baseline.

Glyph ID Run (GIR)

Specifies the IDs of glyphs to be placed along the current baseline.

Glyph Layout Control (GLC)

Specifies control information for the start of one or more glyph runs along the current baseline.

Glyph Offset Run (GOR)

Specifies offsets of glyphs above or below the current baseline.

No Operation (NOP)

Specifies a string of bytes that are to be ignored.

Overstrike (OVS)

Specifies a text field that is to be overstruck with a specified graphic character. The overstrike function is initiated by an OVS control sequence with a non-zero bypass identifier, and is terminated by an OVS control sequence with a zero-value bypass identifier. The fields may not be nested or overlapped. The bypass identifier controls which portions of a line are to be overstruck; this provides for bypassing white space created by AMI, RMI, and space characters.

Relative Move Baseline (RMB)

Establishes the presentation position on the baseline at a new B-axis coordinate, B_{cnew}, which is a specified number of measurement units from the current B-axis coordinate, B_c. There is no change to the current I-axis coordinate, I_c.

Relative Move Inline (RMI)

Establishes the presentation position on the baseline at a new I-axis coordinate, I_{cnew}, which is a specified number of measurement units from the current I-axis position, I_c. There is no change to the current B-axis coordinate, B_c.

Repeat String (RPS)

Specifies a string of characters that are to be repeated until the number of bytes in the graphic characters presented is equal to a specified number of bytes. The string is not checked for control sequences. When the specified number of bytes is equal to the number of bytes in the characters in the data parameter, this control sequence is identical in function to the Transparent Data control sequence.

Set Baseline Increment (SBI)

Specifies the value of the increment to be added to the B-axis coordinate of the current presentation position, B_c, when a Begin Line control sequence is processed.

Set Coded Font Local (SCFL)

Specifies a LID to be used as an index into the font map of the controlling environment to determine which coded font, character rotation, and font modification parameters have been selected for use in the object.

Set Extended Text Color (SEC)

Specifies a color value and defines the color space and encoding for that value. Supports spot (highlight) colors and process colors. The specified color value is applied to foreground areas of the text presentation space, that is, characters, rules, and underscores.

Set Inline Margin (SIM)

Specifies the value to be used as the new I-axis coordinate, I_{cnew}, of the new presentation position after a Begin Line control sequence is processed. The new presentation position is the addressable position nearest to the B-axis at which the character reference point of a graphic character may be placed.

Set Intercharacter Adjustment (SIA)

Specifies the increment to be added to or subtracted from the I-axis coordinate of the current presentation position, I_c. The direction parameter indicates whether to add or subtract the increment. If the direction is positive, the increment is added; if negative, the increment is subtracted. This control sequence may be used to compress or expand words for emphasis, improved appearance, or justification.

Set Text Color (STC)

Specifies a named color value to be applied to foreground areas of the text presentation space, that is, characters, rules, and underscores. The values of the foreground color parameter serve as indexes into the color-value table found in Table 14 on page 94.

Set Text Orientation (STO)

Establishes the positive I-axis orientation as an angular displacement from the X_p-axis, determining the I-direction. This control sequence also establishes the positive B-axis orientation as an angular displacement from the X_p-axis, determining the B-direction.

The I-axis must be parallel to one of the X_p,Y_p coordinate axes and the B-axis must be parallel to the other. The determination of the orientation and direction of the I-axis and B-axis places the origin of the I,B coordinate system at one of the corners of the rectangular object space.

Set Variable Space Character Increment (SVI)

Specifies the increment to be used as the character increment for the character identified as the Variable Space Character by the font or by the controlling environment. This increment is added to the I-axis coordinate of the current presentation position, I_c, when the Variable Space Character code point is processed in order to establish the new presentation position. This has no effect on the B-axis coordinate value.

Temporary Baseline Move (TBM)

Specifies a temporary movement of the current baseline away from the established baseline. The established baseline B-axis coordinate is maintained until a Temporary Baseline Move control sequence occurs. Temporary moves are made by the amount of the temporary baseline increment in one of three ways:

Above	Direction parameter = 3
Below	Direction parameter = 2
Back to the established baseline	Direction parameter = 1

The temporary baseline function is terminated by a TBM control sequence which returns the temporary baseline to the same B-axis coordinate as that of the established baseline.

Transparent Data (TRN)

Specifies a string of characters that are to be presented, but not checked for control sequences.

Underscore (USC)

Specifies a text field that is to be underscored. The underscore function is initiated by an Underscore control sequence with a non-zero bypass identifier, and is terminated by a USC control sequence with a bypass identifier of zero. The fields may not be nested or overlapped. The bypass identifier controls which portions of a line are to be underscored; this provides for bypassing white space created by AMI, RMI, and space characters.

Unicode Complex Text (UCT)

Identifies a sequence of Unicode code points that can be processed as Unicode complex text. The sequence starts with the first byte following the end of the UCT control sequence and ends with the last byte identified by the complex text length parameter in the control sequence. Rendering complex text involves *bidirectional (bidi) layout processing* and *glyph processing*.

Table 5. Summary of PTOCA Control Sequences

Control Sequence Name	Function Type Unchained	Function Type Chained
Inline Controls		
"Set Inline Margin (SIM)" on page 92	X'C0'	X'C1'
"Set Intercharacter Adjustment (SIA)" on page 89	X'C2'	X'C3'
"Set Variable Space Character Increment (SVI)" on page 99	X'C4'	X'C5'
"Absolute Move Inline (AMI)" on page 49	X'C6'	X'C7'
"Relative Move Inline (RMI)" on page 75	X'C8'	X'C9'
Baseline Controls		
"Set Baseline Increment (SBI)" on page 79	X'D0'	X'D1'
"Absolute Move Baseline (AMB)" on page 47	X'D2'	X'D3'
"Relative Move Baseline (RMB)" on page 73	X'D4'	X'D5'
"Begin Line (BLN)" on page 51	X'D8'	X'D9'
"Set Text Orientation (STO)" on page 96	X'F6'	X'F7'
Controls for Data Strings		
"Unicode Complex Text (UCT)" on page 113	X'6A'	
"Glyph Layout Control (GLC)" on page 61		X'6D'
"Glyph ID Run (GIR)" on page 60		X'8B'
"Glyph Advance Run (GAR)" on page 59	X'8C'	X'8D'
"Glyph Offset Run (GOR)" on page 66	X'8E'	X'8F'
"Transparent Data (TRN)" on page 106	X'DA'	X'DB'
"Repeat String (RPS)" on page 77	X'EE'	X'EF'
"No Operation (NOP)" on page 68	X'F8'	X'F9'

Table 5 Summary of PTOCA Control Sequences (cont'd.)

Control Sequence Name	Function Type Unchained	Function Type Chained
Controls for Rules		
"Draw I-axis Rule (DIR)" on page 56	X'E4'	X'E5'
"Draw B-axis Rule (DBR)" on page 54	X'E6'	X'E7'
Character Controls		
"Set Text Color (STC)" on page 93	X'74'	X'75'
"Set Extended Text Color (SEC)" on page 83	X'80'	X'81'
"Set Coded Font Local (SCFL)" on page 81	X'F0'	X'F1'
"Begin Suppression (BSU)" on page 52	X'F2'	X'F3'
"End Suppression (ESU)" on page 58	X'F4'	X'F5'
Field Controls		
"Overstrike (OVS)" on page 69	X'72'	X'73'
"Underscore (USC)" on page 108	X'76'	X'77'
"Temporary Baseline Move (TBM)" on page 101	X'78'	X'79'

Table 6. Explanation of Symbols Used in Tables

Symbol	Meaning
I_c	Current inline addressable position
B_c	Current baseline addressable position
I_{cnew}	New current inline addressable position
B_{cnew}	New current baseline addressable position
I_o	Inline addressable position at origin
B_o	Baseline addressable position at origin
I_h	Initial I-axis coordinate established by data stream
B_h	Initial B-axis coordinate established by data stream
I_{margin}	I-axis coordinate at left margin
B_{est}	Established B-axis coordinate
CI	Character increment specified by font resource
ADJSTMNT	Intercharacter adjustment (increment or decrement)
VSI	Variable Space Character increment
CHAR	Any character with CI > 0 (non-null character)
NULLCHAR	Any character with CI = 0 (null character)

Table 7. Summary of Directive Control Sequences

Control Sequence Name	Control Sequence Mnemonic	Parameter	Effect
Absolute Move Baseline	AMB	DSPLCMNT	$B_{cnew} = B_o$ + DSPLCMNT
Absolute Move Inline	AMI	DSPLCMNT	$I_{cnew} = I_o$ + DSPLCMNT
Begin Line	BLN	None	$I_{cnew} = I_{margin}$ $B_{cnew} = B_c$ + INCRMENT $B_{est} = B_{est}$ + INCRMENT
Begin Suppression	BSU	LID	Do not present text following this control through next ESU with same LID, if LID is active at controlling environment level.
Draw B- Axis Rule	DBR	RLENGTH RWIDTH	Draw rule in B-direction from B_c to B_c + RLENGTH. Rule width = RWIDTH. I_c and B_c are unchanged.
Draw I- Axis Rule	DIR	RLENGTH RWIDTH	Draw rule in I-direction from I_c to I_c + RLENGTH. Rule width = RWIDTH. I_c and B_c are unchanged.
End Suppression	ESU	LID	End suppression of characters if same LID as preceding BSU.
Glyph Advance Run	GAR	ADVANCE	Specifies the displacement of glyphs along the current baseline.

Table 7 Summary of Directive Control Sequences (cont'd.)

Control Sequence Name	Control Sequence Mnemonic	Parameter	Effect
Glyph ID Run	GIR	GLYPHID	Specifies the IDs of glyphs to be placed along the current baseline.
Glyph Layout Control	GLC	IADVNCE OIDLGTH FFNLGTH FONTOID FFONTNME	Specifies control information for the start of one or more glyph runs along the current baseline.
Glyph Offset Run	GOR	OFFSET	Specifies offsets of glyphs above or below the current baseline.
No Operation	NOP	IGNDATA	Ignore bytes IGNDATA. No change to I_c or B_c.
Relative Move Baseline	RMB	INCRMENT	$B_{cnew} = B_c$ + INCRMENT
Relative Move Inline	RMI	INCRMENT	$I_{cnew} = I_c$ + INCRMENT
Repeat String	RPS	RLENGTH RPTDATA	Repeat RPTDATA until RLENGTH bytes from RPTDATA have been presented.
Transparent Data	TRN	TRNDATA	Process all code points in TRNDATA as characters.
Unicode Complex Text	UCT	UCTVERS CTLNGTH CTFLGS BIDICT GLYPHCT ALTIPOS	Process the next CTLNGTH Unicode code points as complex text.

Table 8. Summary of Modal Control Sequences

Control Sequence Name	Control Sequence Mnemonic	Parameter	Effect
Set Baseline Increment	SBI	INCRMENT	Upon execution of BLN, $B_{cnew} = B_c$ + INCRMENT.
Set Coded Font Local	SCFL	LID	Establish active font, character rotation, and font modification parameters.
Set Extended Text Color	SEC	COLSPCE COLSIZE1 COLSIZE2 COLSIZE3 COLSIZE4 COLVALUE	Set process color and highlight color for text, rules, and underscores.
Set Intercharacter Adjustment	SIA	ADJSTMNT DIRCTION	If current character follows another character, $I_{cnew} = I_c$ +/- ADJSTMNT Present character $I_{cnew} = I_c$ + character increment DIRCTION = 0 means ADJSTMNT is positive DIRCTION = 1 means ADJSTMNT is negative

Table 8 Summary of Modal Control Sequences (cont'd.)

Control Sequence Name	Control Sequence Mnemonic	Parameter	Effect
Set Inline Margin	SIM	DSPLCMNT	Upon execution of BLN, $I_{cnew} = I_o$ + DSPLCMNT = I_{margin}
Set Text Color	STC	FRGCOLOR	Set named color for text, rules, and underscores.
Set Text Orientation	STO	IORNTION BORNTION	Establish angle of I-axis and B-axis with respect to X_p-axis.
Set Variable Space Character Increment	SVI	INCRMENT	Establish character increment of Variable Space Character.

Table 9. Summary of Field Control Sequences

Control Sequence Name	Control Sequence Mnemonic	Parameter	Effect
Overstrike	OVS	BYPSIDEN OVERCHAR	Overstrike following text with OVERCHAR. BYPSIDEN controls overstrike of white space. BYPSIDEN = 0 terminates. Baseline reference is B_c
Underscore	USC	BYPSIDEN	Underscore following text. BYPSIDEN controls underscore of white space. BYPSIDEN = 0 terminates. Baseline reference is B_{est}
Temporary Baseline Move	TBM	DIRCTION PRECSION INCRMENT	Create temporary baseline at B_{cnew}=B_c+INCRMENT B_{est} is unchanged

Presentation Text Data Descriptor

The Presentation Text Data Descriptor specifies the units of measure for the Presentation Text object space, the size of the Presentation Text object space, and the initial values for modal parameters, called *initial text conditions*. Initial values not provided are defaulted by the controlling environment or the receiving device. The Presentation Text Data Descriptor provides the following initial values:

- Unit base
- X_p-units per unit base
- Y_p-units per unit base
- X_p-extent of the presentation space
- Y_p-extent of the presentation space
- Initial text conditions

The initial text conditions are values provided by the Presentation Text Data Descriptor to initialize the modal parameters of the control sequences. Modal control sequences typically are characterized by the word *set* in the name of the control sequence. Modal parameters are identified as such in their semantic descriptions.

Initial Text Condition Summary Descriptions

The initial text conditions include the following parameters:

- Baseline increment
- Extended text color
- Coded font local ID

- Initial baseline coordinate
- Initial inline coordinate
- Inline margin
- Intercharacter adjustment
- Text color
- Text orientation

The following pages contain summary descriptions of the initial text conditions. Please refer to "Objects" on page 3 for detailed descriptions of semantics and pragmatics. Also see the corresponding control sequence, if appropriate, for additional information.

Baseline Increment

Specifies the value to be used for the increment parameter of the Set Baseline Increment control sequence. This increment represents the number of measurement units to be added to the B-axis coordinate of the current presentation position, B_c, when a Begin Line control sequence is processed. The current I-axis coordinate, I_c, is unchanged. The default value is the Default Baseline Increment associated with the default coded font of the device.

Coded Font Local ID

Specifies the value to be used as the LID in the Set Coded Font Local control sequence. This LID represents an index into a font map of the controlling environment used in the determination of which font, character rotation, and font modification parameters have been selected for use in the object. The default value is the LID of the default font of the device.

Extended Text Color

Specifies a foreground spot (highlight) color or process color to be used to present graphic characters, rules, and underscores.

Initial Baseline Coordinate

Specifies the value of the current presentation position B-axis coordinate, B_c. This value represents the displacement in the B-direction from the I-axis for the initial position for presentation of graphic characters or processing of control sequences. The default value is device-dependent.

Initial Inline Coordinate

Specifies the value of the current presentation position I-axis coordinate, I_c. This value represents the displacement in the I-direction from the B-axis for the initial position for presentation of graphic characters or processing of control sequences. The default value is zero.

Inline Margin

Specifies the value to be used for the displacement parameter of the Set Inline Margin control sequence. This value represents the I-axis coordinate of the presentation position nearest to the B-axis after a Begin Line control sequence is processed. The default value is zero.

Intercharacter Adjustment

Specifies the value to be used for the adjustment parameter of the Set Intercharacter Adjustment control sequence. This value represents the number of measurement units by which the I-axis coordinate of the current presentation position is adjusted when the SIA control sequence is processed. The direction of the adjustment is determined by the direction parameter. If the direction is positive, the adjustment is added; if

negative, the adjustment is subtracted. The default value is zero for both the adjustment parameter and the direction parameter.

Text Color

Specifies a foreground named color value to be used to present text, rules, and underscores. A foreground color parameter value represents an index into the color-value table in Table 14 on page 94. The default value is X'FF07'.

Text Orientation

Specifies the angular displacement values to be used for the I-axis orientation and the B-axis orientation parameters of the Set Text Orientation control sequence. The I-axis value represents the positive I-axis orientation as an angular displacement from the X_p-axis, and the resultant I-direction. The B-axis value represents the positive B-axis orientation as an angular displacement from the X_p-axis, and the resultant B-direction. The default value for the I-axis is X'0000', that is, zero degrees. The default value for the B-axis is X'2D00', that is, 90 degrees.

Chapter 4. Data Structures in PTOCA

This chapter:
- Describes the role of parameters
- Explains documentation conventions used in this chapter
- Provides a detailed description of the control sequence
- Explains how graphic characters are processed
- Provides a detailed description of the Presentation Text data
- Provides a detailed description of the Presentation Text Data Descriptor

Parameters and Parameter Values

Kinds of Parameters: The control sequences used within the Presentation Text object may contain *parameters* that describe and control the way the control sequence affects the graphic characters to be presented. A parameter is a variable to which a value is assigned. A parameter has an architected length, a set of values, and a functional definition. These parameter values may be numeric, such as those that tell where text is to be presented, or logical, such as those that tell whether data should be suppressed. A parameter value can be the *default indicator*, specified by the value X'F...F'. The default indicator means that the effective hierarchical value is to be used instead of a value explicitly specified at this point. A parameter can be a *reserved field*. A reserved field has a prescribed value, but no functional definition. Reserved fields should be set to zero by PTOCA generators and should be ignored by receivers.

A *mandatory parameter* appears in a control sequence because the function of that parameter is required and an actual value is necessary for proper performance. A mandatory parameter value may be the default indicator, provided that the parameter has an actual value somewhere in the hierarchy. Mandatory parameters must be supported by both PTOCA generators and receivers. In a sense, all parameters are required, since the value of each parameter must be known if it is to have an effect on presentation. However, some parameters are required only for specific types of presentation, and an actual value is not always necessary for some parameters. For example, a default value may be acceptable.

An *optional parameter* need not appear in a control sequence. The function of that parameter may not be required, or if the function is required, a default value may be acceptable instead. An optional parameter may appear if the default value is not acceptable, if it is desirable to make a value explicit for documentation, or to avoid the effect of values specified externally to the Presentation Text object. Optional parameters must be supported by all PTOCA receivers.

Hierarchy: Certain parameters, called *external parameters,* use the following hierarchical techniques in specification. If the parameter is not specified by individual control sequences in the Presentation Text object, that is, the parameter is omitted or the parameter is the default indicator, the parameter may be specified in the Presentation Text Data Descriptor. If it is not, the PTOCA default value is used. Not all parameters need to be set at all levels of the specification hierarchy. Table 10 on page 33 identifies the valid hierarchical specification levels for the parameters, indicated by *X* in the associated column. Note that the hierarchy consists of the control sequence first, then the descriptor, and finally the PTOCA default value. Thus from a receiver's point of view, the primary source for a parameter value is a control sequence. If it is possible for a control sequence to provide the value, there will be an *X* in the control sequence column. If there is no control sequence to provide the needed value, or if the appropriate control sequence is present but specifies the default indicator, the descriptor becomes the source of the value. If it is possible for the descriptor to provide the value, there will be an *X* in the descriptor column. However, if the descriptor cannot provide the value, or if the descriptor specifies the default indicator, the PTOCA default value is used.

Default values and Presentation Text Data Descriptor values are termed *external parameter specifications*, because these parameters need not be explicitly defined in the Presentation Text object. These values become current values each time presentation of a Presentation Text object begins.

Ranges of Values: Every value must fit within the field defined to contain it. Additional constraints on values are defined by the PTOCA subsets. See Chapter 6, “Compliance with PTOCA”, on page 145 and the appendixes for further information about ranges.

Negative numbers are expressed in twos-complement form. If a parameter is less than the minimum specified or more than the maximum specified, an exception condition exists. See the semantics of the affected control for the appropriate exception condition code.

The maximum absolute value of a one-byte arithmetic value is 254 when the default indicator is valid. When the default indicator is specifically excepted, the one-byte maximum arithmetic value is 255. One-byte values are always unsigned. The maximum absolute value of a signed two-byte arithmetic value is 32,768. The maximum absolute value of an unsigned two-byte arithmetic value is 65,534 when the default indicator is valid, and 65,535 when the default indicator is not valid.

The minimum requirements of PTOCA for receivers regarding range is to provide calculation capacity equal in size to the number of bits in the respective parameters. This is limited by the subset. Processing of the Presentation Text object necessitates tracking the current positions within the object space. In addition, PTOCA requires that receivers be capable of tracking the current positions outside of the object space as long as presentation is not attempted.

The following example illustrates the intent of this concept for I_c. A receiver may determine the maximum number of bits for I_c based on the physical size and resolution of the mechanism. For example, the receiver may base it on a carriage width of three inches and a resolution of 1/1440 inch. For this receiver, positioning outside the object space could be handled in the cases where the object space is smaller than the carriage width. But when the object space is equal to or greater than the carriage width, the receiver's calculation capacity may not be large enough to contain the calculations outside the object space, and the results may be unpredictable.

Such *overflow* is not considered to be an exception condition by PTOCA. However, the architectural recommendation to generators is to avoid addressable positions that are outside of the object space.

Parameter Data Types: Every parameter value is one of the following data types:

A *bit string* (BITS) is a string of bits one or more bytes long. Each bit has the value B'1' or B'0'.

A *code* (CODE) is a constant for which PTOCA defines a particular meaning.

A *number* is an unsigned (UBIN) or signed (SBIN) arithmetic value that implies count or magnitude.

A *character string* (CHAR) is one or more bytes of character information.

An *undefined field* (UNDF) is a field that is not defined by PTOCA.

In all cases bytes are composed of eight bits, called bits 0 - 7. Bit 0 is in the high-order position; that is, bit 0 = 2^7 and bit 7 = 2^0. Two bytes considered together are sixteen bits, called bits 0 - 15. Bit 0 is in the high-order position; that is, bit 0 = 2^{15}. and bit 15 = 2^0. The first byte received is the high-order byte, that is, bits 0 - 7. The second byte is the low-order byte, that is, bits 8 - 15. This is called *big-endian bit order* and *big-endian byte order*.

Different syntax is used for the expression of values that pertain to rotation.

The Default Indicator: For certain parameters, the value X'F...F' represents the default indicator. For these parameters, the arithmetic value -1 is not available. The default indicator does not specify a value for a parameter and, therefore, maintains a default value for that parameter. This indicator specifies that the default value be obtained from the hierarchy, not including previous instances of the same parameter in the current object. The syntax of each control sequence specifies whether the default indicator is valid for its parameters.

In general, the default value for an omitted optional trailing parameter is obtained from previous instances of the same parameter in the current object according to the hierarchy.

Table 10. Parameter Specification Hierarchy

Parameter	Set by Control Sequence (highest priority)	Set by Descriptor	PTOCA Default Value (lowest priority)
Measurement Units		X	Receiver dependent
Size		X	Receiver dependent
Baseline Increment	X	X	Receiver dependent, should be based on the coded font
Suppression identifier			None
Coded Font Local ID	X	X	Receiver dependent
Intercharacter Adjustment	X	X	0
Intercharacter Direction	X	X	0
Inline Margin	X	X	0
Initial Baseline Coordinate		X	Receiver dependent
Initial Inline Coordinate		X	0
Foreground Color	X	X	X'FF07'
Text Orientation	X	X	0,90
Rule Length	X		None
Rule Width	X		Receiver dependent
Overstrike Bypass Identifiers	X		X'01'
Overstrike Character	X		Coded font dependent
Temporary Baseline Increment	X		½ the Baseline Increment
Temporary Baseline Direction	X		0
Temporary Baseline Precision	X		0
Underscore Bypass Identifiers	X		X'01'
Variable Space Character Code			Coded font dependent
Variable Space Character Increment	X		Coded font dependent

Control Sequence

The Presentation Text object can specify that text functions are to be performed, such as underlining, margin setting, or justification. These functions are invoked by sequences that must begin with identifiers that distinguish them from code points. A character that delimits a string that must be processed in a different manner may be thought of as an *escape* character. In the Presentation Text object, the equivalent of an escape character is the Control Sequence Prefix. The string it delimits is a control sequence. The Presentation Text object supports only one type of control sequence.

Control Sequence Format

A control sequence contains a *Control Sequence Introducer* followed by parameters. The parameter portion of the control sequence may be from 0 to 253 bytes in length.

Introducer	$Parm_1$	$Parm_2$	$Parm_3$	...	$Parm_n$

An unchained Control Sequence Introducer contains the following fields:
- A one-byte prefix, X'2B'
- A one-byte class, X'D3'
- A one-byte length
- A one-byte function type

Prefix	Class	Length	Function Type
X'2B'	X'D3'		

The length field delimits the control sequence by indicating the last byte in the control sequence. The length field specifies the count of bytes in the control sequence, starting with itself. If the value of the length field is 2, there are no parameters in the control sequence. If the value is 3 or greater, there are parameters.

The function type uniquely identifies a control sequence, defines its syntax, and determines its semantics. The number of parameters and the number of bytes in each parameter are implicit in the function type definition.

A chained Control Sequence Introducer contains the following fields:
- A one-byte length
- A one-byte function type

Length	Function Type

Thus a Control Sequence Introducer is either two bytes or four bytes in length, depending on whether it is chained or unchained.

Control Sequence Chaining

Control sequences may be chained, that is, concatenated. Chaining provides a look-ahead function that permits a processor to avoid changes from processing control sequences to processing graphic characters while scanning or executing Presentation Text Data. When control sequences are chained, the prefix and class bytes are only required in the first control sequence in the chain.

Chaining is signaled by the presence of an odd function type. That is, the low-order bit is B'1'. If a control sequence has a function type with the low-order bit equal to B'1', the string that follows the control sequence is a chained control sequence. A chained control sequence begins with the length field, whereas an unchained control sequence begins with the Control Sequence Prefix. The chain is terminated by a control sequence with an even numbered function type (low-order bit is B'0'), or by the end of the current Presentation Text object. If a

control sequence has a function type with the low-order bit equal to B'0', the string which follows the control sequence may be code points or an unchained control sequence.

Note: In some environments, terminating a chain by ending the Presentation Text object might cause problems, therefore it is recommended that generators always terminate chains with a control sequence whose low-order bit is B'0'.

Chains of control sequences may be as long as desired. Control sequences in a chain are interpreted in the order in which they are encountered.

Table 5 on page 24 lists the control sequences that can appear within Presentation Text data and their function types, both unchained and chained.

Modal Control Sequences

Certain control sequences are modal. That is, they establish a parameter value that persists after the control sequence has been processed. An example is Set Inline Margin, which sets the size of the inline margin. When such a control sequence is processed, its parameter value replaces the existing parameter value. The existing parameter value may have been set in one of the following ways:

- A previous modal control sequence in the current Presentation Text object
- Externally to the Presentation Text object
- By default

The new parameter value remains in effect until a subsequent control sequence for that function is encountered or until the end of the current Presentation Text object.

Architecture Note: Note that when presentation text is processed in a MO:DCA environment where the Presentation Text Data Descriptor (PTD) is carried in the Active Environment Group (AEG) for the page or overlay, or when such text is processed in an IPDS environment, the Presentation Text object is bounded by the beginning of the page and the end of the page. This is sometimes called a *text major* environment. When the PTD is carried in the Object Environment Group (OEG) of a MO:DCA text object, the text object is bounded by the Begin Presentation Text (BPT) and End Presentation Text (EPT) structured fields. For such objects, the PTD in the AEG is ignored.

Application Note: When the Begin Presentation Text (BPT) structured field is encountered in a MO:DCA data stream, all initial text conditions specified in the Presentation Text Data Descriptor (PTD) structured field are set prior to processing the text object. In addition, in AFP environments, whenever a BPT is encountered, AFP presentation servers set the following default page-level initial text conditions *before* the PTD initial conditions are set:

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

Application Note: The PTD also specifies the size of the text object space. When the PTD is specified in the AEG of a MO:DCA page in a text major environment, the extents of the text object space are set equal to the extents of the page. When the PTD is specified in the OEG of a MO:DCA text object, the extents of the text object space can be set to any value in the allowed range.

Control Sequence Default Indicator

The default indicator (X'F..F') in a parameter of a control sequence causes the parameter to use the hierarchical default value for that parameter. The hierarchical default values are listed in Table 10 on page 33.

Control Sequence Introducer

The Control Sequence Introducer is a two-byte or four-byte field, depending on whether the control sequence is unchained or chained.

Syntax: A Control Sequence Introducer can appear only at the beginning of a control sequence.

An *unchained* Control Sequence Introducer has the following format:

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	2–255	Control sequence length	M	N	N
3	CODE	TYPE	X'00' – X'FE'	Control sequence function type	M	N	N

A *chained* Control Sequence Introducer has the following format:

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	UBIN	LENGTH	2–255	Control sequence length	M	N	N
1	CODE	TYPE	X'00' – X'FE'	Control sequence function type	M	N	N

Semantics: A Control Sequence Introducer begins, specifies the length of, and identifies the type of a control sequence. It suspends the processing of text and begins the processing of control sequences.

Pragmatics: A Control Sequence Introducer immediately precedes the data portion of a control sequence.

Control Sequence Prefix

The Control Sequence Prefix marks the beginning of an unchained control sequence. This parameter causes a change in the mode of interpretation of a presentation text stream. When a Control Sequence Prefix is encountered, the bytes immediately following are interpreted as a control sequence rather than as code points. This mode of interpretation continues until the control sequence or control sequence chain is terminated.

Control Sequence Class

The control sequence class characterizes the syntax of the Control Sequence Introducer. This parameter specifies how the introducer of the current control sequence should be interpreted. X'D3' has been assigned for PTOCA. If any other class is encountered, exception condition EC-1C01 exists. The standard action is to ignore the control sequence.

Control Sequence Length

The control sequence length specifies the length of the control sequence beginning with and including itself. The prefix and class are not included in the length. If the length parameter, as specified for individual control sequences, is invalid, exception condition EC-1E01 exists:

- If a mandatory parameter is missing from the control sequence, the standard action is to ignore the control sequence.
- If the control sequence is longer than specified, the standard action is to ignore only the undefined parameters.

If part of an optional parameter field is missing from the control sequence, exception condition EC-1E01 exists. The standard action is to ignore the partially specified optional parameter field.

Control Sequence Function Type

The control sequence function type characterizes the effect and syntax of a control sequence.

This parameter specifies how parameters in the control sequence are to be interpreted. The function type identifies the operation of the control sequence; for example, to set a value or to draw a rule.

Please refer to Table 5 on page 24 for a listing of PTOCA function types. These are the only valid function types. If any other function type is encountered, exception condition EC-0001 exists. The standard action is to ignore the control sequence and continue presenting.

Graphic Character Processing

Format: The format of the graphic characters is specified by the active coded font. The coded font is specified by an external reference to a coded font resource by specification of a coded font local identifier. The coded font determines the length of the code point used to specify each graphic character, and the assignment of a code point to each graphic character. If a character is contained in the Presentation Text object that is not defined in the active coded font, exception condition EC-2100 exists. The standard action is to present the default character defined by the active coded font. Please see "Font Concepts" on page 13 for more information on fonts.

Presentation: Graphic character processing uses the I,B coordinate system. I_c and B_c represent the I and B coordinates of the current presentation position. I_o and B_o represent the origin of the I, B coordinate system. Please refer to Table 11 on page 40, Figure 8 on page 41, and Figure 9 on page 41. Upon entry into the Presentation Text object, I_c and B_c are initialized to the values specified at the data stream level. If values are not specified by the controlling environment, these coordinates are set to their standard action values; that is, $I_c = I_o = 0$ and $B_c = B_o = 0$. See "Enter Object" in Table 11 on page 40. After the object has been entered, the values of I_c and B_c may be changed before the first character is presented, either by parameters in the Presentation Text Data Descriptor or by control sequences in the Presentation Text data.

Each graphic character in a string of Presentation Text data normally causes a character shape, as defined in the active font resource, to be placed on the presentation surface. The character's reference point coincides with the presentation position, I_c,B_c, and the character baseline is made parallel to the baseline by rotation. Simultaneously, the I_c coordinate of the current presentation position is increased by the character increment specified for that character in the active coded font. See "Present character, general case" in Table 11 on page 40.

If a character is to be placed immediately following another character on the presentation surface, the I_c coordinate is first increased or decreased by the intercharacter adjustment. See "Present character, specific cases" in Table 11 on page 40. Then the character shape is placed with the character reference point coincident with this revised presentation position. Last, the current presentation position is incremented by the addition of the character increment value. As each graphic character is placed, the current presentation position is moved in the positive inline direction.

The concept of *between-the-pels* positioning on a presentation surface is important for the presentation of rules. For inline rules, please see Figure 10 on page 42. Use one of the four diagrams, depending upon the rule length and rule width values. When presentation is to begin for an inline rule, the physical pels are located as shown. I and B are not directly represented in the diagrams, since the diagrams are intended to provide the approximate location of the pels to be presented for inline rules. The I and B refer to direction only. For example, if the I-axis is vertical and the I-direction is down as a result of an orientation change, an inline rule of positive length and positive width would still be positioned as in diagram (b).

For baseline rules, please see Figure 11 on page 42. Use one of the four diagrams, depending upon the rule length and rule width values. When presentation is to begin for a baseline rule, the physical pels are located as shown. I and B are not directly represented in the diagrams, since the diagrams are intended to provide the approximate location of the pels to be presented for baseline rules. The I and B refer to direction only. For example, if the B-axis is horizontal and the B-direction is to the left as a result of an orientation change, a baseline rule of positive length and positive width would still be positioned as in diagram (b).

Overrun: If the intercharacter adjustment is zero, *n* characters, each having a fixed character increment of *CI,* may be printed in a presentation space whose I-extent is *n* times *CI*. For example, if *n* is 100 characters and *CI* is 0.1 inch, that is, 10-pitch, all 100 characters will fit on a logical page with an I-extent of 10 inches. In this case, the 100th character just fits within the presentation space. Following placement of the 100th character, the current inline coordinate position, I_c, is one measurement unit beyond the I-extent.

If an attempt is made to present any portion of a character's character box or any portion of a rule beyond the I-extent, exception condition EC-0103 exists. If the presented character is defined in terms of A-space, B-space, and C-space, only the B-space is considered part of the character box. The standard action for this exception condition is to refrain from presenting the character whose character box is outside the object space boundary, and to continue processing without presenting characters. Processing continues until the presentation position is moved to an addressable point that is valid. This exception condition exists regardless of the graphic character causing the condition. For example, if the graphic character is a blank or a variable space character, the exception condition exists even though no marks are made on the presentation surface.

This exception condition ceases to exist when an intercharacter adjustment causes the presented character to move into the object space. The exception condition exists if an intercharacter adjustment causes any part of the character box of the presented character to move out of the object space. Throughout this process the current baseline coordinate, B_c, remains unchanged.

The minimum calculation capacity for overrun handling is equal to the number of bytes that constitutes the parameter. Thus a two-byte parameter requires a minimum of two bytes of storage.

Control sequences are processed according to their semantics without regard to the presence of an overrun. Consider the following example. Assume that the presentation position is I_o, B_o, that is, the upper left corner of the presentation space, and that the character reference point of the character to be presented is the lower left corner of its character box. In this case, an exception condition exists even though the presentation position is within the object space, because some of the character shape extends outside of the object space. Invoking the standard action causes additional character increments to be applied repeatedly, and each character is outside of the object space. In effect, each character received is rejected until a control sequence is received that moves the baseline away from the I-axis, such as Begin Line or Absolute Move Baseline.

Superscripting and Subscripting: Superscripts and subscripts are graphic characters that appear above or below adjacent graphic characters on the same line, and may be smaller than adjacent graphic characters on the same line. They have special purposes, such as representing exponents or footnote numbers. Superscripts and subscripts may be implemented in different ways.

- A font may contain smaller graphic characters which are designed to appear as superscripts or subscripts. These characters may be designed with their character base above, on, or below the nominal baseline. With a font like this, superscripting or subscripting is implicit in presenting one of these graphic characters.
- A font may consist entirely of graphic characters which are designed so that they will appear as superscripts or subscripts when used with other fonts. With a font like this, superscripting or subscripting is implicit in invoking the font.
- A font may be designed without smaller characters for use as superscripts and subscripts. With a font like this, other means must be used to create superscripts and subscripts.
  - Superscripting or subscripting can be accomplished by temporarily lowering or raising the B-axis coordinate of the presentation position. With this technique, the size of the superscript or subscript graphic characters is the same as the immediately preceding graphic characters. This technique is useful for typescript, and the B-axis coordinate is usually lowered or raised ½ line.
  - Superscripting or subscripting can be accomplished by temporarily lowering or raising the B-axis coordinate of the presentation position *and* invoking a different font, whose graphic characters are smaller than the immediately preceding graphic characters. Such smaller graphic characters are usually in a different style specifically designed for superscripting or subscripting. This technique is useful in formal letters and compositions. The distance that the B-axis coordinate is lowered or raised depends on the purpose of the superscript or subscript. For example, a limit to an integral is displaced further than an exponent.

This last technique is the most general, since it can apply to a variety of requirements, including the following:

- Nested superscripts and subscripts, that is, superscripts and subscripts of superscripts or subscripts
- Mathematical symbols of different sizes, for example, integrals, sums, products, and exponents

- Specially stylized superscripts or subscripts, such as italic characters and Greek letters

In the context of superscripting and subscripting, the *established baseline* is the baseline on which a string of graphic characters appears to rest, the *temporary baseline* is the baseline on which a superscript or subscript appears to rest, and the *current baseline* is the baseline on which the next graphic character will appear to rest. The current baseline may be the established baseline or the temporary baseline.

In PTOCA superscripting and subscripting, including the establishment of a temporary baseline, is specified by the Temporary Baseline Move control sequence.

Table 11. Equations for Graphic Character Presentation

WHEN	WHAT	EQUATIONS
Enter Object		
	Use initial values from data stream	$I_c = I_h$ $B_c = B_h$
	Or use default initial values	$I_c = I_o = 0$ $B_c = B_o = 0$
Present character, general case		$B_c = B_o = 0$
	Present variable space character	$I_{cnew} = I_c$ + VSI
	Present graphic character	Present character $I_{cnew} = I_c$ + CI
Present character, specific cases		
	Following incrementing character: Present first character (incrementing) Followed by second character (incrementing)	$I_{cnew} = I_c$ +/- ADJSTMNT Present first character (incrementing) $I_{cnew} = I_c$ + CI $I_{cnew} = I_c$ +/- ADJSTMNT Present second character (incrementing) $I_{cnew} = I_c$ + CI
	Following incrementing character: Present first character (non-incrementing) Present second character (incrementing)	$I_{cnew} = I_c$ +/- ADJSTMNT Present first character (non-incrementing) $I_{cnew} = I_c$ Present second character (incrementing) $I_{cnew} = I_c$ + CI
	Following incrementing character: Present Variable Space Character Followed by first character (incrementing)	$I_{cnew} = I_c$ + VSI Present first character (incrementing) $I_{cnew} = I_c$ + CI

This table shows how the I_{cnew} coordinate is modified during the presentation of characters.

Figure 8. Presentation Position without Intercharacter Adjustment

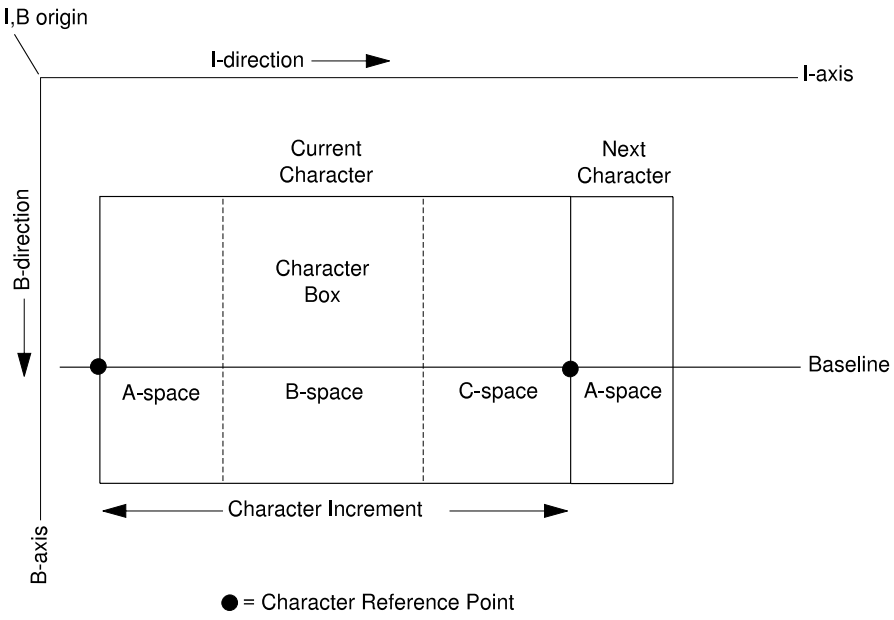


Figure 9. Presentation Position with Intercharacter Adjustment

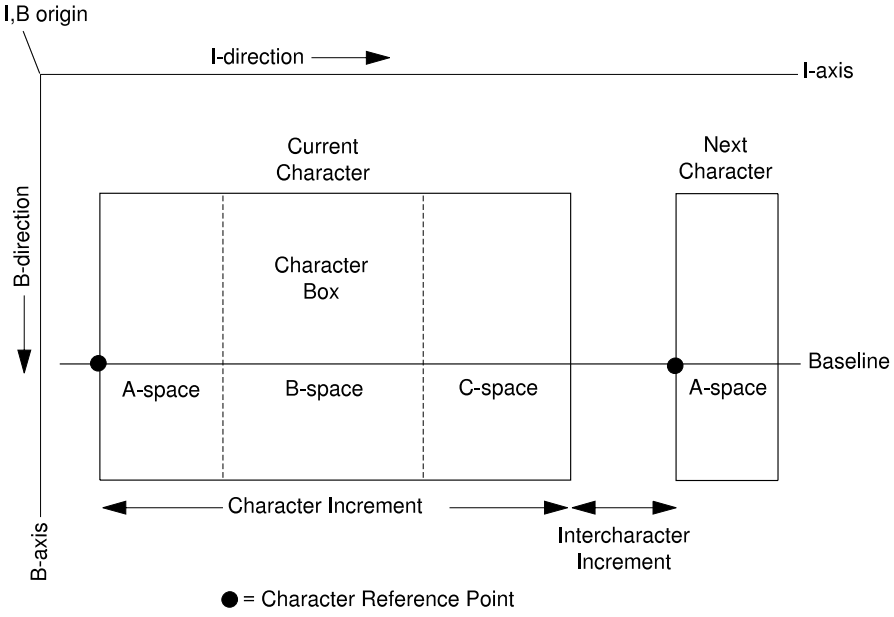


Figure 10. Between-the-Pels Illustrations for Inline Rules

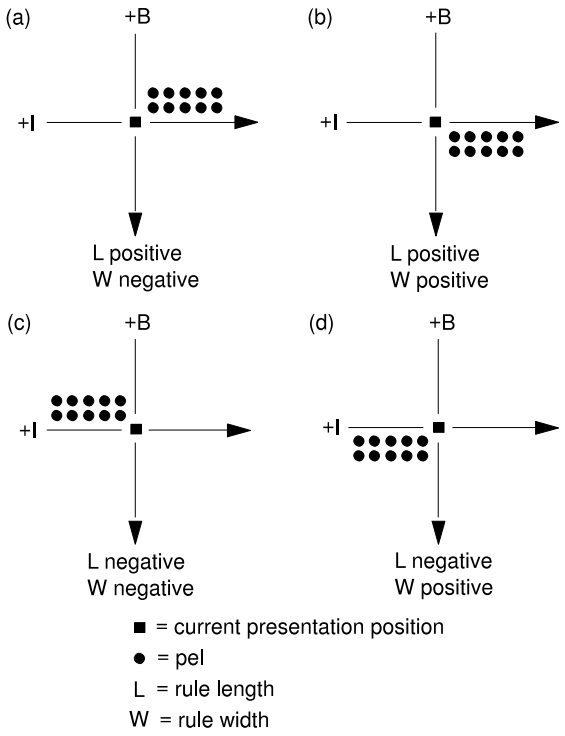
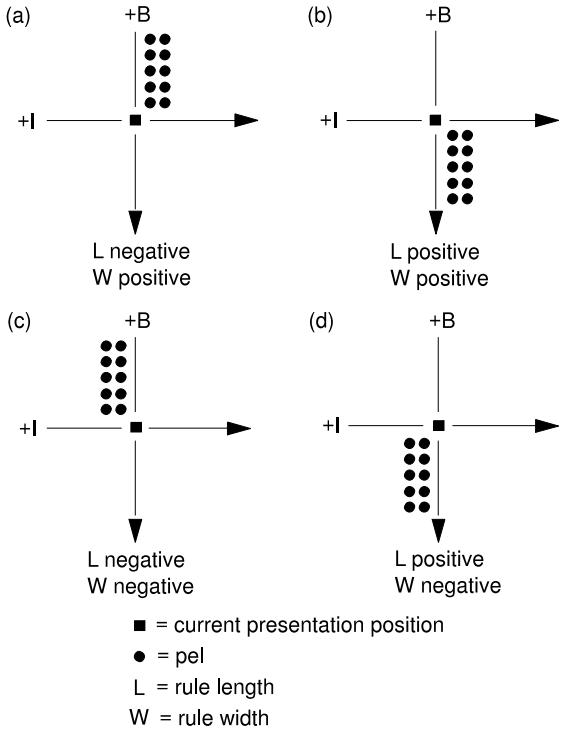


Figure 11. Between-the-Pels Illustrations for Baseline Rules



Exception Conditions: Control sequence processing and graphic character processing can cause the following exception conditions:

- EC-1E01...A mandatory parameter is missing.
- EC-1C01...The control sequence class is invalid.
- EC-1E01...The control sequence length is not valid.
- EC-1E01...An optional parameter is partially missing.
- EC-0001...The control sequence function type is invalid.
- EC-2100...The Presentation Text object contains a graphic character code point that is not defined in the active coded font.
- EC-0103...An attempt is made to present a character or a rule outside of the object space.

Presentation Text Data

Presentation Text data consists of character code points and embedded control sequences, which together are called presentation text. The architected content of the presentation text depends on the subset selected by the generator of the object.

Syntax: Please see the appendixes for environmental information about the syntax of Presentation Text data.

Semantics: Presentation Text data inherits any modal parameter values, such as inline margin and coded font, that were specified externally to the Presentation Text object. It also inherits the current presentation position. These values may be specified by the controlling environment.

The content of Presentation Text data is graphic character code points and control sequences. For the syntax, semantics, and pragmatics of the control sequences, see "Control Sequence Detailed Descriptions" on page 45.

Pragmatics: Presentation text can consist of almost any string of eight-bit bytes. In the single-byte mode, these bytes may be seven-bit code points, with the leading bit zero, or eight-bit code points. In the double-byte mode, they may be sixteen-bit code points. The coded character representation is determined through reference to a coded font in the controlling environment. All code points in the presentation text are translated for presentation by reference to the active coded font, with the following exceptions:

- The Variable Space Character, which is normally X'40' for EBCDIC single-byte fonts, X'20' for ASCII single-byte fonts, X'4040' for EBCDIC double-byte fonts, X'2020' for ASCII double-byte fonts, and X'0020' or X'00A0' for Unicode fonts;
- The Control Sequence Prefix, which is X'2B'.

If it is necessary to present the Control Sequence Prefix code point, use the Transparent Data control sequence.

All control sequence displacements are expressed in terms of the I,B coordinate system. The directions of the I,B coordinates are specified initially in the text orientation initial conditions in the Presentation Text Data Descriptor. They can be respecified within a Presentation Text object with a Set Text Orientation control sequence.

When processing begins for the first Presentation Text Data in a given Presentation Text object, the current presentation position, I_c and B_c, is set using values from the Presentation Text Data Descriptor. The initial inline coordinate value and initial baseline coordinate value are used. When processing begins for subsequent Presentation Text data within the same Presentation Text object, the current presentation position is set to the last presentation position from the previous Presentation Text data.

Each graphic character code point in a Presentation Text object causes the character reference point of the character shape to be placed at I_c, B_c with the character baseline parallel to the baseline. I_c is increased by the character increment and, for a character immediately followed by another character, is adjusted by the intercharacter adjustment.

In addition to graphic character code points, Presentation Text data can contain embedded control sequences. These are strings of two or more bytes which signal an alternate mode of processing for the content of the current Presentation Text data. Although PTOCA allows the definition of various types of control sequences, only one type of control sequence is permitted in the Presentation Text data. The escape character to be used in the Control Sequence Introducer is X'2B'.

As previously stated, control sequences can be chained. However, there is no requirement that control sequences be chained.

Control Sequence Detailed Descriptions

The following pages contain detailed descriptions of the PTOCA control sequences. The descriptions show the syntax, semantics, and pragmatics of the control sequences.

Documentation Conventions: Each PTOCA control sequence is described syntactically in this reference by a table. Following each table is semantic information related to each component of the control sequence. Syntactically descriptive material following the table may indicate that additional restrictions apply to the control sequence defined by the table. Each syntax table has the following columns:

- *Offset* refers to the position of a parameter.
- *Type* denotes the syntax of the parameter by data type. Please see "Parameter Data Types" on page 32 for more information.
- *Name* is a short field name suitable for coding.
- *Range* denotes the smallest and largest valid values that may occur in this field. Negative numbers are expressed in twos-complement form.
- *Meaning* gives an explanatory or descriptive name for the parameter.
- *M/O* refers to the parameter's appearance in the control sequence. *O* means that the parameter is optional. That is, the generator of the Presentation Text object does not have to include this parameter. However, the receiver must support this parameter if it supports the control sequence that contains the parameter. *M* means that the parameter's appearance is mandatory. If a particular control sequence occurs in the object, all mandatory parameters in that control sequence must be present.

  If a mandatory parameter is missing, exception condition EC-1E01 exists. The standard action is to ignore the control sequence to which the missing parameter belongs.
- *Def* refers to the existence of a PTOCA default value for the parameter which is to be used when no explicit value is provided and the standard action is to continue. *Y* means that there is such a value. *N* means that there is no such value. This value, also called the standard action value, is defined in the description which follows each syntax table.
- *Ind* specifies the validity of the default indicator. *Y* means that the default indicator is valid. *N* means that the default indicator is not valid. The default indicator is represented by X'F..F' and is described in "Control Sequence Default Indicator" on page 36.

Reserved Fields Certain fields may be denoted 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 comprising a field defined to be reserved should be given a value of zero by generating applications. Receiving applications should ignore any value in a reserved field.

Interpreting Ranges The range values shown in the syntax tables assume a measurement unit of 1/1440 inch. That is, they assume that the measurement base is ten inches, and that the X_p-units per unit base and Y_p-units per unit base are 14,400. If this assumption is not correct, and a different measurement unit is supported, the correct range values can be determined by using the following steps:

1. Calculate the number of actual supported units per inch (X) as follows:
   - If the measurement base is ten inches, divide the number of supported units per ten inches by 10.
   - If the measurement base is ten centimeters, multiply the number of supported units per ten centimeters by 0.254.
2. Calculate the ratio of actual supported units per inch (X) to the assumed 1,440 units per inch as follows:
   - Divide (X) by 1,440, yielding the ratio (Y).
3. Calculate the new range value in the supported measurement units as follows:
   - Convert the old range value to base ten; then multiply it by the ratio (Y).
   - Round to the nearest integer.

For example, suppose that the specified range is X'8000'–X'7FFF' when using 14,400 units per 10 inches. The equivalent range at a unit of measure of 1/240 of an inch is calculated as follows:

1. Supported units per inch = 2,400 ÷ 10 = 240
2. Ratio of supported units per inch to 1,440 units per inch = 240 ÷ 1,440 = 1/6
3. Range at 2,400 units per 10 inches:
   a. X'8000' = -32,768 (converted to base 10)

      -32,768 × 1/6 = -5,461.3333
   b. X'7FFF' = 32,767(converted to base 10)

      32,767 × 1/6 = 5,461.1667

Therefore, the equivalent range at 2,400 units per 10 inches is -5,461 to 5,461 which in hexadecimal is X'EAAB' to X'1555'.

Absolute Move Baseline (AMB)

The Absolute Move Baseline control sequence moves the baseline coordinate relative to the I-axis.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4	Control sequence length	M	N	N
3	CODE	TYPE	X'D2' – X'D3'	Control sequence function type	M	N	N
4–5	SBIN	DSPLCMNT	X'0000' – X'7FFF'	Displacement	M	N	N

DSPLCMNT is a signed binary number expressed in measurement units. It does not accept the default indicator. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45.

Semantics

This control sequence specifies a displacement, DSPLCMNT, in the B-direction from the I-axis of the object space to a new baseline coordinate position. After execution of this control sequence, presentation is resumed at the new baseline coordinate position. This control sequence does not modify the current inline coordinate position.

$I_{cnew} = I_c$

$B_{cnew} = B_o + DSPLCMNT$

If the value of DSPLCMNT is not supported or is not within the range specified by PTOCA, exception condition EC-1301 exists. The standard action in this case is to continue presentation according to the description given in the Pragmatics section.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, "Compliance with PTOCA", on page 145, Appendix A, "MO:DCA Environment", on page 155, and Appendix B, "IPDS Environment", on page 161. See "Related Publications" on page vi for data-stream documentation.

Pragmatics

If DSPLCMNT is zero, the addressable position is the I-axis, and any intercharacter adjustment is not applied. If a move is to a presentation position for which the character box will exceed the object space and an attempt is made to present there, exception condition EC-0103 exists. The standard action in this case is to refrain from presenting the character that exceeds the object space, and to continue processing without presenting characters until the presentation position occupies a valid addressable position for the character being presented. Then presentation of characters may resume. PTOCA does not constrain the advancement of the baseline coordinate position toward the I-axis. However, a constraint of this type may be imposed by the subset level or by the receiver. If this constraint is applied and DSPLCMNT is negative, exception condition EC-1403 exists. The standard action in this case is to ignore the control sequence.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1301...The value of DSPLCMNT is not supported or is not within the range specified by PTOCA.
- EC-0103...The presentation position is outside the object space and presentation is attempted.
- EC-1403...Negative DSPLCMNT is not valid.

Absolute Move Inline (AMI)

The Absolute Move Inline control sequence moves the inline coordinate position relative to the B-axis.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control sequence prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4	Control sequence length	M	N	N
3	CODE	TYPE	X'C6' – X'C7'	Control sequence function type	M	N	N
4–5	SBIN	DSPLCMNT	X'0000' – X'7FFF'	Displacement	M	N	N

DSPLCMNT is a signed binary number expressed in measurement units. It does not accept the default indicator. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45.

Semantics

This control sequence specifies a displacement, DSPLCMNT, in the I-direction from the B-axis of the object space to a new inline coordinate position, and resumes presentation at the new inline coordinate position. It does not modify the current baseline coordinate position.

$I_{cnew} = I_o + DSPLCMNT$

$B_{cnew} = B_c$

If the value of DSPLCMNT is not supported or is not within the range specified by PTOCA, exception condition EC-1401 exists. The standard action in this case is to continue presentation according to the description given in the Pragmatics section.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, "Compliance with PTOCA", on page 145, Appendix A, "MO:DCA Environment", on page 155, and Appendix B, "IPDS Environment", on page 161. See "Related Publications" on page vi for data-stream documentation.

Pragmatics

If the value of DSPLCMNT is zero, the addressable position is at the B-axis, and any intercharacter adjustment is not applied. If a move is to a presentation position for which the character's character box will exceed the object space and an attempt is made to present there, exception condition EC-0103 exists. The standard action in this case is to refrain from presenting the character that exceeds the object space, and to continue processing without presenting characters until the presentation position occupies a valid addressable position for the character being presented. Then presentation of characters may resume.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1401...The value of DSPLCMNT is not supported or is not within the range specified by PTOCA.

- EC-0103...The presentation position is outside the object space and presentation is attempted.

Begin Line (BLN)

The Begin Line control sequence begins a new line.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control sequence prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	2	Control sequence length	M	N	N
3	CODE	TYPE	X'D8' – X'D9'	Control sequence function type	M	N	N

Semantics

This control sequence marks the beginning of a new line. It increments the current baseline coordinate position by the amount of the baseline increment, INCRMENT. It sets the current inline coordinate to the inline margin. Presentation is resumed at the new baseline coordinate position at the inline margin.

$I_{cnew} = I_{margin}$

$B_{cnew} = B_c + INCRMENT$

Exception Conditions

This control sequence can cause the following exception condition:

- None

Begin Suppression (BSU)

The Begin Suppression control sequence marks the beginning of a string of presentation text that may be suppressed from the visible output.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	3	Control sequence length	M	N	N
3	CODE	TYPE	X'F2' – X'F3'	Control sequence function type	M	N	N
4	CODE	LID	X'00' – X'FF'	Suppression identifier	M	N	N

LID is a code with no units of measure. It does not accept the default indicator.

Semantics

This control sequence marks the beginning of a string of presentation text that may be suppressed from the visible output. It is activated by a local identifier, LID. This control sequence works in conjunction with the End Suppression control sequence, which also contains a LID. Please see “End Suppression (ESU)” on page 58. If the LID in this control sequence has been activated for the current Presentation Text object in the data stream hierarchy, the string of presentation text between this control sequence and the next End Suppression control sequence with the same LID does not appear in the visible output. Even though the text does not appear, all control sequences within the suppressed field are executed, and the I-coordinate and B-coordinate are updated as if the text had appeared. Only the actual presentation of the graphic characters and rules is suppressed. Please see Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161 for further information about suppression.

If the value of the LID is not supported or is not within the range specified by PTOCA, exception condition EC-9801 exists. The standard action in this case is to ignore the control sequence. Please see the Pragmatics section for additional exception conditions.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

If the LID in this control sequence is not activated in the data stream hierarchy, this control sequence and the corresponding End Suppression control sequence are processed as no-operations.

Pragmatics

Several kinds of exception conditions can exist with the Begin and End Suppression control sequences. These exception conditions can exist whether or not the LID has been activated in the data-stream hierarchy.

- Nesting of Begin and End Suppression control sequences is not allowed. If a second Begin Suppression control sequence follows a Begin Suppression control sequence without an intervening and corresponding End Suppression control sequence, exception condition EC-0601 exists. The standard action in this case is to ignore the second Begin Suppression control sequence.

- If a Begin Suppression control sequence is followed by an End Suppression control sequence that has a different value for the LID, exception condition EC-0201 exists. The standard action in this case is to ignore the End Suppression control sequence.
- If an End Suppression control sequence occurs in a Presentation Text object without a previous valid Begin Suppression control sequence, exception condition EC-0201 exists. The standard action is to ignore the End Suppression control sequence.
- If a Begin Suppression control sequence appears in a Presentation Text object with no corresponding End Suppression control sequence, exception condition EC-0401 exists. The standard action in this case is to process the object as if the corresponding End Suppression control sequence were present at the end of the object. That is, all of the data following the Begin Suppression control sequence is suppressed.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-9801...The value of the LID is not supported or is not in the range specified by PTOCA.
- EC-0601...Nesting exists.
- EC-0201...The values of the LIDs do not match within a BSU, ESU pair.
- EC-0201...An ESU control sequence occurs without a preceding BSU control sequence.
- EC-0401...A BSU control sequence occurs without a succeeding ESU control sequence.

Draw B-axis Rule (DBR)

The Draw B-axis Rule control sequence draws a rule in the B-direction.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4,7	Control sequence length	M	N	N
3	CODE	TYPE	X'E6' – X'E7'	Control sequence function type	M	N	N
4–5	SBIN	RLENGTH	X'8000' – X'7FFF'	Length	M	N	N
6–8	SBIN	RWIDTH	See Semantics section	Width	O	Y	Y

RLENGTH and RWIDTH are signed binary numbers in measurement units. The range for RLENGTH assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45.

Semantics

This control sequence specifies the dimensions of a rule that extends from the current presentation position in both the B-direction and the I-direction. The current I-axis and B-axis coordinates are not changed by this control sequence.

The length parameter, RLENGTH, is the length of the rule in the B-direction. If RLENGTH is omitted, exception condition EC-1E01 exists. The standard action is to treat this control sequence as a no-operation. If the value of RLENGTH is not supported or is not within the range specified by PTOCA, exception condition EC-8202 exists. The standard action is to ignore the control sequence and continue presentation with the value determined according to the description given in the Pragmatics section.

The width parameter, RWIDTH, is the width of rule in the I-direction. RWIDTH consists of a three-byte two-part binary number of the form AB. Both A and B are in measurement units.

- A is a two-byte binary number from -32,768 through +32,767
- B is a one-byte binary fraction with bit 0 denoting 1/2 measurement unit, bit 1 denoting 1/4 measurement unit, and bit N denoting $1/(2^{(N+1)})$ measurement unit.

If RWIDTH is the default indicator, a value is obtained from the hierarchy. Please see the Pragmatics section for further information. If the value of RWIDTH is not supported or is not within the range specified by PTOCA, exception condition EC-8002 exists. The standard action is to ignore the control sequence and continue presentation with the value determined according to the description given in the Pragmatics section.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, "Compliance with PTOCA", on page 145, Appendix A, "MO:DCA Environment", on page 155 , and Appendix B, "IPDS Environment", on page 161. See "Related Publications" on page vi for data-stream documentation.

Pragmatics

If a width or length is specified that, when converted to pels, requires finer resolution than a device supports, the device uses the next smaller width or length that it does support. If a specified width or length becomes less than the finest device resolution, the device uses its finest resolution. If the value of RLENGTH or RWIDTH is zero, no marks appear. Such resolution correction does not constitute an exception condition. If a parameter value will cause any part of the rule to extend outside of the object space, exception condition code EC-0103 exists. The standard actions in this case are the following:

- For extensions in the B-direction, truncate the rule at the object space boundary. Receivers using character underscore to create rules must position, begin, or end characters in such a way as to prevent extension beyond the limits of the object space.
- For extensions in the I-direction, limit presentation to the portion of the rule that can be presented within the object space. If the receiver's minimum resolution will cause the object space to be exceeded, do not present the rule.

A negative value for RLENGTH or RWIDTH reverses the direction of the corresponding extent, allowing definition of a rule in four directions from a given presentation position. The support of negative values in the I-direction and the B-direction is optional.

A rule of length +1 must have a different starting point from a rule of -1. The difference, within the tolerances of the receiver, is equal to the receiver's finest resolution. If a parameter value is received by a receiver that does not support it, exception condition EC-8202 exists. The standard action in this case is to ignore the control sequence. The recommended default value for RWIDTH is receiver-dependent. For example, it may be the width of the vertical bar character in the active font. A receiver incapable of drawing the rule may substitute a graphic sequence that represents it.

For further information on rule positioning, please refer to Figure 11 on page 42.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-0103...A parameter value will cause the rule to be outside the object space.
- EC-1E01...RLENGTH is missing.
- EC-8002...The value for RWIDTH is not supported or is not in the range specified by PTOCA.
- EC-8202...The value for RLENGTH is not supported or is not in the range specified by PTOCA; or, a parameter value is negative and the receiver cannot process it.

Draw I-axis Rule (DIR)

The Draw I-axis Rule control sequence draws a rule in the I-direction.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4,7	Control sequence length	M	N	N
3	CODE	TYPE	X'E4' – X'E5'	Control sequence function type	M	N	N
4–5	SBIN	RLENGTH	X'8000' – X'7FFF'	Length	M	N	N
6–8	SBIN	RWIDTH	See semantics section	Width	O	Y	Y

RLENGTH and RWIDTH are signed binary numbers in measurement units. The range for RLENGTH assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45.

Semantics

This control sequence specifies dimensions of a rule that extends from the current presentation position in both the I-direction and the B-direction. The current I-axis and B-axis coordinates are not changed by this control sequence.

The length parameter, RLENGTH, is the length of the rule in the I-direction. If RLENGTH is omitted, exception condition EC-1E01 exists. The standard action is to treat this control sequence as a no-operation. If the value of RLENGTH is not supported or is not within the range specified by PTOCA, exception condition EC-8202 exists. The standard action is to continue presentation with the value determined according to the description given in the Pragmatics section.

The width parameter, RWIDTH, is the width of the rule in the B-direction.

RWIDTH consists of a three-byte two-part binary number of the form AB. Both A and B are in measurement units.

- A is a two-byte binary number from -32,768 through +32,767
- B is a one-byte binary fraction with bit 0 denoting 1/2 measurement unit, bit 1 denoting 1/4 measurement unit, and bit N denoting $1/(2^{(N+1)})$ measurement unit.

If RWIDTH is the default indicator, a value is obtained from the hierarchy. See the Pragmatics section for further information. If the value of RWIDTH is not supported or is not within the range specified by PTOCA, exception condition EC-8002 exists. The standard action is to continue presentation with the value determined according to the description given in the Pragmatics section.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

If a width or length is specified that, when converted to pels, requires finer resolution than a device supports, the device uses the next smaller width or length that it does support. If a specified width or length becomes less than the finest device resolution, the device uses its finest resolution. If the value of RLENGTH or RWIDTH is zero, no marks appear. Such resolution correction does not constitute an exception condition. If a parameter value will cause any part of the rule to extend outside of the object space, exception condition EC-0103 exists. The standard actions in this case are the following:

- For extensions in the I-direction, truncate the rule at the object space boundary. Receivers using character underscore to create rules must position, begin, or end characters in such a way as to prevent extension beyond the limits of the object space.
- For extensions in the B-direction, presentation is limited to the portion of the rule that can be presented within the object space. If the receiver's minimum resolution will cause the object space to be exceeded, do not present the rule.

A negative value of RLENGTH or RWIDTH reverses the direction of the corresponding extent, allowing definition of a rule in four directions from a given presentation position. The support of negative values in the I-direction and B-direction is optional.

A rule of length +1 must have a different starting point from a rule of -1. The difference, within the tolerances of the receiver, is equal to the receiver's finest resolution. If a parameter value is received by a receiver that does not support it, exception condition EC-8202 exists. The standard action in this case is to ignore the control sequence. The recommended default value for RWIDTH is receiver-dependent. For example, it may be the width of the underscore character in the active font. A receiver incapable of drawing the rule may substitute a graphic sequence that represents it.

For further information on rule positioning, please refer to Figure 10 on page 42.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-0103...A parameter value will cause the rule to be outside the object space.
- EC-1E01...RLENGTH is missing.
- EC-8002...The value for RWIDTH is not supported or is not in the range specified by PTOCA.
- EC-8202...The value for RLENGTH is not supported or is not in the range specified by PTOCA; or, a parameter value is negative and the receiver cannot process it.

End Suppression (ESU)

The End Suppression control sequence marks the end of a string of presentation text suppressed from the visible output.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	3	Control sequence length	M	N	N
3	CODE	TYPE	X'F4' – X'F5'	Control sequence function type	M	N	N
4	CODE	LID	X'00' – X'FF'	Suppression identifier	M	N	N

LID is a code with no units of measure. It does not accept the default indicator.

Semantics

This control sequence marks the end of a string of presentation text that has been suppressed. It works in conjunction with the Begin Suppression control sequence. Please see “Begin Suppression (BSU)” on page 52 for information on the Begin Suppression control sequence. If the value of the LID is not supported or is not within the range specified by PTOCA, exception condition EC-9801 exists. The standard action in this case is to ignore this control sequence and continue presentation with the value determined according to the data-stream hierarchy.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

This control sequence does not change the current addressable position. In order to suppress a text string from the visible output, the string should be bounded by a Begin Suppression control sequence and an End Suppression control sequence that have the same values for the LID. In addition, the controlling environment must activate the LID.

Exception Conditions

This control sequence can cause the following exception condition:

- EC-9801...The value of LID is not supported or is not in the range specified by PTOCA.

Glyph Advance Run (GAR)

The Glyph Advance Run control sequence specifies the relative displacement along the current baseline (in the i-direction) to the glyph origin for each glyph ID in the preceding GIR from the text position at the start of the GLC.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	UBIN	LENGTH	X'04' – X'FE'; even values only	Control sequence length. The length is 4 + n·(size of ADVANCE)	M	N	N
1	CODE	TYPE	X'8C', X'8D'	Control sequence function type	M	N	N
2-3			X'0000'	Reserved; should be zero	M	N	N
Zero or more advances along the baseline in the following format:							
+0-1	UBIN	ADVANCE	X'0000' – X'FFFF'	Glyph advance along the baseline	O	N	N

ADVANCE is a signed binary number in measurement units. The range for ADVANCE assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45.

Semantics

This control sequence carries a sequence of glyph advances that correspond to the glyph IDs in the preceding GIR. Each advance is a 2-byte unsigned displacement along the current baseline that is measured from the text position that was defined at the start of the GLC chain to the glyph origin for each glyph ID. The displacement is in the i-axis direction. The first advance corresponds to the first glyph ID, the second advance corresponds to the second glyph ID, and so forth. Advances are specified using the current measurement units.

Each GAR control sequence must be chained to a preceding GIR control sequence. It can be followed by a chained GOR, GIR, or UCT control sequence. If it is followed by a different control sequence, the GAR terminates the GLC chain.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-9C03...Invalid sequence. The GAR is not preceded by a GIR, or, if it indicates chaining, it is not followed by a GOR, GIR, or a UCT.
- EC-9C06...GIR, GAR, or GOR control sequence found outside of a GLC chain. A GIR, GAR, or GOR was found that is not part of a GLC chain.
- EC-9C08...Glyph Advance count mismatch. The number of glyph advances specified is not the same as the number of glyph IDs specified in the preceding GIR.

Glyph ID Run (GIR)

The Glyph ID Run control sequence specifies an array of glyph IDs from the current TrueType/OpenType font.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	UBIN	LENGTH	X'04' – X'FE'; even values only	Control sequence length. The length is 4 + n·(size of GLYPHID)	M	N	N
1	CODE	TYPE	X'8B'	Control sequence function type	M	N	N
2-3			X'0000'	Reserved; should be zero	M	N	N
Zero or more glyph IDs in the following format:							
+0-1	UBIN	GLYPHID	X'0000' – X'FFFF'	Glyph ID	O	N	N

Semantics

This control sequence carries a sequence of glyph ids in the current font or in a font linked to the current font. This font is identified by the font OID in the GLC.If the font is in a font collection, the OID in the GLC corresponds to the font collection and the font name in the GLC is used to identify the font.

A GIR control sequence must be chained to a GLC, if in the first grouping, or to a preceding GAR or GOR, if in subsequent groupings. It must be followed by a chained GAR that contains a glyph advance for each glyph ID in this control sequence.

Pragmatics

It is possible that a text run may require more glyphs than the GIR can contain. Composition applications can handle this by specifying multiple GIR/GAR[/GOR] groupings. If there are n glyph IDs contained in a GIR, then the chained GAR must contain n advances and the optional GOR must contain n offsets.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-9C02...Invalid glyph ID. The current font does not contain a specified glyph ID.
- EC-9C03...Invalid sequence. The GIR is not preceded by either a GLC or a GAR or a GOR.
- EC-9C06...GIR, GAR, or GOR control sequence found outside of a GLC chain. A GIR, GAR, or GOR was found that is not part of a GLC chain.

Glyph Layout Control (GLC)

The Glyph Layout Control control sequence marks the start of one or more glyph run groupings that together render text using arrays of glyph identifiers and positions.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	10 - (p - 1)	Control sequence length	M	N	N
3	CODE	TYPE	X'6D'	Control sequence function type	M	N	N
4–5	SBIN	IADVNCE	X'8000' – X'7FFF'	The advance along the current baseline after processing this GLC chain. A value of X'0000' indicates that the current text position is not changed after processing the GLC chain.	M	N	N
6	UBIN	OIDLGTH	0, 13-129	Length of FONTOID parameter	M	N	N
7	UBIN	FFNLGTH	0 - (255 - (10 + OIDLGTH)), even values only	Length of FFONTNME parameter	M	N	N
8-11			X'00...00'	Reserved; should be zero	M	N	N
12-n	UBIN	FONTOID		Object Identifier for the font that contains the glyphs to be rendered. • Offset 12 – must be X'06' • Offset 13 – length of content bytes • Offset 14 to n – OID content	O	N	N
(n + 1) - p	CHAR	FFONTNME		Full font name of the font identified by the FONTOID parameter, unless the font is in a collection, in which case the OID corresponds to the collection. The name is encoded in UTF-16BE.	O	N	N

IADVNCE is a signed binary number in measurement units. The range for IADVNCE assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in “Interpreting Ranges” on page 45.

IADVNCE specifies the advance along the current baseline (in the inline direction) that is caused by processing this GLC chain. When this advance is added to the text position at the start of the GLC, it defines

the current text position after the GLC chain has been processed. A value of X'0000' indicates that the current text position is not changed after processing the GLC chain.

OIDLGTH specifies the length in bytes of the FONTOID parameter. A value of X'00' indicates that the FONTOID parameter is not specified and the presentation device is to use the FONTOID specified in the previous GLC in the text object that specified this parameter. Note that when a font matching the FONTOID is found, it is always processed with the presentation parameters determined by the font mapping for the LID in the last SCFL control sequence.

Application Note: When OIDLGTH = 0, the presentation device searches backwards through the PTOCA control sequences until a GLC with an OID is found or the beginning of the text object is reached. In MO:DCA environments, the beginning of the text object is indicated by the Begin Presentation Text (BPT) structured field, which causes AFP presentation servers to reset the currently active font to the presentation device default font by issuing a SCFL with LID = X'FF'. The AFP generator must therefore always define an active font at the start of the text object by generating a SCFL control sequence that specifies the font LID and a GLC that specifies the font OID.

FFNLGTH specifies the length in bytes of the FFONTNME parameter. A value of X'00' indicates that the FFONTNME parameter is not specified. This parameter must be set to X'00' if the OIDLGTH parameter is set to X'00', or exception condition EC-9C0B exists.

FONTOID specifies the OID of the font used to present the glyph runs in the GLC chain. If the font is in a font collection, the OID is for the collection. Presentation devices must validate the OID specified in the GLC against the current font or font collection. If the current font is linked to additional fonts, the FONTOID may refer to the current font (the base font) or to any font linked to that base font. Only the font identified by FONTOID will be used with the GLC chain, the other linked fonts are ignored when placing glyphs. See the *Mixed Object Document Content Architecture Reference,* AFPC-0004 for the definition of the algorithm used to calculate the OID of a TrueType/OpenType font.

FFONTNME specifies the Full Font Name (FFN) of the font identified by the FONTOID parameter, unless the FONTOID corresponds to a font collection. If FONTOID corresponds to a font collection, the FFONTNME parameter specifies the Full Font Name of the font in the collection. This parameter is used to search for a specific font in a font collection if the FONTOID parameter identifies a collection. The name is encoded in UTF-16BE. This parameter must not be specified if the FONTOID parameter is not specified; if it is, exception condition EC-9C0B exists.

Semantics

This control sequence marks the start of a run of glyph IDs and positions contained in subsequent glyph run control sequence groupings that make up the GLC chain. The glyph IDs in the TrueType/OpenType font specified by the FONTOID are carried in GIR control sequences. The advances along the current baseline (in the i-axis direction) are carried in GAR control sequences. The optional offsets from the current baseline (in the b-axis direction) are carried in GOR control sequences. These subsequent controls must be chained to the GLC using the PTOCA chaining rules.

The GLC control sequence must be followed by one or more GIR/GAR[/GOR] groupings. (The notation “[/GOR]” will be used to indicate that the GOR is optional.) No other control sequences can be interspersed within the GIR/GAR[/GOR] groupings or between them. Each GIR/GAR[/GOR] grouping causes a set of glyphs to be rendered.

An optional UCT containing the Unicode encoded text may be chained to the last GAR[/GOR] and terminates the chain. This UCT is not rendered. If the UCT is not specified, the last GAR[/GOR] terminates the chain.

After the GLC chain has been processed:
$I_{cnew} = I_c$ + IADVNCE
$B_{cnew} = B_c$

Pragmatics

The GIR/GAR[/GOR] groupings accumulate glyphs from the current font along a single baseline. While there is no restriction on the order of the glyphs, it is recommended that the composition application list them sequentially in the inline direction.

The UCT is optional and carries the Unicode text that the GIR/GAR[/GOR] groupings will present. Its use is recommended as it provides applications the ability to interpret the sequence of glyphs rendered by the presentation device.

The application must specify the glyph IDs, glyph advances, and glyph offsets in the same order. If there are n glyph IDs present in a GIR, then the subsequent GAR must contain n advances, and the optional GOR must contain n offsets.

If a text run requires more glyphs than a GIR can contain, the applications can provide multiple GIR/GAR [/GOR] groupings chained to a single GLC.

In a GIR/GAR[/GOR] grouping, if all glyph offsets are 0, the application can choose not to specify the GOR(s). If one glyph offset is not zero, the application must specify the GOR entries in the same order as the GIR and GAR entries.

The introduction of support for glyph runs affects the operation of some current PTOCA control sequences. The effects are described in the following table:

Table 12. Interaction of GLC chain with other control sequences

Control Sequence	Effect
Modal control sequences	
Set Baseline Increment (SBI)	Has no effect on GLC presentation
Set Coded Font Local (SCFL)	Establishes the current font for use by GLC. When linked fonts are used, the font specified by the SCFL is always the base font.
Set Extended Text Color (SEC)	Specifies the foreground color used to present the glyphs in the subsequent GLC chains
Set Inter-character adjustment (SIA)	Has no effect on GLC presentation
Set Inline Margin (SIM)	Has no effect on GLC presentation
Set Text Color (STC)	Specifies the foreground color used to present the glyphs in the subsequent GLC chains
Set Text Orientation (STO)	Establishes the I-direction and B-direction used to present the glyphs in the subsequent GLC chains
Set Variable Space Character Increment (SVI)	Has no effect on GLC presentation
Field control sequences	
Temporary Baseline Move (TBM)	Temporarily moves the baseline coordinate relative to the established baseline coordinate position. As with character-based text, glyph runs in the GLC chains are positioned relative to this temporary baseline.
Overstrike (OVS), Underscore (USC)	Have no effect on GLC presentation
Directive control sequences	
Absolute Move Baseline (AMB)	Moves the baseline coordinate position relative to the I-axis.

Table 12 Interaction of GLC chain with other control sequences (cont'd.)

Control Sequence	Effect
Absolute Move Inline (AMI)	Moves the inline coordinate position relative to the B-axis
Begin Line (BLN)	Begins a new line
Begin Suppression (BSU)/End Suppression (ESU)	Causes presentation of the glyphs in all GLC chains that are between the BSU and ESU to be suppressed if the corresponding suppression ID is activated
Draw B-axis Rule (DBR)	Has no effect on subsequent GLC presentation
Draw I-axis Rule (DIR)	Has no effect on subsequent GLC presentation
No Operation (NOP)	Has no effect on subsequent GLC presentation
Relative Move Baseline (RMB)	Moves the baseline coordinate relative to the current baseline coordinate position
Relative Move Inline (RMI)	Moves the inline coordinate relative to the current inline position
Repeat String (RPS)	Has no effect on subsequent GLC presentation
Transparent Data (TRN)	Has no effect on subsequent GLC presentation

The following examples demonstrate various valid examples of glyph layout control chaining:

- Example 1. GLC chain without optional controls. The GLC may reference the base font, or any font linked to the current font. Since all glyphs are positioned on the current effective baseline, the GOR control sequence is omitted:

  GLC GIR GAR <<chain ends>>

- Example 2. GLC chain with optional controls. Since one or more glyphs must be positioned with an offset from the baseline, the optional GOR control sequence is included in this chain. The PTOCA chain is terminated by an optional UCT control which carries the Unicode encoded text presented by the glyph layout control sequences:

  GLC GIR GAR GOR UCT <<chain ends>>

- Example 3. GLC chain with multiple GIR/GAR[/GOR] groupings. The text required more glyphs than a single GIR could list, so multiple GIR/GAR[/GOR] groupings are included. Since one or more glyphs in the first grouping must be positioned with an offset from the baseline, the optional GOR control sequence is included in this group. The optional GOR is not needed in the second grouping, and is omitted. The PTOCA chain is terminated by an optional UCT control which carries the Unicode encoded text presented by the glyph layout control sequences:

  GLC GIR GAR GOR GIR GAR UCT <<chain ends>>

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-9C00...Font Mismatch. The object OID specified in the GLC control sequence does not match the object OID of the current font or any font linked to the current font. If the OID matched the OID of a font collection, the Full Font Name did not match any font in the collection.
- EC-9C01...Font format not valid for use with glyph layout control sequences. The current font is not a TrueType/OpenType font.
- EC-9C03...Unexpected control sequence. An unexpected control sequence was encountered between the GLC control sequence and the terminating control sequence.

- EC-9C09...Missing font OID. The GLC specified an OIDLGTH of zero, but no previous font OID was supplied within the text object.
- EC-9C0A...Count mismatch or invalid length. The byte count specified by the OIDLGTH and FFONTNME parameters is not consistent with the overall control sequence length, or the OIDLGTH or FFNLGTH parameters are outside their valid range.
- EC-9C0B...Full Font Name specified without font OID. A font OID was not specified (OIDLGTH = 0), but a Full Font Name was specified (FFNLGTH ≠ 0).

Glyph Offset Run (GOR)

glyph offset run

The Glyph Offset Run control sequence specifies an offset (relative displacement) from the current baseline (in the b-direction) to the glyph origin for each glyph ID in the preceding GIR.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	UBIN	LENGTH	X'04' – X'FE'; even values only	Control sequence length. The length is 4 + n·(size of OFFSET)	M	N	N
1	CODE	TYPE	X'8E', X'8F'	Control sequence function type	M	N	N
2-3			X'0000'	Reserved; should be zero	M	N	N
Zero or more offsets from the current baseline in the following format:							
+0-1	SBIN	OFFSET	X'8000' – X'7FFF'	Glyph offset from the current baseline	O	N	N

Semantics

This control sequence carries a sequence of glyph offsets that correspond to the glyph IDs in the preceding GIR. Each offset is a 2-byte signed displacement from the current baseline in the b-axis direction to the glyph origin for each glyph ID. Application of the offset does not change the position of the current baseline. That is, the offset for each glyph ID is applied against the baseline that was defined at the start of the GLC chain. The first offset corresponds to the first glyph ID and the first glyph advance, the second offset corresponds to the second glyph ID and the second glyph advance, and so forth. Offsets are specified using the current measurement units. A positive offset is measured towards the i-axis; a negative offset is measured away from the i-axis.

Architecture Note: The direction in which GOR offsets are measured, as indicated by the sign of the offset, is opposite to the direction in which increments in a RMB are measured, as indicated by the sign of the increment.

The GOR control sequence is optional, but if present, must be chained to a GAR. It can be followed by a chained GIR or UCT. Composition applications must specify the same number of glyph offsets as glyph advances so that presentation devices can offset the correct glyph.

Pragmatics

If a composition application needs to offset one glyph, it must offset all the glyphs identified in the preceding GIR.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-9C03...Invalid sequence. The GOR is not preceded by a GAR, or, if it indicates chaining, is not followed by a GIR or a UCT.
- EC-9C06...GIR, GAR, or GOR control sequence found outside of a GLC chain. A GIR, GAR, or GOR was found that is not part of a GLC chain.

- EC-9C08...Glyph Offset count mismatch. The number of glyph offsets specified is not the same as the number of glyph IDs specified in the preceding GIR.

No Operation (NOP)

The No Operation control sequence has no effect on presentation.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	2–255	Control sequence length	M	N	N
3	CODE	TYPE	X'F8' – X'F9'	Control sequence function type	M	N	N
4–256	UNDF	IGNDATA	Not applicable	Ignored data	O	N	N

Semantics

This control sequence specifies a string of bytes that are to be ignored.

Exception Conditions

Exception Conditions This control sequence can cause the following exception condition:

- None

Overstrike (OVS)

The Overstrike control sequence identifies text that is to be overstruck with a specified character.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	5	Control sequence length	M	N	N
3	CODE	TYPE	X'72' – X'73'	Control sequence function type	M	N	N
4	BITS	BYPSIDEN	See Semantics	Bypass identifiers	M	Y	Y
5–6	CODE	OVERCHAR	X'0000' – X'FFFF'	Overstrike identifiers	M	N	N

BYPSIDEN is a flag byte with no measurement units. The PTOCA default for BYPSIDEN is X'01'. OVERCHAR is defined as a one-byte or two-byte code point that, when coupled with the active coded font, specifies the character to be used for overstriking. Single-byte code points are loaded in byte 6. OVERCHAR does not accept the default indicator.

Semantics

Overstrike is accomplished with a pair of Overstrike control sequences. A beginning OVS with a non-zero value in BYPSIDEN bits 4-7 activates overstrike. An ending OVS with a zero value in BYPSIDEN bits 4-7 deactivates overstrike. The beginning OVS control sequence immediately precedes the field of text to be overstruck. It specifies the following:

- The overstrike character
- How to place the overstrike characters in relation to the characters in the text field
- Which controlled inline white space is to be overstruck

The text field to be overstruck, called the *overstrike field,* is delimited by the beginning OVS and either an ending OVS control sequence or the end of the Presentation Text object. The overstrike field is a sequential string of presentation text.

BYPSIDEN specifies which controlled inline white space within the overstrike field is to be overstruck. *Controlled inline white space* is that area of the presented line that contains no visible material due to movement of the presentation position in the I-direction caused by the following:

- Absolute Move Inline control sequence
- Relative Move Inline control sequence
- Space character or variable space character

Application Note: The following code points are normally used for the variable space character:

- X'40' in EBCDIC single-byte code pages
- X'20' in ASCII single-byte code pages
- X'4040' in EBCDIC double-byte code pages
- X'2020' in ASCII double-byte code pages

The following code points are used for the variable space character in TrueType/OpenType fonts that use a Unicode (UTF-16) encoding:
- X'0020'
- X'00A0'

Movement of the current inline position in the I-direction to or through a presentation position that already contains material to be overstruck creates controlled inline white space for the entire move in the I-direction. Not all inline white space is controlled.

White space resulting from non-printing characters (other than space characters or variable space characters) within the character set, from substitution of non-printing characters for unsupported characters, from intercharacter adjustment, or from the inline margin is not considered controlled inline white space.

Inline moves that cause the current addressable position to move in a direction opposite to the I-direction, that is, negative inline moves, do not cause overstrike.

BYPSIDEN is bit-encoded as follows.

BIT	MEANING
0-3	Reserved, that is, set to B'0' by generators and ignored by receivers
4	Bypass Relative Move Inline
5	Bypass Absolute Move Inline
6	Bypass space characters and variable space characters
7	No Bypass in Effect

Bits 0-3, Reserved

Reserved bits are set to B'0' by generators and ignored by receivers.

Bit 4, Bypass Relative Move Inline

A value of B'0' in this bit indicates that the controlled white space generated as a result of a Relative Move Inline control sequence is to be overstruck. A value of B'1' in this bit indicates that such controlled white space is not to be overstruck. It should be bypassed.

Bit 5, Bypass Absolute Move Inline

A value of B'0' in this bit indicates that the controlled white space generated as a result of an Absolute Move Inline control sequence is to be overstruck. A value of B'1' in this bit indicates that such controlled white space is not to be overstruck. It should be bypassed.

Bit 6, Bypass space characters

A value of B'0' in this bit indicates that the controlled white space generated as a result of space characters or variable space characters is to be overstruck. A value of B'1' in this bit indicates that such controlled white space is not to be overstruck. It should be bypassed.

Bit 7, No Bypass in Effect

A value of B'0' in this bit activates the other bypass flags. A value of B'1' in this bit indicates that the other bypass flags are overridden, and that all text and white space bounded by the OVS pair should be overstruck. If the value of BYPSIDEN is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

Implementation Note: Most IPDS printers have implemented X'FF' as the default indicator, which results in BYPSIDEN = X'01' - no bypass in effect. However, it could be argued that the proper default indicator is

X'0F', since BYPSIDEN bits 0-3 are reserved, should be set to zero by generators, and should be ignored by receivers. To avoid confusion, it is strongly recommended that a default indicator not be used for this parameter, and that the value X'01' is specified directly if the default is desired.

An *overstrike area* is that portion of the overstrike field for which text is actually overstruck. An overstrike area is delimited by the addressable position in the following cases.:

- A beginning OVS
- Either end of bypassed controlled inline white space
- Either end of a baseline move, which may be for the established baseline or for a temporary baseline
- The beginning of negative changes in the presentation position caused by inline moves or negative intercharacter adjustments
- Boundaries where violation causes truncation
- An ending OVS
- The end of the Presentation Text object

Additionally, the dimension in the positive I-direction of the overstrike field is defined by the minimum and maximum I-coordinates specified between the overstrike area delimiters. White space resulting from the application of the inline margin is overstruck only if this area is entered by means of an inline move.

Overstrike characters are presented side by side without regard to the position of the characters being overstruck. Intercharacter adjustments are not applied to the placement of overstrike characters. The number of overstrike characters required for each overstrike area within an overstrike field is equal to the inline dimension of the overstrike area divided by the character increment of the overstrike character. If the result of this division is less than one, one overstrike character must be presented in the overstrike area. If the result of this division is greater than one and is not an integer, the decision to place an overstrike character is based on rounding to the nearest integer. Normally this rounding is down to the next lower integer, except as specified in the Pragmatics section.

If the value of OVERCHAR is not supported or is not within the range specified by PTOCA, an exception condition exists. See the Pragmatics section for the exception condition code and the standard action.

Pragmatics

The intent of the semantic is to distribute the overstrike characters evenly in the overstrike area without violating the delimiters of the overstrike area, that is, the I-coordinates at the beginning and end of the overstrike area.

Presentation of a portion of an overstrike character is acceptable to avoid overlapping characters.

Characters are overstruck using the current baseline. That is, the overstrike function follows the current baseline even when the baseline is changed by a Temporary Baseline Move control sequence.

OVERCHAR is defined as a one-byte or two-byte code point that, when coupled with the active coded font, specifies the character to be used for overstriking. Single-byte code points are located in byte 6.

The selected overstrike character must be a printable character and must specify a positive, non-zero character increment. If these conditions are not met by the selected overstrike character, exception condition EC-9A01 exists. To avoid an overflow of the overstrike field by the last overstrike character, the character increment of the overstrike character should also be equal to or greater than the character box size. If this is not true, exception condition EC-9A01 may optionally be detected.

Multiple beginning and ending Overstrike pairs may not be nested. However, no exception condition exists if a beginning OVS is processed when another OVS is already active. The subsequent beginning OVS terminates the previous OVS sequence and starts another. If an ending OVS is encountered when there has been no previous OVS in the Presentation Text object, no exception condition exists. Ignore the ending OVS. If a

Presentation Text object contains a beginning OVS without a matching ending OVS, no exception condition exists. Terminate the OVS at the end of the Presentation Text object.

There is no provision in this control sequence to specify a coded font, and Set Coded Font Local control sequences that occur within the overstrike field do not affect the appearance of the overstrike character. If the desired overstrike character is not defined in the active font which is controlling the text presentation, the beginning Overstrike control sequence must be preceded by a Set Coded Font Local control sequence that specifies a different coded font that contains the overstrike character. In this case, the coded font controlling the text presentation must be re-established following the ending Overstrike control sequence. If the graphic character specified in the OVS control sequence is not valid in the active coded font, exception condition EC-2100 exists. The standard action is to use a device default character as the overstrike character. If the graphic character does not have a rotation available that is equivalent to the current character rotation, exception condition EC-3F02 exists. The standard action is to accomplish the overstrike function to the best of the receiver's ability.

There are no syntactic restrictions on the occurrence of Begin Line, Absolute Move Baseline, Relative Move Baseline, and Temporary Baseline Move control sequences within the overstrike field.

Color is not a parameter of this control sequence.

Utilization of algorithms to reposition the overstrike characters within the overstrike areas of an overstrike field is restricted in several ways.

- At least one overstrike character must occur in an overstrike area.
- The overstrike characters must be positioned relative to the delimiters of the overstrike area, rather than to the last overstrike character in any previous overstrike area.
- Overlap of any portion of the B-space of the overstrike character with the B-space of a character not within the overstrike area is considered unacceptable, except in the case of a single character.
- Overlap of any portion of the B-space of the overstrike character with the B-space of another overstrike character within the overstrike field should be avoided. If overlap occurs, it should not be allowed to exceed 1/4 the B-space of either of the characters.
- Rounding off the number of overstrike characters is permitted. The minimum required support is to round off to the nearest integer number of overstrike characters in the overstrike area. This overlap may be allowed at a change of baseline, except when the overlap causes a portion of the character box to extend beyond a boundary of the object space.
- Multiple passes over the same portions of the presentation space through the use of AMI and RMI control sequences are not restricted.
- An area of zero length is not considered to be a valid overstrike area.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-2100...The graphic character specified is not valid in the active font.
- EC-3F02...The graphic character specified does not have a rotation available that is equivalent to the current character rotation.
- EC-9A01...The graphic character specified has an invalid character increment or is not a printable character.

Relative Move Baseline (RMB)

The Relative Move Baseline control sequence moves the baseline coordinate relative to the current baseline coordinate position.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4	Control sequence length	M	N	N
3	CODE	TYPE	X'D4' – X'D5'	Control sequence function type	M	N	N
4–5	SBIN	INCRMENT	X'8000' – X'7FFF'	Increment	M	N	N

INCRMENT is a signed binary number in measurement units. It does not accept the default indicator. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in “Interpreting Ranges” on page 45.

Semantics

This control sequence specifies an increment, INCRMENT, in the B-direction from the current baseline coordinate position to a new baseline coordinate position. After execution of this control sequence, presentation is resumed at the new baseline coordinate position. A positive value causes movement in the B-direction, while a negative value causes movement toward the I-axis. This control sequence does not modify the current inline coordinate position.

$I_{cnew} = I_c$

$B_{cnew} = B_c$ + INCRMENT where 0 <= B_{cnew} <= B-extent

If the value of INCRMENT is not supported or is not within the range specified by PTOCA, exception condition EC-1601 exists. The standard action is to continue presentation according to the description given in the Pragmatics section.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

If the value of INCRMENT is zero, the addressable position is not displaced, and any intercharacter increment is not applied. If a move is to a presentation position for which the character's character box will exceed the object space and an attempt is made to present there, exception condition EC-0103 exists. The standard action is to refrain from presenting the character that exceeds the object space, and to continue processing without presenting characters until the presentation position occupies a valid addressable position for the character being presented. Then presentation of characters may resume. PTOCA does not constrain advancement of the baseline coordinate in the negative B-direction, that is, toward the I-axis. However, a constraint of this type may be imposed by the subset level or by the receiver. If this constraint is applied and

INCRMENT is negative, exception condition EC-1403 exists. The standard action is to ignore the control sequence.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1601...The value of INCRMENT is not supported or is not in the range specified by PTOCA.
- EC-0103..The presentation position is outside the object space and presentation is attempted.
- EC-1403...Negative INCRMENT is not valid.

Relative Move Inline (RMI)

The Relative Move Inline control sequence moves the inline coordinate of the presentation position relative to the current inline position.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4	Control sequence length	M	N	N
3	CODE	TYPE	X'C8' – X'C9'	Control sequence function type	M	N	N
4–5	SBIN	INCRMENT	X'8000' – X'7FFF'	Increment	M	N	N

INCRMENT is a signed binary number in measurement units. It does not accept the default indicator. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in “Interpreting Ranges” on page 45.

Semantics

This control sequence specifies an increment, INCRMENT, in the I-direction from the current inline coordinate position to a new inline coordinate position. After execution of this control sequence, presentation is resumed at the new inline coordinate position. A positive value is in the direction of line growth, while a negative value logically backspaces. This control sequence does not modify the current baseline coordinate position.

$I_{cnew} = I_c + \text{INCRMENT}$ where $0 <= I_{cnew} <=$ I-extent

$B_{cnew} = B_c$

If the value of INCRMENT is not supported or is not within the range specified by PTOCA, exception condition EC-1501 exists. The standard action is to continue presentation according to the description given in the Pragmatics section.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

If the value of INCRMENT is zero, the addressable position is not moved, and any intercharacter adjustment is not applied. If a move is to a presentation position for which the character's character box will exceed the object space and an attempt is made to present there, exception condition EC-0103 exists. The standard action is to refrain from presenting the character that exceeds the object space, and continue processing without presenting characters until the presentation position occupies a valid addressable position for the character being presented. Then presentation of characters may resume.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1501...The value of INCRMENT is not supported or is not in the range specified by PTOCA.
- EC-0103..The presentation position is outside the object space and presentation is attempted.

Repeat String (RPS)

Syntax

The Repeat String control sequence contains a string of graphic character code points that is repeated on the current line.

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4–255	Control sequence length	M	N	N
3	CODE	TYPE	X'EE' – X'EF'	Control sequence function type	M	N	N
4–5	UBIN	RLENGTH	0–32,767	Repeat length	M	N	N
6–256	CHAR	RPTDATA	Not applicable	Repeated data	O	N	N

The contents of RPTDATA are unknown. RLENGTH is a binary number expressing byte count. RLENGTH and RPTDATA do not accept the default indicator.

Semantics

This control sequence specifies a string of bytes that is to be processed entirely as graphic character code points. No code point is recognized as a Control Sequence Prefix. Current inline position is incremented for each graphic character specified by a code point in the string. The baseline position is not changed.

For graphic characters following each other:

$I_{cnew} = I_c$ + intercharacter adjustment + character increment

Intervening non-incrementing characters are ignored.

For graphic characters following RMI, AMI, or BLN control sequences or following a space character or variable space character:

$I_{cnew} = I_c$ + character increment

Intervening non-incrementing characters are ignored.

For the variable space character:

$I_{cnew} = I_c$ + VSI

For non-incrementing characters:

$I_{cnew} = I_c$

In all cases:

$B_{cnew} = B_c$

RPTDATA is repeated until it fills the number of bytes specified by RLENGTH. If the number of bytes specified by RLENGTH is not an integral multiple of the length of the data, a portion of the data is truncated. If the number of bytes specified by the value of RLENGTH is less than the length of the data, a portion of the data is truncated. If the number of bytes specified by RLENGTH is equal to the data provided, the Repeat String control sequence has the same function as the Transparent Data control sequence.

When a double-byte font is active and the length of RPTDATA is an odd number, exception condition EC-1A01 exists. The standard action is to ignore the Repeat String control sequence and continue processing. When a double-byte font is active and RLENGTH is an odd number, exception condition EC-1B01 exists. The standard

action is to ignore the Repeat String control sequence and continue processing. If the value of RLENGTH is not supported or is not within the range specified by PTOCA, exception condition EC-1901 exists. The standard action is to ignore the control sequence and continue presenting.

The subset may limit the range permitted in this control sequence. For detailed information about function subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

The standard action value for RLENGTH is the length of RPTDATA. If any part of a character's character box will exceed the object space as a result of the Repeat String control sequence and an attempt is made to present the string of characters, exception condition EC-0103 exists. The standard action is not to present the character that exceeds the object space and to continue processing without presenting graphic characters until the presentation position is returned to an addressable position that is a valid presentation position for the character being presented.

If the value of the control sequence length parameter is four, indicating a length of four bytes, and the value of RLENGTH is zero, that is, there are no data bytes, this control sequence has the effect of a no-operation. If the value of the control sequence length parameter is greater than four, that is, data bytes are provided, and if the value of RLENGTH is zero, no exception condition exists and the data bytes are ignored. If, however, the value of the control sequence length parameter is four and RLENGTH is not zero, exception condition EC-1F01 exists. The standard action is to ignore this control sequence and continue processing.

If the character encoding is Unicode but the code points in the RPS are not valid Unicode data, exception condition EC-1A03 exists. The standard action is to skip the remainder of the code points in the RPS, repeat only the valid code point sequence, and continue processing.

Architecture Note: If the length of RLENGTH is the same as the length of RPTDATA, the RPS control sequence has the same effect as the Transparent Data control sequence. It is recommended that the RPS control sequence not be used in this manner, and that the TRN control sequence be used instead.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1A01...A double-byte font is active and the length of RPTDATA is an odd number.
- EC-1A03...Invalid Unicode data. This can be caused by one of the following:
  - A high-order surrogate code value was not immediately followed by a low-order surrogate code value. The high-order surrogate range is U+D800 through U+DBFF.
  - A low-order surrogate code value was not immediately preceded by a high-order surrogate code value. The low-order surrogate range is U+DC00 through U+DFFF.
  - An illegal UTF-8 byte sequence, as defined in the Unicode 3.2 Specification, was specified. For more information on illegal UTF-8 byte sequences, see Table 16 on page 120.
- EC-1B01...A double-byte font is active and RLENGTH is an odd number.
- EC-1901...The value of RLENGTH is not supported or is not in the range specified by PTOCA.
- EC-0103...A parameter value will cause part of a character's character box to be outside the object space, and presentation is attempted.
- EC-1F01...The control sequence length parameter is four and RLENGTH is not zero.

Set Baseline Increment (SBI)

The Set Baseline Increment control sequence specifies the increment to be added to the current baseline coordinate when a Begin Line control sequence is executed. This is a modal control sequence.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4	Control sequence length	M	N	N
3	CODE	TYPE	X'D0' – X'D1'	Control sequence function type	M	N	N
4–5	SBIN	INCRMENT	X'0000' – X'7FFF'	Increment	M	Y	Y

INCRMENT is a positive binary number expressed in measurement units. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45. The PTOCA default value for INCRMENT should be the Default Baseline Increment of the default coded font for the device.

Semantics

This control sequence specifies an increment, INCRMENT, in the positive B-direction from the current baseline coordinate position to a new established baseline coordinate position for subsequent presentation text in the current Presentation Text object. INCRMENT is applied when a Begin Line control sequence is executed. If the value of INCRMENT is not supported or is not within the range specified by PTOCA, exception condition EC-1101 exists. The standard action is to ignore this control sequence and continue presentation with the value determined according to the hierarchy. If the value of INCRMENT is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

If INCRMENT is omitted, the standard action is to make no change to the existing Baseline Increment parameter.

If this control sequence is omitted, the Baseline Increment initial text condition parameter in the Presentation Text Data Descriptor (PTD) is used. If this initial text condition is not specified, a receiver default based on the receiver default font should be used.

Note: The baseline increment, whether specified by the SBI control sequence, by the Baseline Increment initial text condition, or by a receiver default, is not affected by the active font or by changes to the active font.

Implementation Note: Most IPDS printers use a default baseline increment of 240/1440 inch = 1/6 inch if this parameter is not specified in an SBI control sequence or as an initial text condition in the PTD.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, "Compliance with PTOCA", on page 145, Appendix A, "MO:DCA Environment", on page 155, and Appendix B, "IPDS Environment", on page 161. See "Related Publications" on page vi for data-stream documentation.

Pragmatics

The baseline coordinate position is incremented by INCRMENT after each Begin Line control sequence is processed. If the value of INCRMENT is zero, each line appears superimposed over the preceding line.

This control sequence overrides the Baseline Increment initial text condition parameter that may occur in the Presentation Text Data Descriptor.

Exception Conditions

This control sequence can cause the following exception condition:

- EC-1101...The value of INCRMENT is not supported or is not in the range specified by PTOCA.

Set Coded Font Local (SCFL)

The Set Coded Font Local control sequence activates a coded font and specifies the character attributes to be used. This is a modal control sequence.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	3	Control sequence length	M	N	N
3	CODE	TYPE	X'F0' – X'F1'	Control sequence function type	M	N	N
4	CODE	LID	X'00' – X'FE'	Local identifier	M	Y	Y

The PTOCA default value for the LID is X'00'.

Semantics

This control sequence specifies a local identifier, LID, which is used by the controlling environment to access a coded font for presentation of subsequent text in the current Presentation Text object. The current presentation position is not changed by this control sequence. If the value of the LID is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

If the LID is omitted, exception condition EC-1E01 exists. The standard action in this case is to ignore the control sequence and continue presentation with the active font determined according to the hierarchy. If the value of the LID is not supported or is not within the range specified by PTOCA, exception condition EC-0C01 exists. The standard action is to ignore the control sequence and continue presentation with the active coded font determined according to the hierarchy.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

The LID is equated to a coded font, character rotation, and font modification parameters by a mapping function in the controlling environment. Please see “Related Publications” on page vi and “Font Concepts” on page 13 for font documentation.

PTOCA expects the local identifier, LID, to be mapped to a global identifier. For example, this mapping could be accomplished in the following ways:

- The receiver provides internal mapping, using device defaults.
- The controlling environment provides the mapping to the receiver.

If a mapping is not provided, exception condition EC-1802 exists. The standard action is to ignore the control sequence, substitute a coded font determined according to the hierarchy, and continue processing. If the coded font specified by the mapping is not available to the receiver, exception condition EC-1802 exists. The standard action in this case is to substitute a coded font, determined according to the hierarchy, for the

specified coded font and continue processing. The PTOCA parameter specification hierarchy is defined in Table 10 on page 33.

If the text orientation is changed and the specified coded font is not compatible with the new orientation, when an attempt is made to present a character from the selected coded font, exception condition EC-3F02 exists. The standard action is to skip the presentation at this presentation position and use the active coded font determined according to the hierarchy. This results in the best possible presentation within the receiver's capability. In this case, any intercharacter adjustment is not applied.

The measurement units used by the medium and those used by the coded font selected for presentation are assumed to be compatible. If they are not compatible, the standard action, when not superseded by the controlling environment, is to present a best fit of the character and continue processing. A best fit could result in no mark or unintelligible marks on the presentation surface. However, incompatible measurement units are not an exception condition in PTOCA.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1E01...The LID is missing.
- EC-0C01...The value of the LID is not supported or is not in the range specified by PTOCA.
- EC-1802...A font mapping has not been provided.
- EC-1802...The coded font specified by the mapping is not available to the receiver.
- EC-3F02...The specified coded font is not compatible with the text orientation.

Set Extended Text Color (SEC)

The Set Extended Text Color control sequence specifies a color value and defines the color space and encoding for that value. The specified color value is applied to foreground areas of the text presentation space. Foreground areas consist of the following:

- The stroked and filled areas of solid text characters, including overstrike characters; with hollow characters, only the stroked portion of the character is considered foreground.
- The stroked area of a rule
- The stroked area of an underscore

All other areas of the text presentation space are considered background.

This is a modal control sequence.

Note: Colors may be specified using the Set Text Color (STC) or the Set Extended Text Color (SEC) control sequences. Both STC and SEC can coexist in the same text object. The last issued control sequence determines the current text color in accordance with the rules defined for modal control sequences. For a definition of modal control sequences, see "Modal Control Sequences" on page 35.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	14–16	Control sequence length	M	N	N
3	CODE	TYPE	X'80' – X'81'	Control sequence function type	M	N	N
4				Reserved; should be zero	M	N	N
5	CODE	COLSPCE	X'01' X'04' X'06' X'08' X'40'	Color space RGB CMYK Highlight color space CIELAB Standard OCA color space	M	N	N
6–9				Reserved; should be zero	M	N	N
10	UBIN	COLSIZE1	X'01' – X'08', X'10'	Number of bits in component 1, see color space definitions	M	N	N
11	UBIN	COLSIZE2	X'00' – X'08'	Number of bits in component 2, see color space definitions	M	N	N
12	UBIN	COLSIZE3	X'00' – X'08'	Number of bits in component 3, see color space definitions	M	N	N

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
13	UBIN	COLSIZE4	X'00' – X'08'	Number of bits in component 4, see color space definitions	M	N	N
14–(*n*-1)		COLVALUE	See Semantics for details	Color specifications	M	N	N

Semantics

COLSPCE is a code that defines the color space and the encoding for the color specification. If the color space is invalid or unsupported, exception condition EC-0E02 exists. The standard action is to use the device default color.

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, and 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 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, and B components are assumed to be gamma-corrected (nonlinear) 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, and K components is 0 to 1, which is mapped to the binary value range 0 to ($2^{ColSizeN}$ - 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 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 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.

If the percent value for component 2 or component 3 is invalid, exception condition EC-0E04 exists. The standard action is to use the maximum valid percent value.

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. In addition, presentation devices may not support the % coverage and % shading parameters for highlight colors in PTOCA text. In that case, these parameters are simulated with 100% coverage and 0% shading, respectively.
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) or an Indexed (IX) Color Management Resource (CMR). When used with an Indexed CMR, component 1 specifies a two-byte value that is the index into the CMR, and components 2 and 3 are ignored . Note that when both a CMT and Indexed CMRs are used, the CMT is always accessed first. To preserve compatibility with existing highlight color devices, indexed color values X'0000' – X'00FF' are reserved for existing highlight color applications and devices. That is, indexed colors values in the range X'0000' – X'00FF', assuming they are not mapped to a different color space in a CMT, are mapped directly to highlight colors. Indexed color values in the range X'0100' – X'FFFF', assuming they are not mapped to a different color space in a CMT, are used to access Indexed CMRs. For a description of the Color Mapping Table and Indexed CMRs in MO:DCA environments, see the *Mixed Object Document Content Architecture Reference,* AFPC-0004.

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 $-(2^{ColSizeN-1} - 1)$ to $+(2^{ColSizeN-1} - 1)$.

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 due to 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 the *Mixed Object Document Content Architecture Reference,* AFPC-0004. 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. The following table defines the valid color values used to specify named colors. The table also specifies the RGB values that can be used 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.

Table 13. SEC Color Values

Value	Color	Red (R)	Green (G)	Blue (B)
X'0000' or X'FF00'	Presentation-process default; see Note 1 on page 87			
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 87	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 87	—	—	—

Table 13 SEC Color Values (cont'd.)

Value	Color	Red (R)	Green (G)	Blue (B)
X'FF08'	Color of medium	—	—	—

Notes:

1. The presentation-process default specified by X'0000' and X'FF00' is resolved as follows:
   - For PTOCA text 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.
5. While the RGB values in the table can be used to render the OCA named colors, many implementations are and have been device-dependent. Nevertheless, it is recommended that OCA Black (X'0008') be rendered as C = M = Y = X'00', and K = X'FF'.

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. If the specified value is invalid or unsupported, exception condition EC-0E05 exists. The standard action is to use the device default color.

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. If the specified value is invalid or unsupported, exception condition EC-0E05 exists. The standard action is to use the device default color.

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. If the specified value is invalid or unsupported, exception condition EC-0E05 exists. The standard action is to use the device default color.

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. If the specified value is invalid or unsupported, exception condition EC-0E05 exists. The standard action is to use the device default color.

COLOR VALUE specifies the color value in the defined format and encoding. If the color value is invalid or unsupported, exception condition EC-0E03 exists. The standard action is to use the device default color.

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 control sequence length is valid.

Architecture Note: For a description of color spaces and their relationships, see Hunt, R., *The Reproduction of Colour in Photography, Printing, and Television*, Fifth Edition, Fountain Press, 1995.

The subset may limit the parameter ranges permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

If the receiver does not support the specified color value exception condition EC-0E03 exists. The standard action in this case is to use the presentation device default color.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-0E02...Invalid or unsupported color space
- EC-0E03...Invalid or unsupported color value
- EC-0E04...Invalid percent value
- EC-0E05...Invalid or unsupported number of bits in a color component

Set Intercharacter Adjustment (SIA)

The Set Intercharacter Adjustment control sequence specifies additional increment or decrement between graphic characters. This is a modal control sequence.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4–5	Control sequence length	M	N	N
3	CODE	TYPE	X'C2' – X'C3'	Control sequence function type	M	N	N
4–5	SBIN	ADJSTMNT	X'0000' – X'7FFF'	Adjustment	M	Y	Y
6	CODE	DIRCTION	X'00' – X'01'	Direction	O	Y	Y

ADJSTMNT is a positive binary number expressed in measurement units. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45. The PTOCA default value for ADJSTMNT is X'0000'. DIRCTION is a code with no measurement units. The PTOCA default value for DIRCTION is X'00'.

Semantics

ADJSTMNT specifies the value of additional space between graphic characters. This space is in the I-direction from the end of the current character increment to the presentation position of the following graphic character. When this value is positive, the adjustment is called an increment. When the value is negative, the adjustment is called a decrement. DIRCTION specifies the direction in which the intercharacter adjustment is to be applied. Intercharacter increment, which occurs when DIRCTION is X'00', is applied in the positive I-direction. Intercharacter decrement, which occurs when DIRCTION is X'01', is applied in the negative I-direction. This control sequence does not change the current presentation position.

For graphic characters following each other:

$I_{cnew} = I_c + ADJSTMNT + CI$

For a graphic character following a RMI, AMI, BLN control sequence or following a space character or variable space character:

$I_{cnew} = I_c + CI$

For a non-incrementing character:

$I_{cnew} = I_c$

For the variable space character:

$I_{cnew} = I_c + VSI$

In all cases:

$B_{cnew} = B_c$

If intercharacter adjustment is valid for a given graphic character, it is applied before presenting the character. If it is an incrementing character, the current inline coordinate is incremented first by the intercharacter adjustment, and then by the character increment. The same is true for a non-incrementing character, but the intercharacter adjustment is inhibited for the character that follows the non-incrementing character. The result

is that the non-incrementing character is coupled with the following graphic character. This accomplishes an overstrike function, and the intercharacter adjustment is applied to the coupled characters as a unit.

Intercharacter adjustment is not applied before or after the following:
- A space character or variable space character
- A Begin Line control sequence
- A Relative Move Inline control sequence
- An Absolute Move Inline control sequence

Non-presenting characters are graphic characters except when identified as the designated variable space character.

Intercharacter adjustment is inserted between the characters that form a word, but not between words. That is, intercharacter adjustment is applied only when in *inword mode.* Inword mode is entered after any incrementing character has been processed. Inword mode is exited after any *word delimiter* has been processed. The following are word delimiters:
- The space character or variable space character
- Begin Line control sequences
- Relative Move Inline control sequences
- Absolute Move Inline control sequences

Application Note: The following code points are normally used for the variable space character:
- X'40' in EBCDIC single-byte code pages
- X'20' in ASCII single-byte code pages
- X'4040' in EBCDIC double-byte code pages
- X'2020' in ASCII double-byte code pages

The following code points are used for the variable space character in TrueType/OpenType fonts that use a Unicode (UTF-16) encoding:
- X'0020'
- X'00A0'

If the value of ADJSTMNT is the default indicator, a value is obtained from the hierarchy. If the value of DIRCTION is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

If the value of ADJSTMNT or DIRCTION is not supported or is not within the range specified by PTOCA, exception condition EC-1201 exists. The standard action is to ignore this control sequence and continue presentation with the parameter values determined according to the hierarchy.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

If the value of ADJSTMNT is zero, no additional intercharacter increment or decrement appears between graphic characters. In this case, DIRCTION is optional, and the SIA control sequence does not change the following:
- The current presentation position
- The current I-unit value
- The current inline margin
- The current intercharacter increment value
- The current intercharacter decrement value

Exception Conditions

This control sequence can cause the following exception condition:

- EC-1201...The value of ADJSTMNT or DIRCTION is not supported or is not in the range specified by PTOCA.

Set Inline Margin (SIM)

The Set Inline Margin control sequence specifies the position of an inline margin. This is a modal control sequence.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4	Control sequence length	M	N	N
3	CODE	TYPE	X'C0' – X'C1'	Control sequence function type	M	N	N
4–5	SBIN	DSPLCMNT	X'0000' – X'7FFF'	Displacement	M	Y	Y

DSPLCMNT is a positive binary number expressed in measurement units. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45. The PTOCA default value for DSPLCMNT is the B-axis, that is, I_c is zero.

Semantics

This control sequence specifies a displacement, DSPLCMNT, from the B-axis in the I-direction that is to be applied when a Begin Line control sequence is processed in the current Presentation Text object. If the value of DSPLCMNT is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

If DSPLCMNT is omitted, exception condition EC-1E01 exists. The standard action is to ignore the control sequence and continue presentation with the Inline Margin that was in effect prior to this control sequence. If the value of DSPLCMNT is not supported or is not within the range specified by PTOCA, exception condition EC-1001 exists. The standard action is to ignore this control sequence and continue presentation with the value determined according to the hierarchy.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, "Compliance with PTOCA", on page 145, Appendix A, "MO:DCA Environment", on page 155, and Appendix B, "IPDS Environment", on page 161. See "Related Publications" on page vi for data-stream documentation.

This control sequence does not change the current addressable position.

Pragmatics

The current addressable position is at the inline margin after each Begin Line control sequence is executed. If the value of DSPLCMNT is zero, the inline margin is at the B-axis.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1E01...DSPLCMNT is missing.
- EC-1001...The value of DSPLCMNT is not supported or is not in the range specified by PTOCA.

Set Text Color (STC)

The Set Text Color control sequence specifies a color attribute for the foreground areas of the text presentation space. Foreground areas consist of the following:

- The stroked and filled areas of solid text characters, including overstrike characters; with hollow characters, only the stroked portion of the character is considered foreground.
- The stroked area of a rule
- The stroked area of an underscore

All other areas of the text presentation space are considered background.

This is a modal control sequence.

Note: Colors may be specified using the Set Text Color (STC) or the Set Extended Text Color (SEC) control sequences. Both STC and SEC can coexist in the same text object. The last issued control sequence determines the current text color in accordance with the rules defined for modal control sequences. For a definition of modal control sequences, see "Modal Control Sequences" on page 35.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4, 5	Control sequence length	M	N	N
3	CODE	TYPE	X'74' – X'75'	Control sequence function type	M	N	N
4–5	CODE	FRGCOLOR	See Semantics section	Foreground color	M	Y	Y
6				Retired parameter, see Architecture Note, also see "Retired Parameters" on page 169.	O		

The PTOCA default value for FRGCOLOR is X'FF07'. Please see the pragmatics section for further details.

Architecture Note: Pre-year 2000 applications and printers support an optional PRECSION parameter in byte 6. This parameter has been retired. It should not be generated by new applications, and should be ignored by new printers. For a definition of this parameter, see "Retired Parameters" on page 169 .

Semantics

The FRGCOLOR parameter specifies a color value. 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. An additional valid value is X'FFFF', which is the default indicator and specifies that the color value is obtained from the hierarchy. Please see the *Pragmatics* section, as well as Table 10 on page 33. The PTOCA default value for FRGCOLOR is X'FF07'. For a definition of the Standard OCA Color Value Table, see the *Mixed Object Document Content Architecture Reference*, AFPC-0004.

If the color is not supported, or if the FRGCOLOR value is not syntactically valid, exception condition EC-5803 exists. The standard action in this case is to use X'FF07'.

The following table defines the valid color values used to specify named colors. The table also specifies the RGB values that can be used 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.

Table 14. STC Color Values

Value	Color	Red (R)	Green (G)	Blue (B)
X'0000' or X'FF00'	Presentation-process default; see Note 1 on page 94			
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 94	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 94	—	—	—
X'FF08'	Color of medium	—	—	—
X'FFFF'	Default indicator	—	—	—

Notes:

1. The presentation-process default specified by X'0000' and X'FF00' is resolved as follows:
   - For PTOCA text 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.
5. While the RGB values in the table can be used to render the OCA named colors, many implementations are and have been device-dependent. Nevertheless, it is recommended that OCA Black (X'0008') be rendered as C = M = Y = X'00', and K = X'FF'.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

The presentation process default color attribute value (FRGCOLOR = X'FFFF') is determined hierarchically. The following order applies:

1. Value set by Text Color initial text condition parameter in descriptor
2. PTOCA default X'FF07'

The *device default value* is the receiver’s default. For example, characters, rules, and underscores will be presented in black on a receiver which supports only black. The receiver’s best possible value means that if the receiver has limited color capabilities, then it may substitute a color it supports for one it does not support. For example, the receiver may substitute red for pink, blue for turquoise, and so forth.

The color attribute values X'FF00' to X'FF06' are translated to X'0000' to X'0006' and treated exactly like those colors. The PTOCA default value of X'FF07', is the device default. For example, characters, rules, and underscores will be presented in black on a device which supports only black. A color attribute value of X'FF08' means that the receiver's default background color should be used for the foreground color. This provides an erase function.

Exception Conditions

This control sequence can cause the following exception condition:

- EC-5803...The value of FRGCOLOR is invalid, or the specified color is not supported.

Architecture Notes:

1. The MO:DCA environment supports a Color Mapping Table (CMT) that may be used to map colors in a PTOCA object to other colors. When a CMT is active, valid FRGCOLOR values are mapped to their target values. The retired PRECSION parameter, if supported, is processed for the target values.
2. The IPDS environment allows a presentation device to implement limited simulated color support for PTOCA text and rules. When the printer is working in a mode where color simulation is allowed, all valid but unsupported color values are accepted and result in a device-dependent simulation of the specified color without the generation of exception EC-5803.

Set Text Orientation (STO)

The Set Text Orientation control sequence establishes the I-direction and B-direction for the subsequent text. This is a modal control sequence.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	6	Control sequence length	M	N	N
3	CODE	TYPE	X'F6' – X'F7'	Control sequence function type	M	N	N
4–5	CODE	IORNTION	See Semantics section	I-axis orientation	M	Y	Y
6–7	CODE	BORNTION	See Semantics section	B-axis orientation	M	Y	Y

The PTOCA default for IORNTION is zero. The PTOCA default for BORNTION is 90.

Semantics

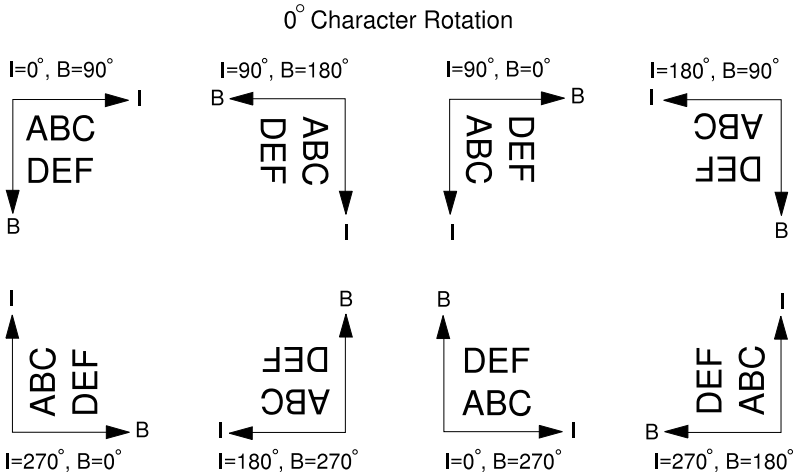
This control sequence specifies the I-axis and B-axis orientations with respect to the X_p-axis for the current Presentation Text object. The orientations are rotational values expressed in degrees and minutes. IORNTION and BORNTION have the same format. Each is a two-byte, three-part binary code of the form ABC.

- A is a nine-bit binary number (bits 0 - 8) which provides from 0 through 359 degrees. Values from 360 through 511 are invalid.
- B is a six-bit binary number (bits 9 - 14) which provides from 0 through 59 minutes. Values from 60 through 63 are invalid.
- C is a one-bit reserved field (bit 15) which must be 0.

The maximum value for IORNTION and BORNTION is X'B3F6' or B'1011001111110110', which is 359 degrees and 59 minutes. Increasing values indicate increasing clockwise rotation of the I,B axes with respect to the X_p, Y_p axes. A value of 0 for IORNTION indicates that there is no rotation relative to the X_p-axis. That is, positive I-direction is parallel to the X_p-axis.

The origin of the I,B axes is always one of the four corners of the object space. If the text orientation is changed, this origin may also change. See Figure 12 on page 97 for the location of the I,B origin for the eight text orientations that are supported by the PT1, PT2, and PT3 subsets. For example, if IORNTION and BORNTION are 0,90 or 90,0, the origin of the I,B axes is at the upper left corner, or origin, of the object space. This is where $X_p = 0$ and $Y_p = 0$. If IORNTION and BORNTION are 180,90 or 90,180, the origin of the I,B axes is at the upper right corner of the object space. This is where $X_p = X_p$-extent, and $Y_p = 0$.

Figure 12. Location of I,B Origin



If the inline direction changes or the origin of the B-axis changes, the inline margin also changes. The new inline margin is parallel to the new B-axis, and it is displaced in the new I-direction from the new B-axis according to the value of the Inline Margin parameter. If the value of IORNTION or BORNTION is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

If IORNTION or BORNTION is omitted, exception condition EC-1E01 exists. The standard action is to assume an orientation with the I-axis parallel to the X_p-axis, and the B-axis parallel to the Y_p-axis. If IORNTION and BORNTION are identical, exception condition EC-0F01 exists. The standard action is to assume an orientation with the I-axis parallel to the X_p-axis, and the B-axis parallel to the Y_p-axis.

Pragmatics

When the text orientation is changed, the text appears to rotate about the current presentation position, which is an X_p,Y_p coordinate. Please see Figure 13 on page 98 for examples of text orientation and character rotation. When such a change occurs, the current presentation position is not changed, but the values of I_c and B_c are adjusted to correspond to the current presentation position relative to the new orientation.

If neither the I-direction nor the B-direction is parallel to the X_p-direction or the Y_p-direction, exception condition EC-0F01 exists. The standard action is to set the I-direction parallel to the X_p-direction and the B-direction parallel to the Y_p-direction.

Orientations other than 0,90 are valid, but may be constrained by receiver limitations or by parameters in the controlling environment. If the orientation is not supported by the receiver, exception condition EC-0F01 exists. The standard action is to use 0,90 degrees for the orientation. This exception condition and standard action also apply to values for IORNTION and BORNTION not within the range specified by PTOCA.

Architecture Notes:

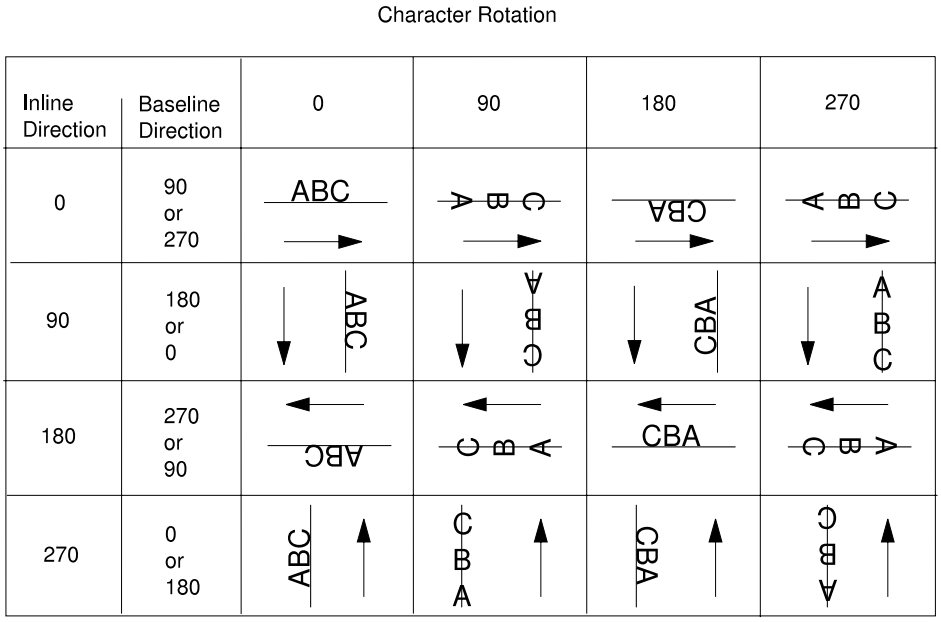
1. The following remain as previously specified:
   - The current presentation position, an Xp,Yp coordinate,
   - The current I-unit value,
   - The current inline margin,
   - The current intercharacter increment value,
   - The current intercharacter decrement value,
   - The current B-unit value,
   - The current baseline increment value,
   - The current coded font.

2. The following will change:
   - The X_p- Y_p-axis from which inline margin is measured (that is, the inline margin appears to rotate).
   - Font character rotations appropriate to the new orientation are used.
   - Presentation position should be respecified if subsequent text is to be positioned elsewhere in the Presentation Text space.
   - Other modal parameter values should be respecified if they are more appropriate to the new orientation.
   - A new coded font should be specified:
     - If the current coded font is not valid in the new text orientation,
     - If it is desired that the graphic characters be rotated in proper orientation with respect to the new baseline.
3. If the Presentation Text object measurement units specified for the X_p-axis are different from the measurement units specified for the Y_p-axis, the result of a Set Text Orientation control sequence may be unexpected and use of a Set Text Orientation control sequence should be avoided.

Exception Conditions

This control sequence can cause the following exception conditions:
- EC-1E01...IORNTION or BORNTION is missing.
- EC-0F01...IORNTION and BORNTION are identical.
- EC-0F01...IORNTION or BORNTION not parallel to X_p-axis or Y_p-axis.
- EC-0F01...IORNTION and BORNTION not supported by receiver.

Figure 13. Examples of Text Orientation and Character Rotation



Set Variable Space Character Increment (SVI)

The Set Variable Space Character Increment control sequence specifies the increment for a variable space character. This is a modal control sequence.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	4	Control sequence length	M	N	N
3	CODE	TYPE	X'C4' – X'C5'	Control sequence function type	M	N	N
4–5	SBIN	INCRMENT	X'0000' – X'7FFF'	Increment	M	Y	Y

INCRMENT is a positive number expressed in measurement units. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in “Interpreting Ranges” on page 45.

Semantics

This control sequence specifies an increment, INCRMENT, for the variable space character. The increment is in the I-direction from the presentation position of the variable space character to the addressable position for the next graphic character or control sequence for subsequent text in the current Presentation Text object. This control sequence does not change the current presentation position.

To return to the current coded font's default value for the variable space character increment, set INCRMENT to the default indicator. If the current coded font does not have such a default value, INCRMENT is set to the character increment for the default variable space character.

If INCRMENT is omitted, exception condition EC-1E01 exists. The standard action is to continue with the presentation using the standard action value for the variable space character increment. If the value of INCRMENT is not supported or is not within the range specified by PTOCA, exception condition EC-1701 exists. The standard action is to ignore this control sequence and continue presentation with the value determined according to the hierarchy. Please refer to the Pragmatics section for details.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

The inline coordinate of the presentation position is incremented by INCRMENT after each variable space character is processed. Each variable space character causes the presentation position to move in the I-direction by the amount of the variable space character increment. If the value of INCRMENT is zero, no variable space appears between words, and intercharacter adjustment is not applied even though the resulting characters appear side-by-side. When a Set Variable Space Increment control sequence is received, the new value of INCRMENT is saved and is applied to any subsequent variable space character received.

The code point used for the variable space character is specified, either implicitly or explicitly, by the active font. In this case, the default value for INCRMENT is the value specified by the active font in its current orientation. The value is obtained in this order:

1. The current variable space character increment
2. The default variable space character increment of the active coded font
3. The character increment of the default variable space character code point

When the value of INCRMENT changes because of changes in the font or the SVI control sequences, the new value is carried but is not used until the variable space character is enabled.

The variable space character increment is not effective for other graphic characters that are not presented, nor for graphic characters that make no marks.

Architecture Note: The following remain as previously specified:

- The current presentation position, an Xp,Yp coordinate
- The current I-unit value
- The current inline margin,
- The current intercharacter increment value
- The current intercharacter decrement value
- The current B-unit value
- The current baseline increment value
- The current coded font

Application Note: The following code points are normally used for the variable space character:

- X'40' in EBCDIC single-byte code pages
- X'20' in ASCII single-byte code pages
- X'4040' in EBCDIC double-byte code pages
- X'2020' in ASCII double-byte code pages

The following code points are used for the variable space character in TrueType/OpenType fonts that use a Unicode (UTF-16) encoding:

- X'0020'
- X'00A0'

Application Note: The Set Variable Space Character Increment (SVI) control sequence is modal, but is not an initial text condition. For text major text, SVI is not reset by AFP presentation servers when a MO:DCA BPT structured field is encountered, and therefore later text on the same page will inherit any SVI set previously on the page. If no SVI is specified in a page, then the default variable space character increment of the active coded font is used.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1E01...INCRMENT is missing.
- EC-1701...The value of INCRMENT is not supported or is not in the range specified by PTOCA.

Temporary Baseline Move (TBM)

The Temporary Baseline Move control sequence changes the position of the baseline without changing the established baseline.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	3, 4, 6	Control sequence length	M	N	N
3	CODE	TYPE	X'78' – X'79'	Control sequence function type	M	N	N
4	CODE	DIRCTION	X'00' – X'03'	Direction	M	Y	Y
5	BITS	PRECSION	X'00' – X'01'	Precision	O	Y	Y
6–7	SBIN	INCRMENT	X'0000' – X'7FFF'	Temporary baseline increment	O	Y	Y

INCRMENT is a positive number expressed in measurement units. The range for this parameter assumes a measurement unit of 1/1440 inch. If it is necessary to convert to different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45. The PTOCA default value for INCRMENT is 1/2 the baseline increment value. The PTOCA default value for DIRCTION and PRECSION is zero.

Semantics

This control sequence does one of the following:

- Change the current baseline coordinate by the amount specified by INCRMENT in the direction specified by DIRCTION
- Return the baseline coordinate to the established baseline coordinate position

This control sequence does not change the position of the established baseline coordinate or the inline presentation position. INCRMENT specifies the magnitude of the temporary baseline increment. PRECSION specifies the method by which the receiver exhibits the change in the baseline coordinate. DIRCTION specifies the direction of the change.

DIRCTION is a bit-encoded value that specifies the following:

Value	Meaning
X'00'	Do not change the baseline.
X'01'	Return to the established baseline. Delete the temporary baseline created by TBM control sequences.
X'02'	Move the temporary baseline away from the I-axis one value of INCRMENT, performing a subscript function. The increment is applied to the current baseline coordinate, not to the established baseline, and has no effect on the established baseline.
X'03'	Move the temporary baseline toward the I-axis one value of INCRMENT, performing a superscript function. The increment is applied to the current baseline coordinate, not to the established baseline, and has no effect on the established baseline.

The following equations apply to DIRCTION:
If DIRCTION = 0:
$B_{cnew} = B_c$
If DIRCTION = 1:
$B_{cnew} = B_{est}$
If DIRCTION = 2:
$B_{cnew} = B_c + INCRMENT$
If DIRCTION = 3:
$B_{cnew} = B_c - INCRMENT$
In all cases:
$I_{cnew} = I_c$

PRECSION is bit-encoded as follows.

Bits 0-6

Bits 0-6 are reserved. They must be set to B'0' by generators, and ignored by receivers.

Bit 7

If bit 7 is B'0', the receiver must accurately place and represent the character using the coded font that is active when the control sequence is executed. In this case, the movement of the baseline is not simulated, and the character presented on the shifted baseline is the same as the characters used in the surrounding text. That is, it is not a character specially designed for subscripting or superscripting. However, this does not prohibit changing the coded font. If this bit is B'0', the intent is to ensure that the receiver has the word processing capability of producing formal documents.

If bit 7 is B'1', the movement of the baseline may be simulated using specially designed subscript or superscript characters which appear smaller than the surrounding text.

If the value of INCRMENT, PRECSION, or DIRCTION is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

If the value of INCRMENT, PRECSION, or DIRCTION is not supported or is not within the range specified by PTOCA, exception condition EC-9803 exists. The standard action is to ignore this control sequence and continue presentation with the value determined according to the hierarchy. Please see the Pragmatics section for details.

The subset may limit the range permitted in this control sequence. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Pragmatics

- How TBM operates:

  Changing the baseline coordinate with this control sequence does not modify the position of the established baseline coordinate.

  Once the baseline coordinate has been changed by a TBM control sequence, it will remain at the new location until it is terminated by another TBM control sequence or the end of the Presentation Text object.

  After processing a TBM control sequence, additional TBM control sequences are processed relative to the temporary baseline coordinate position, not the established baseline coordinate position. That is, a second TBM with magnitude and direction equal to the first TBM causes the temporary baseline coordinate to be

moved farther from the established baseline coordinate. A second TBM with magnitude equal to the first but opposite direction cancels the effect of the first TBM and terminates the temporary baseline function.

The Temporary Baseline Increment parameter is modal. That is, once it is set, it remains set until it is changed by another TBM control sequence. It is not necessary to generate a TBM control sequence in order to give this parameter a value. The parameter is required only if its value is to be changed. If a TBM control sequence does not include INCRMENT or specifies the default indicator for INCRMENT, the default value is used. The default is 1/2 the current baseline increment.

The temporary baseline may be canceled, that is, reset to the established baseline coordinate position, by specifying the “Return to Established Baseline” function. This is specified by setting DIRCTION equal to X'01'. If a TBM control sequence is processed with DIRCTION set to X'01' when a temporary baseline has not been established, the result is a no-operation. If a TBM control sequence specifies the Return to Established Baseline function but includes the Temporary Baseline Increment parameter INCRMENT, the baseline coordinate is returned to the established baseline coordinate position, and INCRMENT is changed to the new value specified. PRECSION has no effect on the Return to Established Baseline function.

The temporary baseline function is terminated if the temporary baseline coordinate coincides with the established baseline coordinate, that is, when the established baseline and the current baseline are equal at the end of processing the TBM control sequence. Therefore, creating a temporary baseline on one side of the established baseline followed by another temporary baseline on the other side of the established baseline without terminating the temporary baseline field is possible by changing the value of INCRMENT.

- How TBM affects other control sequences:

  The temporary baseline field is not canceled by any other control sequences. Here are some examples:

  A Begin Line control sequence is processed relative to the established baseline coordinate position, not the temporary baseline coordinate position. Following the processing of a Begin Line control sequence, new baseline coordinate positions are determined. The Baseline Increment is added to the current baseline coordinate position to provide a new current baseline position. The Baseline Increment is also added to the established baseline coordinate position to provide a new established baseline position. Following the processing of a Begin Line control sequence, the temporary baseline is continued until a terminating TBM control sequence ends the temporary baseline function.

  A Relative Move Baseline control sequence is processed relative to the established baseline coordinate position, not the temporary baseline coordinate position. Following the processing of a Relative Move Baseline control sequence, new baseline coordinate positions are determined. The relative baseline increment is added to the current baseline coordinate position to provide a new current baseline position. The relative baseline increment is also added to the established baseline coordinate position to provide a new established baseline position. Following execution of a Relative Move Baseline control sequence, the temporary baseline is continued until a terminating TBM control sequence ends the temporary baseline function.

- How TBM uses the PRECSION parameter:

  PRECSION specifies how the receiver should exhibit characters at the temporary baseline. There is an *actual placement* method and a *substitution* method. If PRECSION is X'01', a receiver may simulate the temporary baseline move by substituting subscript or superscript graphics for graphics actually positioned below or above the established baseline. This is the substitution method. If PRECSION is X'00', the receiver is required to support the physical shift in the baseline coordinate, and may not substitute special graphics for those specified at the temporary baseline. This is the actual placement method. Receivers must support one of these two methods of presentation.

  Receivers that implement the substitution method should support the other aspects of the TBM control sequence just as if an actual shift in the baseline coordinate had occurred. That is, the character increment of the active font, the current baseline, the established baseline, and the baseline increment must all be maintained, just as though the characters being used were from the active font and the shift of the current baseline had actually occurred.

If a receiver implements only the substitution method and PRECSION is set to X'00', exception condition EC-9803 exists. The standard action is to perform the substitution method. Furthermore, if a receiver using the substitution method cannot generate the necessary substitution character, exception condition EC-9803 exists. The standard action is to present the requested character at the established baseline coordinate. It is not an exception if a TBM control sequence with PRECSION set to X'01' is encountered by a receiver which implements only the actual placement method. If a receiver supports the substitution method in addition to the actual placement method, setting PRECSION to X'01' causes selection of the substitution method.

PRECSION is not a modal parameter. However, if another TBM control sequence that does not terminate the temporary baseline function is received, and PRECSION is not specified, then the current value of PRECSION is assumed. If the value of PRECSION is changed from X'00' to X'01' or from X'01' to X'00', the precision change must be honored according to the rules for PRECSION as indicated in the semantics.

- How TBM relates to underscore and overstrike:

  For receivers that support the actual placement method, the baseline position of characters resulting from the underscore function is the established baseline. The baseline position of characters resulting from the overstrike function is the current baseline, that is, the overstriking characters follow the temporary baseline. If an overstrike or underscore function continues after the temporary baseline function is terminated, no exception condition exists. The corresponding function continues relative to the current baseline, which is equal to the established baseline. For receivers that support only the substitution method, the overstrike and underscore functions occur relative to the established baseline.

- Miscellaneous TBM exception conditions:

  A receiver that supports the actual placement method for the TBM control sequence establishes the maximum temporary baseline displacement that it can support. If the Temporary Baseline Increment parameter exceeds the limits supported by the receiver, exception condition EC-9803 exists. The standard action is to use the established maximum limit that was exceeded as the standard action value for this parameter.

  More than one TBM control sequence without an intervening termination, such as a TBM of equal magnitude and opposite direction, or a TBM specifying Return to Established Baseline, is called *multi-offsetting*. Multi-offsetting support is required by PTOCA for receivers that support the actual placement method. It is not required to know the number of multi-offsets that have been accepted.

  Since receivers that support the substitution method do not physically create the temporary baseline, they cannot support multi-offset. If a multi-offset TBM is received by a substitution method receiver, exception condition EC-9803 exists. The standard action is to present according to the substitution method.

  If processing a TBM causes any portion of a character box on the current baseline to exceed the boundaries of the object space and presentation is attempted, exception condition EC-0103 exists. The standard action is to continue processing without presenting characters until the addressable point specified is valid for use as a presentation position, and then start presenting again. While processing without presenting characters, the current presentation position is maintained as though the affected characters were being presented.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-9803...The value of INCRMENT, PRECSION, or DIRCTION is not supported or is not in the range specified by PTOCA.
- EC-9803...The PRECSION parameter specifies the actual placement method but the receiver does not support it.
- EC-9803...A receiver using the substitution method cannot generate the required substitution character.
- EC-9803...For a receiver that uses the actual placement method, the INCRMENT parameter exceeds the physical limit.
- EC-9803...A multi-offset TBM control sequence is received by a receiver that uses the substitution method.

- EC-0103...The control sequence will cause part of a character's character box to be outside of the object space, and presentation is attempted.

Transparent Data (TRN)

The Transparent Data control sequence contains a sequence of code points that are presented without a scan for embedded control sequences.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	2–255	Control sequence length	M	N	N
3	CODE	TYPE	X'DA' – X'DB'	Control sequence function type	M	N	N
4–256	CHAR	TRNDATA	Not applicable	Transparent data	O	N	N

The contents of TRNDATA are unknown. TRNDATA does not accept the default indicator, but X'F....F' is valid.

Semantics

This control sequence specifies a string of code points, all of which are to be processed as graphic characters. No code point within the data field is recognized as a Control Sequence Prefix. The current inline position is incremented for each graphic character in the string.

For graphic characters following each other:

$I_{cnew} = I_c + ADJSTMNT + CI$

Intervening non-incrementing characters are ignored.

For graphic characters following RMI, AMI, BLN control sequences or following a space character or variable space character:

$I_{cnew} = I_c + CI$

Intervening non-incrementing characters are ignored.

For the variable space character:

$I_{cnew} = I_c + VSI$

For a non-incrementing character:

$I_{cnew} = I_c$

In all cases:

$B_{cnew} = B_c$

Pragmatics

No absolute move, relative move, baseline positioning, or other immediate or modal function is available within TRNDATA. If code points representing control sequences are processed within TRNDATA, they are presented as graphic characters, and the active coded font determines whether any character shapes appear on the presentation surface. If a Transparent Data control sequence causes any part of a character's character box to exceed the object space, exception condition EC-0103 exists. The standard action is not to present the character that exceeds the object space, and continue processing without presenting characters until the presentation position is returned to an addressable position within the object space that is a valid presentation position for the character being presented. Then presentation of characters may resume.

The data length must be an even number for double-byte fonts. If the Transparent Data control sequence length is an odd number when a double-byte font is active, exception condition EC-1A01 exists. The standard action is to process the Transparent Data control sequence up to the last byte, skip the odd byte, and continue processing.

If the character encoding is Unicode but the code points in the TRN are not valid Unicode data, exception condition EC-1A03 exists. The standard action is to skip the remainder of the code points in the TRN and continue processing.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-0103...The control sequence will cause part of a character's character box to be outside the object space, and presentation is attempted.
- EC-1A01...The control sequence length is an odd number, but a double-byte font is active.
- EC-1A03...Invalid Unicode data. This can be caused by one of the following:
  - A high-order surrogate code value was not immediately followed by a low-order surrogate code value. The high-order surrogate range is U+D800 through U+DBFF.
  - A low-order surrogate code value was not immediately preceded by a high-order surrogate code value. The low-order surrogate range is U+DC00 through U+DFFF.
  - An illegal UTF-8 byte sequence, as defined in the Unicode 3.2 Specification, was specified. For more information on illegal UTF-8 byte sequences, see Table 16 on page 120.

Underscore (USC)

The Underscore control sequence identifies text fields that are to be underscored.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	3	Control sequence length	M	N	N
3	CODE	TYPE	X'76' – X'77'	Control sequence function type	M	N	N
4	BITS	BYPSIDEN	See Semantics section	Bypass identifiers	M	Y	Y

BYPSIDEN is a binary field with no measurement units. The PTOCA default is X'01', which means no bypass is in effect.

Semantics

Underscore is accomplished with a pair of USC control sequences. Underscore is activated with a beginning USC with a non-zero value for BYPSIDEN bits 4-7. Underscore is deactivated with an ending USC with a zero value for BYPSIDEN bits 4-7. This control sequence immediately precedes the field of text to be underscored, which is called the *underscore field*. The control sequence specifies that text bounded by the two USC control sequences is to be underscored, and it specifies which controlled inline white space within that text is to be underscored.

The underscore field is delimited by a beginning USC control sequence and either an ending USC control sequence or the end of the Presentation Text object. The underscore field is a sequential string of text, that is, graphic characters or control sequences.

BYPSIDEN specifies which controlled inline white space within the underscore field is to be underscored. *Controlled inline white space* is that area of the presented line that contains no visible material due to movement of the presentation position in the I-direction caused by the following:

- Absolute Move Inline control sequence
- Relative Move Inline control sequence
- Space character or variable space character

Application Note: The following code points are normally used for the variable space character:

- X'40' in EBCDIC single-byte code pages
- X'20' in ASCII single-byte code pages
- X'4040' in EBCDIC double-byte code pages
- X'2020' in ASCII double-byte code pages

The following code points are used for the variable space character in TrueType/OpenType fonts that use a Unicode (UTF-16) encoding:

- X'0020'
- X'00A0'

Movement of the current inline position in the I-direction to or through a presentation position that already contains material to be underscored creates controlled inline white space for the entire move in the I-direction. Not all inline white space is controlled. White space resulting from non-printing characters (other than space characters or variable space characters) within the character set, from substitution of non-printing characters for unsupported characters, from intercharacter adjustment, or from the inline margin is not considered controlled inline white space.

Inline moves that cause the current addressable position to move in a direction opposite to the inline direction, that is, in the negative inline direction, do not cause underscore.

BYPSIDEN is bit encoded as follows.

BIT	MEANING
0-3	Reserved, that is, set to 0 by generators and ignored by receivers
4	Bypass Relative Move Inline
5	Bypass Absolute Move Inline
6	Bypass space characters and variable space characters
7	No Bypass in effect

Bits 0-3, Reserved

Reserved bits are set to 0 by generators and ignored by receivers.

Bit 4, Bypass Relative Move Inline

A value of B'0' in this bit indicates that the controlled white space generated as a result of a Relative Move Inline control sequence is to be underscored. A value of B'1' in this bit indicates that such controlled white space is not to be underscored. It should be bypassed.

Bit 5, Bypass Absolute Move Inline

A value of B'0' in this bit indicates that the controlled white space generated as a result of an Absolute Move Inline control sequence is to be underscored. A value of B'1' in this bit indicates that such controlled white space is not to be underscored. It should be bypassed.

Bit 6, Bypass space characters

A value of B'0' in this bit indicates that the controlled white space generated as a result of space characters or variable space characters is to be underscored. A value of B'1' in this bit indicates that such controlled white space is not to be underscored. It should be bypassed.

Bit 7, No Bypass in Effect

A value of B'0' in this bit activates the other bypass flags. A value of B'1' in this bit indicates that the other bypass flags are overridden, and that all text and white space bounded by the USC pair should be underscored. If the value of BYPSIDEN is the default indicator, a value is obtained from the hierarchy. Please see Table 10 on page 33.

Implementation Note: Most IPDS printers have implemented X'FF' as the default indicator, which results in BYPSIDEN = X'01' - no bypass in effect. However, it could be argued that the proper default indicator is X'0F', since BYPSIDEN bits 0-3 are reserved, should be set to zero by generators, and should be ignored by receivers. To avoid confusion, it is strongly recommended that a default indicator not be used for this parameter, and that the value X'01' is specified directly if the default is desired.

An *underscore area* is that portion of the underscore field for which text is actually underscored. An underscore area is delimited by the addressable position in the following cases.

- A beginning USC

- Either end of bypassed controlled inline white space
- Either end of a baseline move, which may be for the established baseline or for a temporary baseline
- The beginning of negative changes in the presentation position caused by inline moves or negative intercharacter adjustments
- Boundaries where violation causes truncation
- An ending USC
- The end of the Presentation Text object

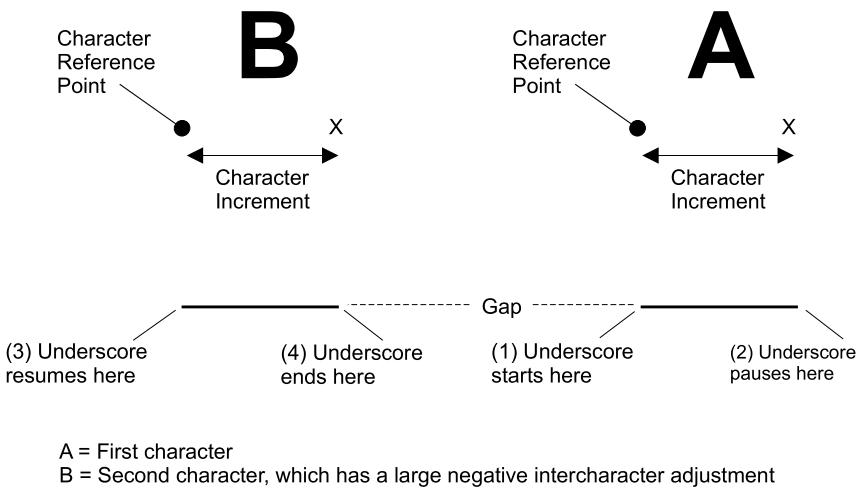
The dimension in the positive I-direction of the underscore field is defined by the minimum and maximum I-coordinates specified between the underscore area delimiters. White space resulting from the application of the inline margin is underscored only if this area is entered by means of an inline move.

Pragmatics

The underscore area must be underscored with a solid line. For each character to be underscored, the solid line must extend for the entire character box and for any intercharacter adjustment. The solid line may not extend past the character box for the final character in the underscore field. That is, the solid line may extend up to but not include the current presentation position, I_c.

Underscore does not occur in the negative I-direction. If there is negative intercharacter adjustment, the underscore is applied from the beginning of the character reference point, after it is placed, to a point which is one character increment from there in the positive I-direction. If the negative intercharacter adjustment is large enough to move the presentation position of the next character to a point which precedes the previous character's presentation position by more than the character increment of the next character, then an underscore beginning at the reference point of the next character will cause a gap in the underscore, as shown in Figure 14.

Figure 14. Example of Intercharacter Increment in Underscore



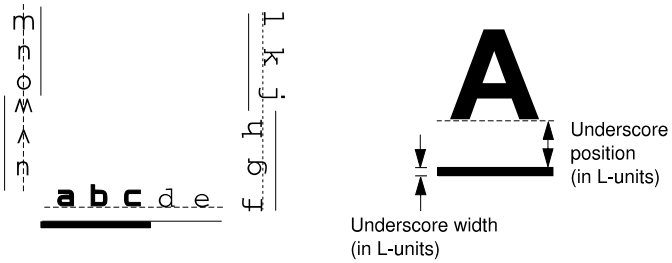
Multiple beginning and ending USC pairs may not be nested. However, no exception condition exists if a beginning USC control sequence is processed when another USC control sequence is already active. The subsequent beginning USC terminates the previous USC and starts another. If an ending USC is encountered when there has been no previous USC, no exception condition exists. Ignore the ending USC. If a Presentation Text object contains a beginning USC without a matching ending USC, no exception condition exists. Terminate the USC at the end of the Presentation Text object.

There is no provision in the USC control sequence to specify a coded font. It is assumed that the receiver can underscore. Underscore positioning is determined by the active coded font. If the active coded font is changed

within the underscore field, discontinuity of the underscore, such as mismatched lines, different line weights, or multiple lines, could result. However, this does not constitute a violation of the requirement for a solid line.

If an STO control sequence changes the text orientation within an underscore field, the position of the underscore is still determined by the active coded font. Please see Figure 15.

Figure 15. Relationship of Underscore to Changes in Font, Orientation, and Rotation



Notes:

1. The dashed lines represent character baselines. The solid lines represent underscores.
2. The underscore is always parallel to the character baseline. Positive underscore positions are placed under the character baseline in the direction of baseline progression; negative underscore positions are placed above the character baseline in the opposite direction. The underscore position for Latin languages is positive as shown in these examples. For Eastern writing systems, the underscore position can be negative.
3. The font changes between "c" and "d"; text orientation and character rotation do not change.
4. Text orientation changes between "e" and "f", from 0, 90 degrees to 270, 0 degrees. The I-axis coordinate and the B-axis coordinate change. The font and the character rotation do not change.
5. The character rotation changes between "h" and "j", from 0 to 180 degrees, requiring a font change. Text orientation changes to 270, 180 degrees.
6. The "m n o" characters are presented at a text orientation of 90, 0 degrees, with a character rotation of 270 degrees. The "u v w" characters are presented at a text orientation of 270, 180 degrees, with a character rotation of 270 degrees. Again, the font has been changed.
7. In all cases, position and width of the underscore is determined by the active coded font.

Typically an STO control sequence is accompanied by an SCFL control sequence that specifies the appropriate coded font to use in the new orientation. The coded font specified by the SCFL determines where the underscore is positioned in the new orientation. The requirement for a solid line is not violated as long as the underscore extends for the character increment and any intercharacter adjustment. For some combinations of text orientation and character rotation, valid underscore might be discontinuous.

An underscore area is delimited by a beginning USC, bypassed controlled inline white space, and either an ending USC or the end of the Presentation Text object. Additionally, the dimension of this area in the I-direction is defined by the minimum and maximum I-coordinates specified between the underscore delimiters.

There are no syntactic restrictions on the occurrence of Begin Line, Absolute Move Baseline, Relative Move Baseline, and Temporary Baseline Move control sequences within an underscore field.

Characters occurring at a temporary baseline coordinate are underscored at the established baseline coordinate position.

Color is not a parameter of this control sequence.

Exception Conditions

This control sequence can cause the following exception condition:

- None

Unicode Complex Text (UCT)

Architecture Note: The recommended method for rendering Unicode complex text is to use GLC chains. With that method, the UCT may optionally be specified within a GLC chain as a carrier for metadata that specifies the original Unicode code points and control information describing how those code points may be rendered. In that usage, the UCT does not cause any characters to be rendered. Applications can use the UCT in GLC chains to “virtually“ render the code points in order to better correlate substrings of glyph runs with their corresponding code points. The use of the UCT as a stand-alone control sequence to render complex text has not been implemented widely and is not supported in IPDS environments.

The Unicode Complex Text (UCT) control sequence marks the start of a sequence of Unicode code points. This sequence starts with the first byte following the end of the UCT control sequence and ends with the last byte identified by the complex text length parameter in the control sequence. All bytes in this sequence are processed as code points without a scan for embedded control sequences. There is no odd function type defined for the UCT, therefore the UCT can be part of a chain of control sequences but will always terminate the chain.

If the active font is a data-object font and the data is encoded in a Unicode-based character encoding such as UTF-8 or UTF-16, the code points in the text that follows UCT can be processed as Unicode complex text. Under the following conditions the code points cannot be processed as Unicode complex text and instead are processed by the presentation device as if they were within a TRN control sequence, in which case all UCT control information is ignored:

- the active font is not a data-object font
- the data is not encoded in a Unicode-based character encoding
- the writing mode is vertical, as determined by a font character rotation of 90° or 270°

The UCT can be used in two ways.

1. When the UCT is specified within a GLC chain, that is when it is chained from a GAR or a GOR, it is used to identify the code points rendered by the glyphs in the GLC chain. In that case, the code points following the UCT are not rendered, since the corresponding glyph IDs are rendered by the GLC chain. How much of the UCT controls are processed by the receiver to correlate code points to glyphs is device-dependent. If bidi processing and glyph processing are enabled within the UCT, a receiver may choose to process all of the UCT controls to actually lay out the code points in order to more closely associate code points with rendered glyphs. Or a receiver may choose to ignore all of the UCT controls, in which case the association of code points with glyph IDs can be more coarse. Processing of the UCT has no effect on the current inline position, I_c, or on any other PTOCA environment controls.
2. When the UCT is specified outside a GLC chain (a “stand-alone“ UCT), it is used to process and render the code points that follow.

   Architecture Note: The processing and rendering of a UCT outside a GLC chain is not part of the PTOCA PT4 function set. Not all presentation devices support this function; consult the appropriate product documentation. IPDS printers generate an exception when a UCT is specified outside a GLC chain. Some IPDS printers, under control of an environment-specified text fidelity control, will render the UCT code points in a one-to-one manner.

Rendering complex text involves two types of special processing:

- *bidirectional (bidi) layout processing.* When the writing mode is horizontal, as determined by a character rotation in the active font of 0° or 180°, characters are presented on the current PTOCA baseline in a direction determined by the Unicode bidi algorithm. For a description of the Unicode bidi algorithm, see `http://unicode.org/reports/tr9`. The direction can be left-to-right (L→R), right-to-left (R→L), or a combination. Characters are presented at the character rotation specified in the active font, and the appropriate horizontal metrics are used to position the glyphs. See Table 15 on page 119 for a description of UCT character positioning.

Application Note: It is strongly recommended that the logical (storage) order of Unicode text strings be preserved when such strings are placed into text objects. This will allow the text, when carried in an interchange format such as a MO:DCA document, to be interchanged among applications that support text processing functions such as searching, sorting, and indexing. If the logical order is modified, which might be tempting as an aid to formatting, the capability to apply such text processing functions is lost.

- *glyph processing.* Characters may require language-specific shaping such as arabic character shaping, composition/decomposition, and position adjustments. The rendering is no longer one code point to one character to one glyph, and requires the assistance of a Unicode layout engine.

  Implementation Note: In AFP environments, the ICU ParagraphLayout API is used to apply Unicode bidi layout processing and glyph processing to complex text. This API is developed by the International Components for Unicode (ICU) project, which is jointly managed by a group of companies and individuals throughout the world; see `http://site.icu-project.org`.

Complex text processing for UCT text is enabled or disabled by controls in the UCT. If both bidi layout processing and glyph processing are disabled, the code points in the text that follows UCT are not processed as complex text and instead are processed as if they were contained within a TRN control sequence.

If either bidi layout processing or glyph processing is to be applied, the sequence of Unicode code points must first be normalized. This involves, for example, replacing composite character sequences with their equivalent composed character. The Unicode normalization format used in PTOCA objects is Normalization Form C (NFC). If the normalization step was not applied by the formatter that generated the complex text, it is applied by the presentation device before bidi layout processing or glyph processing is applied.

Architecture Note: The Unicode character encoding is defined in the Unicode Standard, which is available from the Unicode Consortium at `http://unicode.org/standard/standard.html`.

Syntax

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	PREFIX	X'2B'	Control Sequence Prefix	M	N	N
1	CODE	CLASS	X'D3'	Control sequence class	M	N	N
2	UBIN	LENGTH	X'10'	Control sequence length	M	N	N
3	CODE	TYPE	X'6A'	Control sequence function type	M	N	N
4	CODE	UCTVERS	X'01'	UCT version level **X'01'** Base level	M	N	N
5				Reserved; should be zero	M	N	N
6–7	UBIN	CTLNGTH	0 – 32,767	Length of complex text data that follows this control sequence	M	N	N
8	BITS	CTFLGS	See Semantics section	Complex text processing control flags	M	N	N
9				Reserved; should be zero	M	N	N

10	CODE	BIDICT	X'02', X'04', X'05', X'12', X'20', X'22', X'23'	Bidi layout processing control: **X'02'** Enable; default p.d. is L→R **X'04'** Enable; set p.d. L→R **X'05'** Enable; set p.d. R→L **X'12'** Enable; p.d. set from previous UCT; default L→R **X'20'** Disable **X'22'** Disable; t.d. L→R **X'23'** Disable; t.d. R→L	M	N	N
11	CODE	GLYPHCT	X'01', X'20'	Glyph processing control: **X'01'** Enable **X'20'** Disable	M	N	N
12 – 15				Reserved; should be zero	M	N	N
16 – 17	SBIN	ALTIPOS	X'8000' – X'7FFF'	Alternate current inline position	M	N	N

Table Note: The terms 'p.d.' and 't.d.' stand for paragraph direction and text direction, respectively.

ALTIPOS is a signed binary number in measurement units.The range for ALTIPOS assumes a measurement unit of 1/1440 inch. If it is necessary to convert to a different measurement unit, please see the conversion routine described in “Interpreting Ranges” on page 45.

Semantics

This control sequence marks the start of a string of code points, all of which are to be processed as graphic characters. No code point within the marked string of text is recognized as a Control Sequence Prefix. This string of code points is also called “UCT text”.

If the code points in the text that follows the UCT control sequence cannot be processed as Unicode complex text, or if glyph processing and bidi processing are disabled, the code points are processed as if they were contained in a TRN control sequence. In that case, the current inline position is incremented for each graphic character in the string as follows:

- For graphic characters following each other:
  $I_{cnew} = I_c + ADJSTMNT + CI$
  and intervening non-incrementing characters are ignored.
- For graphic characters following RMI, AMI, or BLN control sequences or following a space character or variable space character:
  $I_{cnew} = I_c + CI$
  and intervening non-incrementing characters are ignored.
- For the variable space character:
  $I_{cnew} = I_c + VSI$
- For a non-incrementing character:
  $I_{cnew} = I_c$
- In all cases:
  $B_{cnew} = B_c$

If the code points in the text that follows the UCT control sequence can be processed as Unicode complex text, and if glyph processing or bidi processing is enabled, glyph positions and advances are determined with the aid of a Unicode layout engine. In that case the advancement of the current text position is no longer a direct function of the individual character increments. Additionally, the optional placement of text at the alternate inline position may dictate the new text position.

If CTFLGS bit 3 = B'0' - advance I_c, the current text position I_{cnew} at the completion of a UCT is then generated as follows:

- If I_c was used to position the UCT text:
  $I_{cnew} = I_c + sum\{G_I\}$
- If I_a was used to position the UCT text:
  $I_{cnew} = I_a$

If CTFLGS bit 3 = B'1' - maintain I_c, the current text position I_{cnew} at the completion of a UCT is given by:
$I_{cnew} = I_c$

Where:

- I_c = the current text position at the start of UCT processing
- I_a = the alternate text position at the start of UCT processing
- G_I = the increment for a grapheme in the UCT
- sum = summation over all the graphemes that were presented for the UCT

UCTVERS specifies the functional level of the complex text support in the UCT.

Value	Definition
X'01'	Base level of complex text support
All others	Reserved

CTLNGTH specifies the total number of bytes in the sequence of code points that follows UCT; that is, it specifies the length of the text that follows UCT. The bytes identified by this parameter are processed as code points and are presented without a scan for embedded control sequences.

CTFLGS is a bit-encoded parameter that specifies controls for processing the Unicode complex text code points that follow, and is defined as follows:

BIT	MEANING
0	Normalization
1	Alternate inline position (I_a) valid
2	Alternate inline position (I_a) coordinates
3	Maintain current inline position (I_c)
4	Reset paragraph direction
5-7	Reserved, that is, set to B'0' by generators and ignored by receivers

Bit 0, Normalization

A value of B'0' in this bit indicates that the Unicode complex text code points that follow have not been normalized. The presentation device will apply a normalization step as long as bidi layout processing or glyph processing (or both) is to be applied. If neither bidi layout processing or glyph processing is to be applied, the normalization step is not applied. The normalization format generated is Normalization Form C (NFC). A value of B'1' in this bit indicates that the Unicode complex text code points that follow have been normalized by the generator of the text object. No additional normalization is applied by the presentation device. The normalization format assumed is Normalization Form C (NFC).

Reserved bits are set to B'0' by generators and ignored by receivers.

Architecture Note: The definition of Normalization Format C (NFC) is available at
`http://unicode.org/reports/tr15`.

Bit 1, Alternate inline position (I_a) valid

A value of B'0' in this bit indicates that the alternate inline position parameter ALTIPOS is not valid and cannot be used to position UCT text. A value of B'1' in this bit indicates that the alternate inline position parameter ALTIPOS is valid and can be used to position UCT text.

Bit 2, Alternate inline position (I_a) coordinates

A value of B'0' in this bit indicates that the alternate inline position parameter I_a is specified using absolute i-coordinate values. In this case the range of I_a is restricted to positive values. A value of B'1' in this indicates that the alternate inline position parameter I_a is specified using relative i-coordinate values that are measured with respect to the current inline position I_c. In this case the range of I_a allows positive and negative values.

Bit 3, Maintain current inline position (I_c)

A value of B'0' in this bit indicates that the current inline position I_c is advanced in accordance with the equations for I_{cnew} at the completion of the UCT. A value of B'1' in this bit indicates that the current inline position I_c should not be advanced when the UCT is processed. In this case the value of I_c at the completion of UCT processing is equal to the value of I_c at the start of UCT processing.

Bit 4, Reset paragraph direction

A value of B'0' in this bit has no effect on the paragraph direction and is treated as a No-op. A value of B'1' in this bit coupled with CTLNGTH= X'0000' causes the paragraph direction to be reset to an undefined state. In that case all other UCT controls are ignored. If this bit is set to B'1' and CTLNGTH≠X'0000', the bit is treated as if it were set to B'0'.

Implementation Note: When AFP print servers process a Begin Presentation Text (BPT) structured field in a MO:DCA data stream, they issue a set of PTOCA control sequences to establish default initial text conditions for processing the text object. When the server supports the processing of stand-alone UCTs (UCTs that are not part of a GLC chain) in attached presentation devices, this set must include a UCT control sequence with CTLNGTH = X'0000' and CTFLGS bit 4 = B'1' to reset the paragraph direction to an undefined state at the beginning of each text object. When the server only supports the processing of UCTs within GLC chains in attached presentation devices, this additional control is not necessary.

Bits 5-7, Reserved

Reserved bits are set to B'0' by generators and ignored by receivers.

BIDICT is a code that specifies controls for processing the Unicode complex text code points that follow with the Unicode bidi algorithm. In all cases, characters are presented at the character rotation specified in the active font. If the active font is not a data-object font, or if it is a data-object font that does not specify a Unicode-based character encoding, this parameter is ignored. For most BIDICT values, this parameter establishes the Unicode paragraph direction, which is used as an input to the Unicode bidi algorithm. Table 15 on page 119 shows how the paragraph direction for Unicode complex text affects the positioning of the UCT text. See “Bidi Layout Processing for UCT Text” on page 120 for a description of bidi layout processing for UCT text.

Value	Description
X'02'	Enable Unicode bidi layout processing for the complex text code points that follow. The paragraph direction is determined by the complex text string based on the first strong directional character that is encountered. If no paragraph direction can be determined, use a left-to-right default paragraph direction.
X'04'	Enable Unicode bidi layout processing for the complex text code points that follow. The paragraph direction is set to left-to-right.

X'05'	Enable Unicode bidi layout processing for the complex text code points that follow. The paragraph direction is set to right-to-left.
X'12'	Enable Unicode bidi layout processing for the complex text code points that follow. The paragraph direction is determined by the previously processed complex text string in this text object. If this is the first complex text string that is encountered in this text object, or if the paragraph direction is undefined, the paragraph direction is based on the first strong directional character that is encountered. If no paragraph direction can be determined, use a left-to-right default paragraph direction.
X'20'	Disable Unicode bidi layout processing for the complex text code points that follow. The current PTOCA inline direction determines the text direction on the current PTOCA baseline. The code points in the UCT are processed with respect to text direction as if they were within a TRN.
X'22'	Disable Unicode bidi layout processing for the complex text code points that follow. The UCT contains a single directional run and is processed with a fixed text direction that is left-to-right. There are no text direction changes for this UCT. The alternate inline position parameter I_a, if specified, is not used.
X'23'	Disable Unicode bidi layout processing for the complex text code points that follow. The UCT contains a single directional run and is processed with a fixed text direction that is right-to-left. There are no text direction changes for this UCT. The alternate inline position parameter I_a, if specified, is not used.
All others	Reserved

GLYPHCT is a code that specifies controls for applying glyph processing to the Unicode complex text code points that follow. In all cases, characters are presented at the character rotation specified in the active font. If the active font is not a data-object font, or if it is a data-object font that does not specify a Unicode-based character encoding, this parameter is ignored.

Value	Description
X'01'	Enable glyph processing for the complex text code points that follow. Language-specific processing is applied to runs of characters and may result in shaping, composition/ decomposition, and positional adjustments.
X'20'	Disable glyph processing for the complex text code points that follow. The code points in the UCT are processed with respect to glyph processing as if they were within a TRN. Characters are rendered in a one-to-one fashion using the selected glyph in the active font.
All others	Reserved

If Unicode bidi layout processing is disabled (BIDICT=X'20') and if Unicode glyph processing is disabled (GLYPHCT=X'20') the UCT code points are not treated as complex text and are processed as if they were within a TRN control sequence.

ALTIPOS is a parameter that specifies an alternate inline position (I_a) on the current baseline that may be used in place of the current inline text position (I_c) when processing Unicode complex text and the paragraph direction is opposite the writing mode direction. This parameter is ignored if Unicode bidi layout processing is disabled for the code points that follow. This parameter is also ignored if the active font is not a data-object font, or if it is a data-object font that specifies a non-Unicode-based character encoding. The use of this parameter depends on the current character rotation–as specified in the active font–and the paragraph direction for the UCT. Table 15 on page 119 shows how the text in a UCT is positioned either with respect to the current inline position I_c, or with respect to the alternate inline position I_a.

Table 15. UCT Text Positioning

PTOCA (i,b) Orientation	Character Rotation	UCT Paragraph Direction	UCT Character Position w.r. to I_a or I_c	UCT Character Position w.r. to I_c only
All 8 (i,b) orientations	Character rotation 0°: Horizontal L→R writing mode	L→R	Start at I_c; place UCT text so it extends in (+) i-direction	Start at I_c; place UCT text so it extends in (+) i-direction
		R→L	Start at I_a; place UCT text so it extends in (-) i-direction	Start at I_c; place UCT text so it extends in (+) i-direction
All 8 (i,b) orientations	Character rotation 180°: Horizontal R→L writing mode	R→L	Start at I_c; place UCT text so it extends in (+) i-direction	Start at I_c; place UCT text so it extends in (+) i-direction
		L→R	Start at I_a; place UCT text so it extends in (-) i-direction	Start at I_c; place UCT text so it extends in (+) i-direction
All 8 (i,b) orientations	Character rotation 270°: Vertical T→B writing mode	N/A	Start at I_c; place UCT text so it extends in (+) i-direction	Start at I_c; place UCT text so it extends in (+) i-direction
All 8 (i,b) orientations	Character rotation 90°: Vertical B→T writing mode	N/A	Start at I_c; place UCT text so it extends in (+) i-direction	Start at I_c; place UCT text so it extends in (+) i-direction

Table Notes:

1. The terms 'left', 'right', 'top', and 'bottom–as in 'L→R', 'R→L', 'T→B', and 'B→T'–only have meaning when the media has been oriented so that the characters appear right-side-up.
2. The (+) i-direction is the direction of increasing i-values; the (-) i-direction is the direction of decreasing i-values.

Pragmatics

The CTLNGTH parameter must specify an even number when the character encoding is double-byte, as in UTF-16. If CTLNGTH is an odd number when the character encoding is double byte, exception condition EC-1A01 exists. The standard action is to process the code point sequence up to the last byte, skip the odd byte, exit UCT processing mode, and continue processing. If the CTLNGTH value is out of the architected range, exception condition EC-9B01 exists. The standard action is to process the UCT code points up to the maximum valid value of CTLNGTH, exit UCT processing mode, and continue processing.

If the UCTVERS, BIDICT, or GLYPHCT parameters specify an invalid value, exception condition EC-9B01 exists. The standard action is to process the UCT code points as if they were within a TRN.

The code points that follow the UCT may not specify valid Unicode data. In that case exception condition EC-1A03 exists, and the standard action is to skip the remainder of the UCT code points, exit UCT processing mode, and continue processing.

The UCT control sequence always terminates chaining. Therefore, in the exception cases noted, when the presentation device exits UCT processing mode, it continues processing by interpreting the next bytes as either code points or the start of an unchained control sequence.

Exception Conditions

This control sequence can cause the following exception conditions:

- EC-1A01...The CTLNGTH parameter is an odd number, but the character encoding is double byte.

- EC-9B01...The CTLNGTH, UCTVERS, BIDICT, or GLYPHCT parameter values are invalid.
- EC-1A03...Invalid Unicode data. This can be caused by one of the following:
  - A high-order surrogate code value was not immediately followed by a low-order surrogate code value. The high-order surrogate range is U+D800 through U+DBFF.
  - A low-order surrogate code value was not immediately preceded by a high-order surrogate code value. The low-order surrogate range is U+DC00 through U+DFFF.
  - An illegal UTF-8 byte sequence, as defined in the Unicode 3.2 Specification, was specified. For more information on illegal UTF-8 byte sequences, see Table 16.

Application Note. For a formal definition of UTF-8, consult the Unicode 3.2 Specification. The illegal UTF-8 byte sequences can be summarized as follows:

- The value in the 1st byte of the UTF-8 byte sequence was not in the legal UTF-8 range (X'00' - X'7F' and X'C2' - X'F4').
- The value in the 2nd byte of the UTF-8 byte sequence was not in the legal UTF-8 range allowed by the value in the 1st byte. The valid ranges for the 2nd byte are shown in Table 16.

Table 16. Valid Values for UTF-8 First and Second Bytes

First Byte	Second Byte
X'C2'–X'DF'	X'80"–X'BF'
X'E0'	X'A0"–X'BF'
X'E1"–X'EC'	X'80"–X'BF'
X'ED'	X'80"–X'9F'
X'EE"–X'EF'	X'80"–X'BF'
X'F0'	X'90"–X'BF'
X'F1"–X'F3'	X'80"–X'BF'
X'F4'	X'80"–X'8F'

- The value in the 3rd or 4th byte of the UTF-8 byte sequence was not in the legal UTF-8 range for that byte (X'80' - X'BF').

Bidi Layout Processing for UCT Text

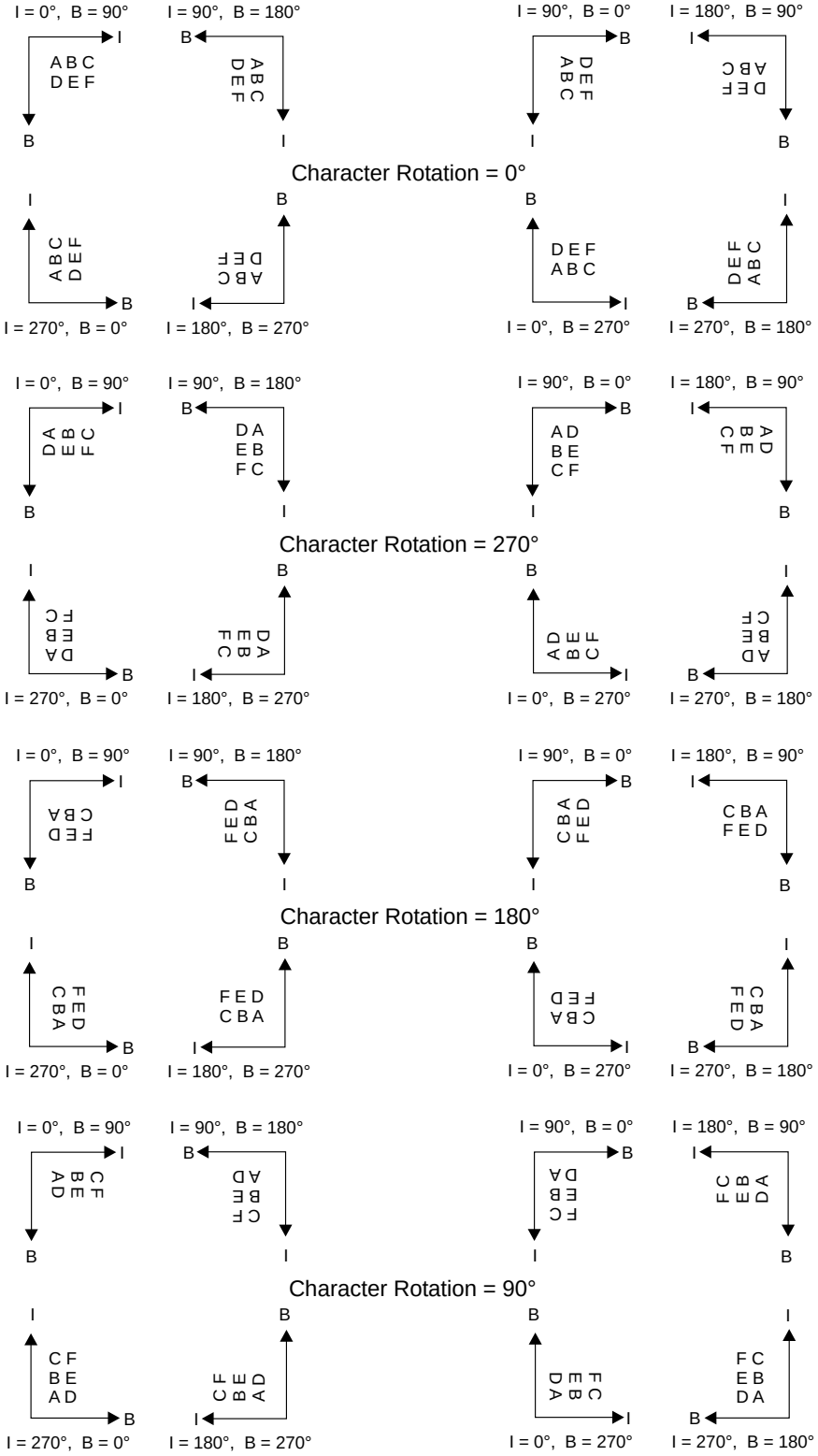
A number of parameters–both outside the UCT and inside the UCT–determine how Unicode bidi layout processing is applied to text in a UCT:

- **Text orientation.** The orientation of the (i,b) coordinate system which specifies the baseline on which glyphs are positioned and the reference inline direction for progressing text directional runs.
- **Character rotation.** Alignment of a character with respect to the baseline. Differentiates between horizontal and vertical writing modes.
- **Writing mode.** Can be horizontal (L→R or R→L) or vertical (top->bottom or bottom->top). Determines–together with the paragraph direction–how UCT text is positioned with respect to the current inline position (I_c) or with respect to an alternate inline position (I_a).
- **Bidirectional character property.** A property value assigned by the Unicode standard to each character, including unassigned characters. Values include strong L→R, strong R→L, weak L→R, weak R→L, and neutral characters.
- **Text direction.** Specifies the visual ordering of characters in a given directional run. The inherent directional properties of Unicode characters dictate the text direction assigned many characters, while the Unicode bidi layout algorithm will assign a direction to characters such as punctuation marks which have a neutral directional property.

- **Paragraph direction.** Specifies the dominant text direction for a UCT. Used as an input to the Unicode bidi layout algorithm where it influences the ordering of directional runs and of directionally-neutral characters in a UCT.

The processing of the Unicode complex text in a UCT occurs within the context of the PTOCA and AFP presentation models, which define 32 ways to print text based on the 8 (i,b) text orientations and 4 character rotations. Figure 16 on page 122 shows the 32 ways to print text in AFP environments.

Figure 16. 32 Ways to Print Text in AFP Environments



UCT Bidi Processing Examples

The following examples illustrate how the writing mode, text direction, and paragraph direction influence the rendering of UCT Unicode bidi text. For all the examples, assume a character string that is stored in logical (storage) order as:

R0 R1 R2 L3 L4 L5 R6 R7 R8 N9

where the R_i are strong R→L characters, such as Arabic and Hebrew characters, the L_i are strong L→R characters, such as English (Latin) characters, and the N_i have a neutral directional property, such as a punctuation mark.

The *text direction* affects character ordering when processing a given directional run from logical (storage) order into visual (rendering) order.

- The text direction of the L_i characters is L→R. In that case the characters are rendered as:
  L3 L4 L5
- The text direction of the R_i characters is R→L. In that case the characters are rendered as:
  R8 R7 R6 and R2 R1 R0
- The text direction can also be set by the BIDICT parameter on the UCT. With BIDICT values X'22'and X'23', the text direction is fixed as either L→R or R→L, respectively, regardless of the directional property of the characters. Care must be taken when using these BIDICT values or undesirable results may occur, as for example, in the following cases:
  - If BIDICT=X'22' (set L→R text direction), the R_i strings are rendered as:
    R0 R1 R2 and R6 R7 R8
  - Similarly, if BIDICT=X'23' (set R→L text direction), the L_i string is rendered as:
    L5 L4 L3

The *paragraph direction* affects the ordering of directional runs in a UCT and of directionally-neutral characters in a UCT. Assume that the complete string in the previous example is included in a single UCT.

- If the paragraph direction is R→L, the complete string is rendered as:
  N9 R8 R7 R6 L3 L4 L5 R2 R1 R0
- If the paragraph direction is L→R, the complete string is rendered as:
  R2 R1 R0 L3 L4 L5 R8 R7 R6 N9

The *writing mode* determines how UCT text is positioned on the current baseline. Assume the current inline position is I_c.

- Writing mode L→R, paragraph direction L→R:
  (I_c) R2 R1 R0 L3 L4 L5 R8 R7 R6 N9
- Writing mode R→L, paragraph direction L→R:
  R2 R1 R0 L3 L4 L5 R8 R7 R6 N9 (I_c)
- Writing mode L→R, paragraph direction R→L:
  (I_c) N9 R8 R7 R6 L3 L4 L5 R2 R1 R0
- Writing mode R→L, paragraph direction R→L:
  N9 R8 R7 R6 L3 L4 L5 R2 R1 R0 (I_c)

The alternate inline position (I_a), if specified, can affect how the UCT text is positioned on the baseline as follows.

- Writing mode L→R, paragraph direction L→R:
  (I_c) R2 R1 R0 L3 L4 L5 R8 R7 R6 N9 (I_a)
- Writing mode L→R, paragraph direction R→L:
  (I_c) N9 R8 R7 R6 L3 L4 L5 R2 R1 R0 (I_a)
- Writing mode R→L, paragraph direction R→L:
  N9 R8 R7 R6 L3 L4 L5 R2 R1 R0 (I_c) (I_a)
- Writing mode R→L, paragraph direction L→R:
  (I_c (I_a) R2 R1 R0 L3 L4 L5 R8 R7 R6 N9

Unicode bidi layout processing is not supported in vertical T→B or B→T writing modes. When characters that have a horizontal directional attribute–such as Latin characters–are encountered while in a vertical writing mode, they are presented in the same direction and at the same character rotation as the characters with a vertical directional attribute. There is also no additional Unicode bidi layout processing for directional changes due to embedded L→R or R→L sequences.

Positioning Considerations for UCT Text

The above examples showing the effect of paragraph direction assume that the complete text string fits on one line, can be included in a single UCT, and can be treated as a single paragraph. This is important because proper rendering of Unicode bidi text requires that the Unicode bidi layout algorithm be applied to a complete paragraph. This is not always practical. It may be necessary to break up a paragraph into multiple UCTs to insert formatting commands for functions such as line breaks, page breaks, font changes, and color changes. In those cases the single paragraph is segmented into multiple UCTs so that the appropriate PTOCA control sequences can be inserted between UCTs.

Unfortunately, when a paragraph is segmented into multiple UCTs, the scope of the original paragraph is lost. Applying a paragraph direction to each segmented UCT will, in general, not result in the correct rendering of the original paragraph. To avoid this, the PTOCA generator needs to invoke the Unicode bidi layout algorithm to break the paragraph into directional runs and then package these directional runs into individual UCTs that specify the desired text direction (BIDICT values X'22'and X'23'). For example the complete string could be broken into the following three UCTs when a R→L paragraph direction is used. These UCTs must appear in the data stream in the order shown to preserve the logical order of the Unicode text:

UCT1{TD=R→L} R0 R1 R2 UCT2{TD=L→R} L3 L4 L5 UCT3{TD=R→L} R6 R7 R8 N9

Since the substrings now no longer belong to the same UCT, care must be taken by the PTOCA generator to ensure that the UCTs containing the substrings are positioned properly. Setting the writing mode to match the paragraph direction for the complete string will minimize the need for explicit UCT positioning. In our example, if the writing mode is R→L, the three UCTs would be rendered correctly based on the default text position advancements:

(I_{c3}) N9 R8 R7 R6 (I_{c2}) L3 L4 L5 (I_{c1}) R2 R1 R0 (I_c)

where I_c is the text position before the first UCT (UCT1) is processed, and I_{c1} is the text position after UCT1 is processed. However if the writing mode is L→R, the three UCTs would not be rendered correctly using the default text position advancements:

(I_c) R2 R1 R0 (I_{c1}) L3 L4 L5 (I_{c2}) N9 R8 R7 R6 (I_{c3})

In many circumstances explicit positioning of segmented UCTs is unavoidable, even if the writing mode matches the paragraph direction of the complete string. Consider the need to print the character L5 in a bold font:

UCT1{TD=R→L} R0 R1 R2
UCT2a{TD=L→R} L3 L4
    << font change >>
UCT2b{TD=L→R} L5
    << font change >>
UCT3{TD=R→L} R6 R7 R8 N9

Even with a R→L writing mode, the default positioning would result in the following incorrect rendering:

(I_{c3}) N9 R8 R7 R6 (I_{c2b}) L5 (I_{c2a}) L3 L4 (I_{c1}) R2 R1 R0 (I_c)

In this case, UCT2a, UCT2b, and UCT3 must be explicitly positioned to maintain the correct layout, as follows. Assume that M_i is the movement of the text position caused by processing UCT1; that is, it is the sum of the grapheme increments for UCT1:

UCT1{TD=R→L} R0 R1 R2
    << Relative Move Inline in (+) i-direction by M2b >>

UCT2a{TD=L→R} L3 L4
<< Relative Move Inline in (-) i-direction by {M2a + M2b}>>
<< font change >>
UCT2b{TD=L→R} L5
<< font change >>
<< Relative Move Inline in (+) i-direction by M2a >>
UCT3{TD=R→L} R6 R7 R8 N9

This results in the correct rendering of the string:

(I_{c1}) R2 R1 R0 (I_c)
(I_{c2a}) L3 L4 (I_{c1}) R2 R1 R0 (I_c)
(I_{c2a}) L3 L4 (I_{c2b}) L5 (I_{c1}) R2 R1 R0 (I_c)
(I_{c3}) N9 R8 R7 R6 (I_{c2a}) L3 L4 (I_{c2b}) L5 (I_{c1}) R2 R1 R0 (I_c)

Implementation Note: The use of the ICU ParagraphLayout API is recommended for PTOCA generators that need to segment bidi paragraphs. This API can provide the information needed to specify a text direction and position for each segmented UCT. See `http://site.icu-project.or`.

Effect of Other PTOCA Control Sequences on UCT Text

The unique characteristics of complex text require that some PTOCA control sequences have a different effect on UCT text processing than they do on non-UCT text processing. These differences are defined as follows:

- *Overstrike (OVS).* Table 17 on page 126 defines which Unicode space characters are treated as PTOCA space characters or variable space characters when an overstrike is generated. This applies to both UCT text and Unicode-encoded non-UCT text.
- *Set Intercharacter Adjustment (SIA).* If glyph processing or bidi layout processing is enabled, intercharacter adjustment is not applied to UCT text, and the current inline position is incremented for each graphic character as follows:
  $I_{cnew} = I_c + CI$
  In all other cases the SIA control sequence is processed the same for UCT text as for non-UCT text.
- *Set Text Orientation (STO).* If bidi layout processing is enabled and the writing mode is horizontal, characters in complex text may be progressed in the negative i-direction as defined in Table 15 on page 119. In all other cases the STO control sequence is processed the same for UCT text as for non-UCT text.
- *Set Variable Space Character Increment (SVI).* Table 17 on page 126 defines which Unicode space characters are mapped to the PTOCA variable space character and are assigned the increment specified by the SVI control sequence. This applies to both UCT text and Unicode-encoded non-UCT text.
- *Temporary Baseline Move (TBM).* If glyph processing or bidi layout processing is enabled, the precision parameter in this control sequence is ignored for complex text, and such text is processed as if the precision parameter were set to B'0' - actual placement method. This means that the presentation device must actually move the baseline and position the complex text characters on the temporary baseline. In all other cases the TBM control sequence is processed the same for UCT text as for non-UCT text.
- *Underscore (USC).* Table 17 on page 126 defines which Unicode space characters are treated as PTOCA space characters or variable space characters when an underscore is generated. This applies to both UCT text and Unicode-encoded non-UCT text.

All other PTOCA control sequences are processed the same for UCT text as for non-UCT text.

Table 17. Unicode Space Characters Mapped to PTOCA Space and Variable Space Characters

Unicode Space Character Name	Unicode Code Point	PTOCA Space Character	PTOCA Variable Space Character
SPACE	U+0020	yes	yes
NO-BREAK SPACE	U+00A0	yes	yes

Presentation Text Data Descriptor

The Presentation Text Data Descriptor provides initial parameters for the Presentation Text Object.

Syntax: Please see the appendixes for information about the syntax of the Presentation Text Data Descriptor and its parameters within the MO:DCA and IPDS environments.

Semantics: A Presentation Text Data Descriptor describes a Presentation Text object. It specifies the measurement units for the object and the size of the object. The descriptor also may describe the object through a list of initial text conditions, which contain initial values such as the size of the inline margin.

Pragmatics: The Presentation Text Data Descriptor is optional, because its parameters may be specified by the controlling environment or by default.

The descriptor parameters fall into the following categories:

1. Measurement units parameters:
   - Unit base
   - X_p-units per unit base
   - Y_p-units per unit base
2. Size parameters
   - X_p-extent of the presentation space
   - Y_p-extent of the presentation space
3. Initial text condition parameters

The following section describes these parameters.

Measurement Unit Parameters

Semantics: These parameters specify the measurement units for the Presentation Text object space::

Unit base	A number from 0 through 1
X_p-units per unit base	A number from 1 through 32,767
Y_p-units per unit base	A number from 1 through 32,767

The unit base parameter specifies the measurement base. It has the following values:

Values	Description
0	Ten inches
1	Ten centimeters
2-254	Reserved

If the value of the unit base parameter is not supported or is not within the range specified by PTOCA, exception condition EC-0505 exists. The standard action is to ignore this parameter and continue presentation with the value determined according to the hierarchy. The subset may limit the range permitted. For detailed information about subsets, please see Chapter 6, “Compliance with PTOCA”, on page 145, Appendix A, “MO:DCA Environment”, on page 155, and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data stream documentation.

Units of measure are defined as the measurement base divided by the units per unit base. That is, if the measurement base is 10 inches and the units per unit base are 5,000, the units of measure are 10 inches / 5000 or one five-hundredth of an inch. Here are further examples.

Table 18. Examples of Measurement Units

Units/inch	Parameter Values	Comments
800 X 800 units/in.	Unit base = 0 X_p-units per unit base = 8,000 Y_p-units per unit base = 8,000	8,000 divisions in 10 in. on both axes
80 X 77 units/cm.	Unit base = 1 X_p-units per unit base = 800 Y_p-units per unit base = 770	800 divisions in 10 cm. on X_p 770 divisions in 10 cm. on Y_p
203.3 X 195.5 units/in.	Unit base = 0 X_p-units per unit base = 2,033 Y_p-units per unit base = 1,955	2,033 divisions in 10 in. on X_p 1,955 divisions in 10 in. on Y_p
240 x 240 units/in.	Unit base = 0 X_p-units per unit base = 2,400 Y_p-units per unit base = 2,400	2,400 divisions in 10 in. on both axes
1,440 x 1,440 units/in.	Unit base = 0 X_p-units per unit base = 14,400 Y_p-units per unit base = 14,400	14,400 divisions in 10 in. on both axes

In row 1 of the table, the measurement base is 10 inches. The units per unit base value specifies 8,000 divisions in 10 inches, which is 800 divisions per inch. In row 2 of the table, the measurement base is 10 centimeters. The units per unit base value for the X_p-axis specifies 800 divisions in each 10 centimeters, which is 80 divisions per centimeter. The units per unit base value for the Y_p-axis specifies 770 divisions per measurement base, which is 77 divisions per centimeter. In this case, the units per unit base values differ along the X_p-axis and Y_p-axis.

If the value of the X_p-units per unit base parameter or the Y_p-units per unit base parameter is not supported or is not within the range specified by PTOCA, exception condition EC-0605 exists. The standard action in this case is to ignore the parameter and continue presentation with the value determined according to the hierarchy. The subset may limit the range permitted.

Measurement units specified for the X_p-axis may be different from the measurement units specified for the Y_p-axis.

Pragmatics: The measurement units for the I-axis are the same as the measurement units for the X_p,Y_p axis that is parallel to it. The units of the B-axis are the same as those of the other X_p,Y_p axis. The origin and orientation of the I-axis and B-axis can be specified by the Text Orientation initial text conditions, and by the Set Text Orientation control sequence. The values of the presentation text measurement units cannot be changed within a Presentation Text object, but the values of presentation text orientation can be changed.

Size Parameters

Semantics: Size consists of two parameters that specify the dimensions of a Presentation Text object, that is, the length of the X_p-axis and the Y_p-axis.

These dimensions, or *extents*, are described as follows.

X_p-extent A number specifying the size of the Presentation Text object along the X_p-axis.

Y_p-extent A number specifying the size of the Presentation Text object along the Y_p-axis.

Each extent is measured in X_p-units or Y_p-units. For information about the syntax of the extent fields, please see Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161.

If the value of the X_p-extent parameter or the Y_p-extent parameter is not supported or is not within the range specified by PTOCA, exception condition EC-0705 exists. The standard action is to ignore the invalid

parameter and continue presentation with the parameter value determined according to the hierarchy. The subset may limit the range permitted.

These extents are used in conjunction with the measurement units to specify the size of the Presentation Text object and for positions and displacements within the object.

Architecture Note: Note that when presentation text is processed in a MO:DCA environment where the Presentation Text Data Descriptor (PTD) is carried in the Active Environment Group (AEG) for the page, or when such text is processed in an IPDS environment, the Presentation Text object is bounded by the beginning of the page and the end of the page. This is sometimes called a *text major* environment. When the PTD is carried in the Object Environment Group (OEG) of a MO:DCA text object, the text object is bounded by the Begin Presentation Text (BPT) and End Presentation Text (EPT) structured fields. For such objects, the PTD in the AEG is ignored.

The range of supported values is receiver dependent.

Pragmatics: If the object's origin, orientation, and size are such that the object projects beyond the associated object area, the applicable data-stream standard action or default action must be invoked. This allows the controlling environment to control clipping.

If a control sequence, parameter, or other data element within the object causes presentation outside of the object space, exception condition EC-0103 exists when an attempt is made to present. The standard action is to ignore the character or control sequence that is presenting outside of the object space.

If any part of the character box for a character exceeds the boundaries of the object space, an exception condition exists even though the character's presentation position is within the object space. Please see “Graphic Character Processing” on page 38.

Exception Conditions: The measurement units parameters and the size parameters can cause the following exception conditions:

- EC-0505...The value of the unit base parameter is not supported, or is not in the range specified by PTOCA.
- EC-0605...The value of the X_p-units per unit base parameter or the Y_p-units per unit base parameter is not supported, or is not in the range specified by PTOCA.
- EC-0705...The value of the X_p-extent parameter or the Y_p-extent parameter is not supported, or is not in the range specified by PTOCA.
- EC-0103...The contents of the Presentation Text object will cause presentation outside of the object space.

Presentation Text Flags

Semantics: This parameter is reserved. Generators should set it to zero and receivers should ignore it.

Initial Text Condition Parameters

The Initial Text Condition parameters specify initial values for the Presentation Text object. They include the following parameters:

- Baseline increment
- Coded font local ID
- Extended text color
- Initial baseline coordinate
- Initial inline coordinate
- Inline margin
- Intercharacter adjustment
- Text color
- Text orientation

The following pages contain the semantics and the pragmatics of the PTOCA initial text condition parameters. For the syntax, please see Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161.

Baseline Increment

Baseline Increment specifies the distance between successive baselines.

Semantics: This parameter specifies the initial value of the increment in the B-direction from the current baseline position to a new baseline position. If the value of this parameter is not supported or is not within the range specified, exception condition EC-1101 exists. The standard action is to ignore this parameter and continue presentation with the value determined according to the hierarchy. The subset level may limit the range permitted. See Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161 for more information about valid ranges.

The default value is the Default Baseline Increment associated with the default coded font of the device.

Pragmatics: The baseline position is incremented by this value after each Begin Line control sequence is encountered in a text stream. If the value of this parameter is zero, each line appears superimposed over the preceding line. This value can be overridden for the duration of a Presentation Text object by a Set Baseline Increment control sequence.

Exception Conditions

This parameter can cause the following exception condition:

- EC-1101...The value of the baseline increment parameter is not supported or is not in the range specified by PTOCA.

Coded Font Local ID

coded Font Local ID specifies a font.

Semantics: This parameter specifies the initial value of the coded font local identifier (LID). The LID is used by the controlling environment to access a coded font when presenting subsequent text in the current Presentation Text object. If the value of the LID is not supported or is not within the range specified, exception condition EC-0C01 exists. The standard action is to ignore this parameter and to use the active coded font and character attributes determined according to the hierarchy. The subset level may also limit the range permitted. See Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161 for more information about valid ranges.

The default value is the LID of the default coded font of the device.

Pragmatics: The LID is equated to a coded font, character rotation, and font modification parameters by a mapping function in the controlling environment. If the text orientation is changed and the coded font selected by this parameter is not compatible with the new orientation, when an attempt is made to present a character from the selected coded font, exception condition EC-3F02 exists. The standard action is to complete the presentation at this presentation position using the receiver's default font. This results in the best possible presentation within the limits of the receiver's capability. If there is an intercharacter adjustment value, it is not applied. The measurement units used in the object and those used by the coded font selected for presentation are assumed to be compatible. If the measurement units are not compatible, the standard action is to present a best fit of the character and continue processing. A best fit could result in no mark or unintelligible marks on the presentation surface. Incompatibility of measurement units is not an exception condition in PTOCA.

Exception condition EC-1802 occurs in the following cases:
- The Coded Font Local ID parameter is present in the Presentation Text Data Descriptor but a corresponding mapping function does not exist in the controlling environment.
- An equate of the local identifier to a global identifier does not exist, or substitution parameters do not exist, in the controlling environment. The coded font identified by the map is not available in the receiver.

The standard action is to substitute the receiver's default font for the requested font and continue processing.

Exception Conditions

This parameter can cause the following exception conditions:
- EC-0C01...The value of the LID parameter is not supported or is not in the range specified by PTOCA.
- EC-3F02...The font is not compatible with the text orientation.
- EC-1802...The font requested cannot be provided.

Extended Text Color

The Extended Text Color parameter specifies a process or highlight color value and defines the color space and encoding for that value. The specified color value is applied to foreground areas of the text presentation space. Foreground areas consist of the following:
- The stroked and filled areas of solid text characters, including overstrike characters. With hollow characters, only the stroked portion of the character is considered foreground.
- The stroked area of a rule.
- The stroked area of an underscore.

All other areas of the text presentation space are considered background.

Semantics: Refer to "Set Extended Text Color (SEC)" on page 83 for a description of the parameter semantics. The controlling environment may limit the range permitted. See "Related Publications" on page vi for the appropriate data-stream documentation. Note that the Extended Text Color parameter is not supported as an initial text condition in IPDS Environments, see "Presentation Text Data Descriptor for Text-Major Text" on page 162.

Pragmatics: If the receiver does not support the specified color value exception condition EC-0E03 exists. The standard action in this case is to use the presentation device default color.

Exception Conditions

This parameter can cause the following exception conditions:
- EC-0E02...Invalid or unsupported color space
- EC-0E03...Invalid or unsupported color value
- EC-0E04...Invalid percent value
- EC-0E05...Invalid or unsupported number of bits in a color component

Initial Baseline Coordinate

The Initial Baseline Coordinate specifies the point on the B-axis within the object space to which the current addressable position will be initialized. This value is called the B-displacement.

Semantics: This parameter specifies a displacement in the B-direction from the I-axis for the initial addressable position. If the value of the B-displacement is not supported or is not within the range specified, exception condition EC-6B02 exists. The standard action in this case is to ignore the invalid parameter and continue presentation with the value determined according to the hierarchy. The subset level may also limit the range permitted. See Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161 for more information about valid ranges.

The default value is device-dependent.

Pragmatics: If the value of the B-displacement is X'0000', the initial addressable position in the B-direction is at the I-axis. This does not affect the inline margin.

Exception Conditions

This parameter can cause the following exception condition:
- EC-6B02...The value of the B-displacement parameter is not supported or is not in the range specified by PTOCA.

Initial Inline Coordinate

The Initial Inline Coordinate specifies the point on the I-axis within the object space to which the current addressable position will be initialized. This value is called the I-displacement.

Semantics: This parameter specifies a displacement in the I-direction from the B-axis for the initial addressable position. If the value of the I-displacement is not supported or is not within the range specified, exception condition EC-6A02 exists. The standard action in this case is to ignore the invalid parameter and continue presentation with the value determined according to the hierarchy. The subset level may also limit the range permitted. See Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161 for more information about valid ranges.

The default value is zero.

Pragmatics: If the value of the I-displacement is X'0000', the initial addressable position in the I-direction is at the B-axis. This does not affect the inline margin.

Exception Conditions

This parameter can cause the following exception condition:
- EC-6A02...The value of the I-displacement parameter is not supported or is not in the range specified by PTOCA.

Inline Margin

Inline Margin specifies the position of the inline margin.

Semantics: This parameter specifies the initial value for the location of the inline margin. This value is a displacement in the I-direction from the B-axis to the inline margin in the current Presentation Text object. If this parameter is not supported or is not within the range specified, the exception condition EC-1001 exists. The standard action is to ignore this parameter and continue presentation with the value determined according to the hierarchy. The subset level may limit the range permitted. See Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161 for more information about valid ranges.

The default value is zero.

Pragmatics: The addressable position for the first line of presentation text is specified by the Initial Baseline Coordinate and Initial Inline Coordinate initial text conditions. After each Begin Line control sequence is processed, the addressable position is placed at the inline margin. If the value of the inline margin parameter is X'0000', the inline margin is at the B-axis.

Exception Conditions

This parameter can cause the following exception condition:
- EC-1001...The displacement parameter is not supported or is not in the range specified by PTOCA.

Intercharacter Adjustment

Intercharacter Adjustment specifies additional increment or decrement between graphic characters.

Semantics: The adjustment parameter specifies the initial value of additional space between graphic characters. This space is in the I-direction from the end of the current character increment to the presentation position of the following graphic character. When this value is positive, the adjustment is called an increment. When the value is negative, the adjustment is called a decrement. The direction parameter specifies the direction in which the intercharacter adjustment is to be applied. Intercharacter increment, which occurs when the direction parameter is X'00', is applied in the positive I-direction. Intercharacter decrement, which occurs when the direction parameter is X'01', is applied in the negative I-direction. The default value for the direction parameter is X'00'.

Intercharacter adjustment is not applied before or after the following:
- A space character or variable space character
- A Begin Line control sequence
- A Relative Move Inline control sequence
- An Absolute Move Inline control sequence

For non-incrementing characters, the adjustment is applied from the current addressable position to locate the presentation position of the non-incrementing character, but the current addressable position is unchanged. The effect is that the non-incrementing character may be coupled with a graphic character, resulting in an overstrike function, and the adjustment is applied to the coupled characters.

If the value of the adjustment parameter or the direction parameter is not supported or is not within the range specified, exception condition EC-1201 exists. The standard action in this case is to ignore the Intercharacter Adjustment parameter and continue presentation with the parameter values determined according to the hierarchy. The subset level may also limit the range permitted. See Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161 for more information about valid ranges.

The default value is zero for both the adjustment parameter and the direction parameter.

Pragmatics: If the value of the adjustment parameter is X'0000', no additional intercharacter increment or decrement appears between graphic characters.

Exception Conditions

This parameter can cause the following exception conditions:
- EC-1201...The value of the adjustment parameter or the direction parameter is not supported, or is not in the range specified by PTOCA.

Text Color

The Text Color parameter specifies a named color that selects the foreground color of subsequent text characters, rules, and underscores.

Architecture Note: Pre-year 2000 applications and printers support an optional PRECSION parameter for text color. This parameter has been retired. It should not be generated by new applications, and should be ignored by new printers. For a definition of this parameter, see "Retired Parameters" on page 169.

Semantics: The named color value is applied to foreground areas of the text presentation space. Foreground areas consist of the following:

- The stroked and filled areas of solid text characters, including overstrike characters. With hollow characters, only the stroked portion of the character is considered foreground.
- The stroked area of a rule.
- The stroked area of an underscore.

All other areas of the text presentation space are considered background. Please refer to Table 14 on page 94 for the foreground color values and their associated colors. The default color attribute value is X'FF07'.

If the value of the foreground color attribute is not supported or is not within the range specified by PTOCA, exception condition EC-5803 exists. The standard action is to ignore these parameters and continue presentation with the value determined according to the hierarchy. The controlling environment may limit the range permitted. See "Related Publications" on page vi for the appropriate data-stream documentation.

Pragmatics:

The default color attribute value (FRGCOLOR = X'FFFF') is determined hierarchically. The following order applies:

1. Value set by Text Color initial text condition parameter in Descriptor
2. PTOCA default – X'FF07'.

The *device default value* is the receiver's default. For example, characters, rules, and underscores will be presented in black on a receiver which supports only black. The receiver's best possible value means that if the receiver has limited color capabilities, then it may substitute a color it supports for one it does not support. For example, the receiver may substitute red for pink, blue for turquoise, and so forth.

The color attribute values X'FF00' to X'FF06' are translated to X'0000' to X'0006' and treated exactly like those colors. The PTOCA default value of X'FF07' is the device default. For example, characters, rules, and underscores will be presented in black on a device which supports only black. A color attribute value of X'FF08' means that the receiver's default background color should be used for the foreground color. This provides an erase function.

Exception Conditions

This parameter can cause the following exception condition:

- EC-5803...The foreground color parameter (FRGCOLOR) value is invalid, or the specified color is not supported.

Text Orientation

Text Orientation consists of two parameters that establish the I-direction and the B-direction for presentation text.

Semantics: These parameters specify the initial I-axis and B-axis orientations with respect to the X_p-axis. These orientations are expressed in degrees and minutes. The same format is used for both orientations. Each is a two-byte, three-part code of the form ABC.

- A is a nine-bit binary number (bits 0 - 8) which provides from 0 through 359 degrees. Values from 360 through 511 are invalid.
- B is a six-bit binary number (bits 9 - 14) which provides from 0 through 59 minutes. Values from 60 through 63 are invalid.
- C is a one-bit reserved field (bit 15) which must be B'0'.

A value of X'0000' for the I-axis orientation parameter indicates that the positive direction is parallel to the X_p-axis. Increasing values for these parameters indicate increasing clockwise rotation.

The default value for the I-axis is X'0000', that is, zero degrees. The default value for the B-axis is X'2D00', that is, 90 degrees. The maximum value for IORNTION and BORNTION is X'B3F6' or B'1011001111110110', which is 359 degrees and 59 minutes.

Pragmatics: I-direction is the direction in which additional characters appear in a line of text. B-direction is the direction in which additional lines of text appear. If the I-axis is not parallel to either the X_p-axis or Y_p-axis, exception condition EC-6802 exists. If the B-axis is not parallel to either the X_p-axis or Y_p-axis, exception condition EC-6902 exists. The standard action in this case is to set the I-axis equal to the X_p-axis direction and the B-axis equal to the Y_p-axis direction. The origin of the I-axis and the B-axis is always one of the four corners of the object space.

Orientations other than 0,90 are valid, but may be constrained by receiver limitations or parameters in controlling environment. If the I-axis orientation is not supported by the receiver, exception condition EC-6802 exists. If the B-axis orientation is not supported by the receiver, exception condition EC-6902 exists. The standard action is to use 0,90 degrees. These exception condition codes and standard actions also apply to orientation values not within the range specified by PTOCA.

Exception Conditions

This parameter can cause the following exception conditions:

- EC-6802...The I-axis is not parallel to the X_p-axis or the Y_p-axis.
- EC-6902...The B-axis is not parallel to the X_p-axis or the Y_p-axis.

Chapter 5. Exception Handling in PTOCA

This chapter:
- Describes exception condition detection
- Describes exception responses and standard actions
- Lists the PTOCA exception condition codes

Faithful Reproduction

PTOCA is intended to be precise enough to permit multiple products to reproduce the Presentation Text object faithfully. Faithful reproduction includes such aspects as the size and relative positions of graphic characters and strings of graphic characters. Examples are the placement of columns in tables, mathematical constructs such as subscripts or limits of integrals, and the appearance of graphic characters. PTOCA can only make faithful reproduction possible. The responsibility for faithful reproduction belongs to the process that presents the object.

PTOCA is also designed to permit less than faithful reproduction of the Presentation Text object. It is possible to specify those exception conditions for which continuation of processing is acceptable. This permits a process that cannot faithfully reproduce the object to continue with its best approximation. If less than faithful reproduction is acceptable for an application, interchange among a larger set of receivers is possible.

If a requirement for faithful reproduction is specified, and if a process cannot present a faithful reproduction, reproduction is not continued.

To satisfy these objectives, PTOCA anticipates the existence of exception conditions, specifies how each is to be handled so that results are predictable, and lets the controlling environment control exception condition actions.

Exception Conditions

An exception condition in the object is the appearance of the following:
- Invalid or unsupported parameter value
- Invalid or unsupported parameter
- Invalid or unsupported control sequence

PTOCA specifies valid values for parameters, appropriate and inappropriate parameters, and valid and invalid combinations of control sequences. In addition, an implementation may accept only a subset of valid values or only a subset of appropriate parameters and control sequences. However, PTOCA specifies, by subsetting, which of its controls and parameters are to be supported by the implementations of a subset.

Exception conditions can be classified as:
- Syntactic
- Semantic
- Pragmatic

A syntactic exception condition is a violation of a structural architectural specification.

Syntactic exception conditions defined for PTOCA include:
- Invalid control sequence
- Invalid parameter value
- Control sequence appearing in invalid context

A semantic exception condition is a violation of a functional architectural specification, that is, what a parameter, structured field, or control sequence does, independent of how it looks or how it is used.

Semantic exception conditions defined for PTOCA include:
- Selection of inconsistent or contradictory functions
- Loss of presentation information

A pragmatic exception condition is an incorrect usage of an architectural specification that is valid structurally and semantically. It is normally caused by an incompatibility between a Presentation Text object and a product that processes or presents it. Pragmatic exception conditions are not defined by PTOCA and cannot be detected by inspection of a Presentation Text object.

Pragmatic exception conditions include:
- Mismatch of characteristics of Presentation Text object and presentation product
- Unavailable resource, for example, coded font
- Unavailable function, for example, overstrike
- Unsupported control sequence

A product may be unable to distinguish between a syntactic exception and a pragmatic exception, for example, between an invalid parameter value and a parameter value out of the product's range.

Exception Condition Detection

A potential exception condition may exist, yet not be detected during processing of a Presentation Text object. A receiver is not required to process a Presentation Text object beyond its need to perform a specified function. Therefore, a process usually detects only those exception conditions that pertain to the function it is performing. PTOCA defines specific exception conditions that occur within the object, and enables the controlling environment to provide a continuation capability by specifying standard action values to use when no other source is available for the parameter values. In addition, PTOCA does not require that an implementation of the controlling environment do anything more than receive exception conditions and terminate processing the Presentation Text object as a result of them. However, the controlling environment may place more stringent requirements.

Syntactic exception conditions can be detected without regard to the value of any other parameter or structured field. A syntactic or semantic exception condition can be detected by inspection of a Presentation Text object. A pragmatic exception condition cannot be detected by inspection of a Presentation Text object alone, but requires knowledge of characteristics of the receiver. If a product that produces or processes a Presentation Text object knows the characteristics of one or more receivers, it can avoid or detect pragmatic exception conditions. If it does not, this detection must be performed by the receiver.

Exception Condition Handling

All processors of a Presentation Text object need not have the same capability for detecting, processing, or reporting exception conditions. Processors of similar capabilities may handle the same exception condition in different ways. A processor may provide alternative ways to handle an exception condition; however, the processor's capabilities are limited by PTOCA.

The controlling environment of a Presentation Text object can specify the response a receiver should make when exception conditions are encountered. This is specified in structures contained in the controlling environment. These structures identify one or more exception conditions, and specify the exception response or effect the exception condition will have on the document presentation process. For example, a document may be terminated when an exception condition is encountered, or processing may continue using the architected standard action for the exception condition.

A standard action is specified in PTOCA for many exception conditions. For example, if an implementation cannot process some of the Presentation Text object, the standard action could be to present it with

unrecognized control sequences omitted or with specified valid parameters substituted for invalid parameters. The standard actions are defined independent of where the exception condition is detected. That is, the receiver may be an application, program product, mechanical device, and so forth. The process always initiates the specified action, and is responsible for its satisfactory completion.

Exception Responses

When an exception condition is detected by a receiver, an exception response is assumed. Exception responses are not specified by PTOCA. The exception condition codes specified are for reference purposes. If the controlling environment specifies a different exception condition code for the same exception condition, the controlling environment's exception condition code overrides the code specified by PTOCA. For example, if a DBR control sequence will cause a rule to extend outside the boundaries of the object space, PTOCA specifies that exception condition code EC-0103 be recognized. However, in the IPDS environment, exception ID 08C1..00 would be recognized. Please see Appendix A, “MO:DCA Environment”, on page 155 and Appendix B, “IPDS Environment”, on page 161. See “Related Publications” on page vi for data-stream documentation.

Some exception responses can be common to all exception conditions. Others are specific to particular exception conditions.

Exception responses that can be common to all exception conditions include the following:
- Terminate processing Presentation Text object
- Ignore the control that caused the exception condition and continue processing the object
- Partially process the control that caused the exception condition
- Report exception condition back to generator or forward it to the presenter of the object
- Cause an intervention-required condition to occur at the receiver
- Mark the presentation information with diagnostic information

An example of an exception response that can vary depending on the exception condition is to present the data that caused the exception condition. The data presented may be a receiver-dependent approximation if faithful reproduction is not required. The data presented may be a special, recognizable marker instead of, or in addition to, other data at the position of an exception condition. For example, a blotch may appear where a rule is drawn beyond an object space extent. Another exception response is to present no data at the position of an exception condition.

A response for each exception condition may be selected in a manner independent of any other exception condition. Multiple responses may be selected for one exception condition. Certain exception condition actions are mutually exclusive by their nature. PTOCA assumes that the controlling environment provides structures external to the Presentation Text object for handling the responses to exception classes or specific exception conditions received from the object processor.

Standard Actions

PTOCA specifies the standard actions that it assumes to be taken by the Presentation Text object processor for specific exception conditions that can occur in the object. The receiver is expected to implement the PTOCA standard action for those exception conditions it can recognize as part of the support of the subset level. If the controlling environment specifies a different standard action for the same exception condition, the action specified by the controlling environment overrides the action specified by PTOCA. For example, if an invalid control sequence generates an exception condition, PTOCA may specify that the presentation process ignore the invalid control sequence and continue presenting. However, for the same exception condition, IPDS may require the presentation process to stop processing the Presentation Text object. Thus the PTOCA processor has two options when it recognizes an exception condition: it can simply report the exception condition to the controlling environment, and let the environment handle it; or it can apply the PTOCA standard action.

Exception Condition Codes

The following list contains the exception conditions that implementations of PTOCA are obligated to test for. The codes are consistent with those of the data streams, and conform to the registry of exception condition codes found in IPDS. Please see the appendixes for information about exception conditions in the controlling environment.

Table 19. PTOCA Exception Conditions

PTOCA Exception Condition	Meaning	Comments
EC-0001	Invalid text control.	• Invalid or unsupported function type in control sequence. • Invalid control sequence or initial text condition parameter. • Invalid or unsupported initial text condition parameter identifier. • Control sequence or initial text condition parameter is not in the supported subset.
EC-0103	Character box exceeds object space.	• A character has been positioned so that a portion of its character box will exceed the object space in the I-direction or the B-direction. • *Caution* - this exception condition is applicable only within a valid object area. If the boundary of the object space coincides with or extends beyond the boundary of the object area, the appropriate data-stream exception may take precedence over this exception condition.
EC-0201	Invalid LID in ESU.	• The active BSU LID is not the same as the LID specified in the ESU. • No active BSU LID when an ESU is processed.
EC-0401	The end of the object is encountered before a text suppression has ended.	
EC-0505	PTD unit base specified is not a valid or supported value.	
EC-0601	Nested BSU.	• BSU is encountered before the previous suppression has ended.
EC-0605	PTD X_p- or Y_p–units per unit base specified is not a valid or supported value.	
EC-0705	PTD X_p- or Y_p–extent specified is not a valid or supported value.	
EC-0C01	Coded Font LID specified is not a valid or supported value.	
EC-0E02	Invalid or unsupported color space in SEC.	
EC-0E03	Invalid or unsupported color value in SEC.	
EC-0E04	Invalid percent value in SEC.	The percent coverage parameter or the percent shading parameter in the SEC (Highlight color space) contains an invalid value.
EC-0E05	Invalid or unsupported number of bits for a color component in SEC.	

Table 19 PTOCA Exception Conditions (cont'd.)

PTOCA Exception Condition	Meaning	Comments
EC-0F01	Invalid text orientation in STO.	• Baseline or inline orientation specified is not a valid or supported value. • The I and B orientations are identical. • Neither the I-direction nor the B-direction is parallel to the X_p–direction.
EC-1001	SIM displacement specified is not a valid or supported value.	
EC-1101	SBI baseline increment specified is not a valid or supported value.	
EC-1201	SIA adjustment or direction specified is not a valid or supported value.	
EC-1301	AMB displacement specified is not a valid or supported value.	
EC-1401	AMI displacement specified is not a valid or supported value.	
EC-1403	Advancement of the baseline coordinate toward the I-axis is specified but is not supported.	
EC-1501	RMI increment specified is not a supported value.	
EC-1601	RMB increment specified is not a supported value.	
EC-1701	SVI increment specified is not a valid or supported value.	
EC-1802	Invalid Coded Font LID.	• The necessary mapping is not provided to support the specified coded font. • The specified coded font is not available to the receiver.
EC-1901	Invalid or unsupported RPS target count.	• The target count parameter for RPS is invalid or unsupported.
EC-1A01	RPS or TRN source string length error or UCT data length error.	• The data string length for TRN or RPS is an odd number. It must be even for double-byte fonts. • The byte count specified for code points following UCT is an odd number. It must be even for double–byte encoded data.

Table 19 PTOCA Exception Conditions (cont'd.)

PTOCA Exception Condition	Meaning	Comments
EC-1A03	Invalid Unicode data.	• A high-order surrogate code value was not immediately followed by a low-order surrogate code value. The high-order surrogate range is U+D800 through U+DBFF. • A low-order surrogate code value was not immediately preceded by a high-order surrogate code value. The low-order surrogate range is U+DC00 through U+DFFF. • An illegal UTF-8 byte sequence, as defined in the Unicode 3.2 Specification, was specified. For a discussion of illegal UTF-8 byte sequences, see the Application Note on page 120.
EC-1B01	RPS target string length error.	• The RPS repeat length is an odd number when a double-byte font is active. It must be even for double-byte fonts.
EC-1C01	Invalid control sequence class.	• The class of a X'2B' control sequence is not X'D3'.
EC-1E01	Invalid length for control sequence or initial text condition parameter.	• A required parameter has not been not specified. • Invalid control sequence or initial text condition parameter length. • Part of an optional parameter in a control sequence is missing. • A Coded Font LID has been omitted in a SCFL control sequence or in a Coded Font Local ID initial text condition parameter. • SVI control sequence increment parameter is missing. • DBR or DIR length parameter is missing. • SIM displacement parameter is missing. • I-orientation parameter or B-orientation parameter is missing in an STO control sequence.
EC-1F01	RPS length error.	• The RPS control sequence length is four and the repeat length is not zero.
EC-2100	Invalid character.	
EC-3F02	Text orientation is incompatible with selected font.	
EC-5803	An STC color or color attribute cannot be supported as specified.	
EC-6802	Initial I-orientation specified is not a valid or supported value.	
EC-6902	Initial B-orientation specified is not a valid or supported value.	
EC-6A02	Initial I-displacement specified is not a valid or supported value.	
EC-6B02	Initial B-displacement specified is not a valid or supported value.	
EC-8002	DBR or DIR rule width specified is not a valid or supported value.	

Table 19 PTOCA Exception Conditions (cont'd.)

PTOCA Exception Condition	Meaning	Comments
EC-8202	DBR or DIR rule length specified is not a valid or supported value.	
EC-9801	BSU or ESU LID specified is not a valid or supported value.	
EC-9803	TBM error.	• Receiver is unable to support TBM by printing full size characters. • Receiver cannot support substitution character in the TBM field. • Temporary move size exceeds the device limitations. • Substitution method receiver cannot support multi-offset temporary baselines.
EC-9A01	OVS selected overstrike character that has an invalid character increment or is a non-printing character.	• Character increment of overstrike character is less than or equal to zero. • Character increment of overstrike character is less than the character-box size. • Overstrike character is a non-printing character.
EC-9B01	UCT parameter values for CTLNGTH, UCTVERS, BIDICT, or GLYPHCT are invalid.	

Chapter 6. Compliance with PTOCA

This chapter:

- Describes the base level of PTOCA
- Describes the PT1 subset of PTOCA
- Describes the PT2 subset of PTOCA
- Describes the PT3 subset of PTOCA
- Describes the PT4 subset of PTOCA
- States general requirements for compliance

Base Level

The mandatory base level contains functions and facilities required by all implementations. By itself the mandatory base level is not sufficient; it must always be supplemented by a higher subset such as PT1, PT2, or PT3. The base level represents the base set of functions defined within PTOCA. It is the minimum set of functions required to be supported in any environment, and consists of the following minimum general communication capabilities:

- Recognition of control sequences, chained or unchained
- Interpretation and validation of the control sequence
- Rejection of control sequences and parameters, including return of error data, that are not supported within the supported subset
- Reporting, on request from the controlling environment, the supported features
- Reporting exception conditions to the controlling environment

PT1 Subset

The following table shows the control sequences that are valid for a PT1 subset compliant receiver, and the associated parameter ranges. The offset shown in the table indicates the beginning of the parameter data within the control sequence.

Control Sequence	Offset (Unchained)	Parameter	PT1 Range	Notes
AMB	4-5	DSPLCMNT	X'0000'-X'7FFF'	4
AMI	4-5	DSPLCMNT	X'0000'-X'7FFF'	4
BLN				6
BSU	4	LID	X'01'-X'7F'	
DBR	4-5	RLENGTH	X'0000'-X'7FFF'	1,4
	6-8	RWIDTH	X'0000'-X'00C0'	1,2,3
DIR	4-5	RLENGTH	X'0000'-X'7FFF'	1,4
	6-8	RWIDTH	X'0000'-X'00C0'	1,2,3
ESU	4	LID	X'01'-X'7F'	
NOP	4-256	IGNDATA		7
RMB	4-5	INCRMENT	X'8000'-X'7FFF'	1,4
RMI	4-5	INCRMENT	X'8000'-X'7FFF'	1,4
RPS	4-5	RLENGTH	X'0000'-X'7FFF'	
	6-256	RPTDATA		8
SBI	4-5	INCRMENT	X'0000'-X'7FFF'	3,4
SCFL	4	LID	X'00'-X'7F'	3
SIA	4-5	ADJSTMNT	X'0000'-X'0FFF'	3,4
	6	DIRCTION	X'00'	9
SIM	4-5	DSPLCMNT	X'0000'-X'7FFF'	3,4
STC	4-5	FRGCOLOR	X'FF07'	3,5
	6	PRECSION	X'00'-X'01'	3,10
STO	4-5	IORNTION	X'0000', X'2D00', X'5A00', X'8700'	3
	6-7	BORNTION	X'0000', X'2D00', X'5A00', X'8700'	3
SVI	4-5	INCRMENT	X'0000'-X'0FFF'	3,4
TRN	4-256	TRNDATA		8

Notes:

1. This parameter is a signed binary number that may be positive or negative. Negative numbers are expressed in twos-complement form.
2. The PTOCA range for RWIDTH is X'8000' - X'7FFF' plus a fractional value byte ranging from X'00' - X'FF'. That is, the fractional values range from 1/2 measurement unit to 1/256 measurement unit. The PT1 subset range for RWIDTH is X'0000' - X'00C0' in bytes 6 and 7. Byte 8 is always X'00', unless bytes 6 and 7 are X'FFFF', in which case byte 8 is X'FF'.
3. The default indicator is allowed, meaning obtain a value from the hierarchy.
4. The range values shown assume a measurement unit of 1/1440 inch. That is, the measurement base is assumed to be ten inches, and the X_p-units per unit base and Y_p-units per unit base are assumed to be

14,400. If a different measurement unit is used, the correct range values can be determined using the conversion routine described in "Interpreting Ranges" on page 45.

5. For the PTOCA range, see "Set Text Color (STC)" on page 93. The PT1 range is X'FF07'.
6. The Begin Line (BLN) control sequence has no parameters.
7. The No Operation (NOP) control sequence may contain any data that does not exceed the field length. The data is ignored.
8. The Transparent Data (TRN) and Repeat String (RPS) control sequences may contain any data that does not exceed the field length. However, the data will be presented as a character string.
9. The default indicator is not allowed for this parameter in this subset.
10. The STC PRECSION parameter has been retired; see "Retired Parameters" on page 169. New PTOCA generators should not specify this parameter and new receivers should ignore it..

PT2 Subset

The following table shows the control sequences that are valid for a PT2 subset compliant receiver, and the associated parameter ranges. The offset shown in the table indicates the beginning of the parameter data within the control sequence.

Control Sequence	Offset (Unchained)	Parameter	PT2 Range	Notes
AMB	4-5	DSPLCMNT	X'0000'-X'7FFF'	4
AMI	4-5	DSPLCMNT	X'0000'-X'7FFF'	4
BLN				6
BSU	4	LID	X'01'-X'7F'	
DBR	4-5	RLENGTH	X'0000'-X'7FFF'	1,4
	6-8	RWIDTH	X'0000'-X'00C0'	1,2,3
DIR	4-5	RLENGTH	X'0000'-X'7FFF'	1,4
	6-8	RWIDTH	X'0000'-X'00C0'	1,2,3
ESU	4	LID	X'01'-'X'7F'	
NOP	4-256	IGNDATA		7
OVS	4	BYPSIDEN	X'00'-X'0E'	3
	5-6	OVERCHAR	X'0000'-X'FFFF'	
RMB	4-5	INCRMENT	X'8000'-X'7FFF'	1,4
RMI	4-5	INCRMENT	X'8000'-X'7FFF'	1,4
RPS	4-5	RLENGTH	X'0000'-X'7FFF'	
	6-256	RPTDATA		8
SBI	4-5	INCRMENT	X'0000'-X'7FFF'	3,4
SCFL	4	LID	X'00'-X'FE'	3
SIA	4-5	ADJSTMNT	X'0000'-X'0FFF'	3,4
	6	DIRCTION	X'00'-X'01'	3
SIM	4-5	DSPLCMNT	X'0000'-X'7FFF'	3,4
STC	4-5	FRGCOLOR	X'0000', X'FF00', X'FF07', X'FFFF'	3,5
	6	PRECSION	X'00'-X'01'	3,9
STO	4-5	IORNTION	X'0000', X'2D00', X'5A00', X'8700'	3
	6-7	BORNTION	X'0000', X'2D00', X'5A00', X'8700'	3
SVI	4-5	INCRMENT	X'0000'-X'0FFF'	3,4
TBM	4	DIRCTION	X'00'-X'03'	3
	5	PRECSION	X'00'-X'01'	3
	6-7	INCRMENT	X'0000'-X'7FFF'	3,4
TRN	4-256	TRNDATA		8
USC	4	BYPSIDEN	X'00'-X'0E'	3

Notes:

1. This parameter is a signed binary number that may be positive or negative. Negative numbers are expressed in twos-complement form.
2. The PTOCA range for RWIDTH is X'8000' - X'7FFF' plus a fractional value byte ranging from X'00' - X'FF'. That is, the fractional values range from 1/2 measurement unit to 1/256 measurement unit. The PT2 subset range for RWIDTH is X'0000' - X'00C0' in bytes 6 and 7. Byte 8 is always X'00', unless bytes 6 and 7 are X'FFFF', in which case byte 8 is X'FF'.
3. The default indicator is allowed, meaning obtain a value from the hierarchy.
4. The range values shown assume a measurement unit of 1/1440 inch. That is, the measurement base is assumed to be ten inches, and the X_p-units per unit base and Y_p-units per unit base are assumed to be 14,400. If a different measurement unit is used, the correct range values can be determined using the conversion routine described in “Interpreting Ranges” on page 45.
5. For the PTOCA range, see “Set Text Color (STC)” on page 93. The PT2 range is X'0000', X'FF00', X'FF07', and X'FFFF'.
6. The Begin Line (BLN) control sequence has no parameters.
7. The No Operation (NOP) control sequence may contain any data that does not exceed the field length.
8. The Transparent Data (TRN) and Repeat String (RPS) control sequences may contain any data that does not exceed the field length. However, the data is presented as a character string.
9. The STC PRECSION parameter has been retired; see “Retired Parameters” on page 169. New PTOCA generators should not specify this parameter and new receivers should ignore it.

PT3 Subset

The following table shows the control sequences that are valid for a PT3 subset compliant receiver, and the associated parameter ranges. The offset shown in the table indicates the beginning of the parameter data within the control sequence.

Control Sequence	Offset (Unchained)	Parameter	PT3 Range	Notes
AMB	4-5	DSPLCMNT	X'0000'-X'7FFF'	4
AMI	4-5	DSPLCMNT	X'0000'-X'7FFF'	4
BLN				6
BSU	4	LID	X'01'-X'7F'	
DBR	4-5	RLENGTH	X'0000'-X'7FFF'	1,4
	6-8	RWIDTH	X'0000'-X'00C0'	1,2,3
DIR	4-5	RLENGTH	X'0000'-X'7FFF'	1,4
	6-8	RWIDTH	X'0000'-X'00C0'	1,2,3
ESU	4	LID	X'01'-'X'7F'	
NOP	4-256	IGNDATA		7
OVS	4	BYPSIDEN	X'00'-X'0E'	3
	5-6	OVERCHAR	X'0000'-X'FFFF'	
RMB	4-5	INCRMENT	X'8000'-X'7FFF'	1,4
RMI	4-5	INCRMENT	X'8000'-X'7FFF'	1,4
RPS	4-5	RLENGTH	X'0000'-X'7FFF'	
	6-256	RPTDATA		8
SBI	4-5	INCRMENT	X'0000'-X'7FFF'	3,4
SCFL	4	LID	X'00'-X'FE'	3
SEC	5 10 11 12 13 14-17	COLSPCE COLSIZE1 COLSIZE2 COLSIZE3 COLSIZE4 COLVALUE	X'01', X'04', X'06' X'08', X'40' X'01'-X'08', X'10' X'00'-X'08' X'00'-X'08' X'00'-X'08' Full range allowed by the color space	
SIA	4-5	ADJSTMNT	X'0000'-X'0FFF'	3,4
	6	DIRCTION	X'00'-X'01'	3
SIM	4-5	DSPLCMNT	X'0000'-X'7FFF'	3,4
STC	4-5	FRGCOLOR	X'0000', X'FF00', X'FF07', X'FFFF'	3,5
	6	PRECSION	X'00'-X'01'	3,9
STO	4-5	IORNTION	X'0000', X'2D00', X'5A00', X'8700'	3
	6-7	BORNTION	X'0000', X'2D00', X'5A00', X'8700'	3
SVI	4-5	INCRMENT	X'0000'-X'0FFF'	3,4
TBM	4	DIRCTION	X'00'-X'03'	3

Control Sequence	Offset (Unchained)	Parameter	PT3 Range	Notes
	5	PRECSION	X'00'-X'01'	3
	6-7	INCRMENT	X'0000'-X'7FFF'	3,4
TRN	4-256	TRNDATA		8
USC	4	BYPSIDEN	X'00'-X'0E'	3

Notes:

1. This parameter is a signed binary number that may be positive or negative. Negative numbers are expressed in twos-complement form.
2. The PTOCA range for RWIDTH is X'8000' - X'7FFF' plus a fractional value byte ranging from X'00' - X'FF'. That is, the fractional values range from 1/2 measurement unit to 1/256 measurement unit. The PT3 subset range for RWIDTH is X'0000' - X'00C0' in bytes 6 and 7. Byte 8 is always X'00', unless bytes 6 and 7 are X'FFFF', in which case byte 8 is X'FF'.
3. The default indicator is allowed, meaning obtain a value from the hierarchy.
4. The range values shown assume a measurement unit of 1/1440 inch. That is, the measurement base is assumed to be ten inches, and the X_p-units per unit base and Y_p-units per unit base are assumed to be 14,400. If a different measurement unit is used, the correct range values can be determined using the conversion routine described in “Interpreting Ranges” on page 45.
5. For the PTOCA range, see “Set Text Color (STC)” on page 93. The PT3 range is X'0000', X'FF00', X'FF07', and X'FFFF'.
6. The Begin Line (BLN) control sequence has no parameters.
7. The No Operation (NOP) control sequence may contain any data that does not exceed the field length.
8. The Transparent Data (TRN) and Repeat String (RPS) control sequences may contain any data that does not exceed the field length. However, the data is presented as a character string.
9. The STC PRECSION parameter has been retired; see “Retired Parameters” on page 169. New PTOCA generators should not specify this parameter and new receivers should ignore it..

PT4 Subset

The following table shows the control sequences that are valid for a PT4 subset compliant receiver, and the associated parameter ranges. The offset shown in the table indicates the beginning of the parameter data within the control sequence.

Control Sequence	Offset (Unchained)	Parameter	PT4 Range	Notes
AMB	4-5	DSPLCMNT	X'0000'-X'7FFF'	3
AMI	4-5	DSPLCMNT	X'0000'-X'7FFF'	3
BLN				5
BSU	4	LID	X'01'-X'7F'	
DBR	4-5	RLENGTH	X'8000'-X'7FFF'	1, 3
	6-7 8	RWIDTH	X'8000'-X'7FFF' X'00'-X'FF'	1,2
DIR	4-5	RLENGTH	X'8000'-X'7FFF'	1,3
	6-7 8	RWIDTH	X'8000'-X'7FFF' X'00'-X'FF'	1,2
ESU	4	LID	X'01'-'X'7F'	
GAR	4–253 (chained)	ADVANCE	X'0000'-'X'FFFF'	3
GIR	4–253 (chained)	GLYPHID	X'0000'-'X'FFFF'	
GLC	4-5	IADVNCE	X'8000'-X'7FFF'	1, 3
	6	OIDLGTH	0, 13-129	
	7	FFNLGTH	0-(255-(10+OIDLGTH))	
	12 -n	FONTOID (first byte)	X'06'	
	(n+1)–p	FFONTNME	valid UTF-16BE code points	
GOR	4–253 (chained)	OFFSET	X'8000'-X'7FFF'	1
NOP	4-256	IGNDATA		6
OVS	4	BYPSIDEN	X'00'-X'0E'	2
	5-6	OVERCHAR	X'0000'-X'FFFF'	
RMB	4-5	INCRMENT	X'8000'-X'7FFF'	1,3
RMI	4-5	INCRMENT	X'8000'-X'7FFF'	1,3
RPS	4-5	RLENGTH	X'0000'-X'7FFF'	
	6-256	RPTDATA		7
SBI	4-5	INCRMENT	X'0000'-X'7FFF'	2,3
SCFL	4	LID	X'00'-X'FE'	2
SEC	5 10 11 12 13 14-17	COLSPCE COLSIZE1 COLSIZE2 COLSIZE3 COLSIZE4 COLVALUE	X'01', X'04', X'06' X'08', X'40' X'01'-X'08', X'10' X'00'-X'08' X'00'-X'08' X'00'-X'08' Full range allowed by the color space	

Control Sequence	Offset (Unchained)	Parameter	PT4 Range	Notes
SIA	4-5	ADJSTMNT	X'0000'-X'7FFF'	2,3
	6	DIRCTION	X'00'-X'01'	2
SIM	4-5	DSPLCMNT	X'0000'-X'7FFF'	2,3
STC	4-5	FRGCOLOR	X'0000'-X'0010', X'FF00'-X'FF08', X'FFFF'	2,4
	6	PRECSION	X'00'-X'01'	2,8
STO	4-5	IORNTION	X'0000', X'2D00', X'5A00', X'8700'	2
	6-7	BORNTION	X'0000', X'2D00', X'5A00', X'8700'	2
SVI	4-5	INCRMENT	X'0000'-X'7FFF'	2,3
TBM	4	DIRCTION	X'00'-X'03'	2
	5	PRECSION	X'00'-X'01'	2
	6-7	INCRMENT	X'0000'-X'7FFF'	2,3
TRN	4-256	TRNDATA		7
UCT				9
USC	4	BYPSIDEN	X'00'-X'0E'	2

Notes:

1. This parameter is a signed binary number that may be positive or negative. Negative numbers are expressed in twos-complement form.
2. The default indicator is allowed, meaning obtain a value from the hierarchy.
3. The range values shown assume a measurement unit of 1/1440 inch. That is, the measurement base is assumed to be ten inches, and the X_p-units per unit base and Y_p-units per unit base are assumed to be 14,400. If a different measurement unit is used, the correct range values can be determined using the conversion routine described in "Interpreting Ranges" on page 45.
4. For the PTOCA range, see "Set Text Color (STC)" on page 93. The PT4 range is the full Standard OCA Color Value Table found in the MO:DCA Reference.
5. The Begin Line (BLN) control sequence has no parameters.
6. The No Operation (NOP) control sequence may contain any data that does not exceed the field length.
7. The Transparent Data (TRN) and Repeat String (RPS) control sequences may contain any data that does not exceed the field length. However, the data is presented as a character string.
8. The STC PRECSION parameter has been retired; see "Retired Parameters" on page 169. New PTOCA generators should not specify this parameter and new receivers should ignore it.
9. The UCT must be chained to a GAR or GOR and is not rendered; all parameters are ignored.

General Requirements for Compliance

In claiming support as a PTOCA receiver, a product is stating that it has identified the subsets sufficient for its needs, and has implemented all mandatory and optional control sequences, parameters, and ranges within those subsets, including support of the PTOCA default values specified for those control sequences and parameters. A receiver may also support additional function beyond the PTOCA subset that it claims to support.

In claiming support as a PTOCA generator, a product is stating that it has identified which subset is sufficient for its needs, and that the Presentation Text object content it produces is limited to the control sequences and parameters defined in that subset. Additionally, it is stating that it has not violated the syntactic, semantic, and pragmatic restrictions specified in PTOCA. There is no requirement that any control sequence or parameter must appear in the object.

A receiver need not check the Presentation Text object for compliance to the PTOCA subset beyond what the receiver requires for the processing that represents its function. A printer, for example, normally checks all the control sequences, parameters, and ranges within the object it is presenting. But a transform product might only need to check the structured field introducers. A receiver may optionally provide warnings if it detects non-compliance.

Appendix A. MO:DCA Environment

This appendix describes how Presentation Text objects may be included within a MO:DCA document for the purpose of interchanging Presentation Text objects between a generating node and a receiving node. See *Mixed Object Document Content Architecture Reference,* AFPC-0004, for a full description of the MO:DCA architecture.

The Presentation Text Data Descriptor (PTD) and Presentation Text Data (PTX) structured fields, which carry Presentation Text objects in MO:DCA architecture documents, are defined in the following pages.

To guarantee interchange, a MO:DCA document carrying a Presentation Text object must include all information related to the object. The MO:DCA document must therefore contain not only the definition of the Presentation Text object, but it must also provide linkage to the resources the object references.

The discussion of MO:DCA structured fields is included in this appendix solely for setting the context of their use by PTOCA.

Compliance with MO:DCA Interchange Sets

When Presentation Text objects are interchanged with the purpose of outputting the objects on a display, printer, or other output device, it is very important that visual fidelity be maintained as far as possible. In an attempt to satisfy this objective, PTOCA defines the following for the MO:DCA environment:

- A set of rules that must be followed by all generators when constructing Presentation Text objects.
- A set of Presentation Text processing capabilities that are guaranteed to be supported by all receivers.

In order to comply with a particular MO:DCA Interchange Set, products that generate Presentation Text objects must only generate objects that contain Presentation Text items and values defined in that interchange set. Including items or values not in the interchange set can result in processing exceptions being raised by the receiving processor, and exception actions being taken. However, a generator must not rely on a receiver taking these actions.

In order to conform to a particular MO:DCA Interchange Set, products that receive Presentation Text objects and convert them using a processor for output to some device are required to support all the Presentation Text facilities defined in that interchange set.

It is optional whether the processor in the receiving node checks each Presentation Text object for compliance with the relevant interchange set. A receiver can optionally raise warnings, in the form of errors, if a non-compliant Presentation Text object is detected.

The Presentation Text object space is the presentation space within which the object generator may position graphic characters without error, and it is the responsibility of the generator to ensure that the graphic characters it places in the object space are positioned so that they do not exceed the object space.

Presentation Text Structured Fields in the MO:DCA Architecture

Presentation Text Data Descriptor (PTD)

The PTOCA Presentation Text Data Descriptor is carried in the MO:DCA Presentation Text Data Descriptor (PTD) structured field.

Structured Field Introducer				
SF Length	SF Identifier **X'D3B19B'**	Flags	Reserved **X'0000'**	PTD Data

The following table describes the contents of the Presentation Text Data Descriptor (PTD) structured field in the MO:DCA architecture, which has a structured field identifier of X'D3B19B'.

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	XPBASE	X'00'–X'01'	Unit base for X axis; must be the same as the unit base for Y axis: X'00' Ten inches X'01' Ten centimeters	M	N	N
1	CODE	YPBASE	X'00' –X'01'	Unit base for Y axis; must be the same as the unit base for X axis: X'00' Ten inches X'01' Ten centimeters	M	N	N
2–3	UBIN	XPUNITVL	X'0001' – X'7FFF'	X_p-units per unit base	M	N	N
4–5	UBIN	YPUNITVL	X'0001' – X'7FFF'	Y_p-units per unit base	M	N	N
6–8	UBIN	XPEXTENT	X'000001' – X'007FFF'	X_p-extent. See notes 1 and 2.	M	N	N
9–11	UBIN	YPEXTENT	X'000001' – X'007FFF'	Y_p-extent. See notes 1 and 2.	M	N	N
12–13		TEXTFLAGS		Reserved. See note 3.	O	Y	N
14–end of PTD		TXTCONDS		Initial text conditions	O	Y	See note 4.

Notes:

1. The range values shown assume a measurement unit of 1/1440 inch. That is, the measurement base is ten inches, and the X_p- and Y_p-units per unit base are 14,400. If it is necessary to convert to a different measurement unit, please see the conversion routine described in "Interpreting Ranges" on page 45.
2. The default indicator is not allowed for this parameter in this subset.
3. The TEXTFLAGS parameter is reserved. Generators should set this parameter to X'0000', and receivers should ignore it.
4. See the description of the control sequence that specifies the initial text condition.

When the Presentation Text Data Descriptor (PTD) is included in the Active Environment Group (AEG) for a MO:DCA page, the PTOCA object space and object area coincide with the page presentation space, therefore the size of the Presentation Text object space is set equal to the size of the page presentation space. When a Presentation Text object is transformed into an IPDS data stream, the PTD parameters are mapped to IPDS Logical Page Descriptor (LPD) command parameters. Optionally, some PTD parameters may be transformed into control sequences as part of an IPDS Write Text command. When the PTD is specified in the OEG of a MO:DCA text object, the extents of the text object space are defined by the XPEXTENT and YPEXTENT parameters and can be set to any value in the allowed range.

Architecture Note: When the PTD is included in the AEG for a MO:DCA 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.

The coded font information from the MO:DCA Map Coded Font (MCF) and Map Data Resource (MDR) structured fields is used to determine what the Load Font Equivalence command content should be in the IPDS environment. See Appendix B, “IPDS Environment”, on page 161 for more information about the IPDS environment.

Initial text conditions are specified by including the control sequence that contains the parameter to be initialized. It is recommended that exactly one chain be used to specify initial text conditions. The first control sequence that specifies an initial text condition parameter starts with the X'2BD3' prefix and class code and indicates chaining with an odd function type (low-order bit is B'1') if other control sequences follow. All control sequences that follow, except for the last, are chained control sequences that start with their length byte (not with X'2BD3') and have an odd function type. The last control sequence in the chain starts with the length byte but indicates termination of the chain with an even function type (low-order bit is B'0'). For a description of chaining, see “Control Sequence Chaining” on page 34.

Table 20 on page 157 shows which control sequence may be used to specify a particular initial text condition parameter. Control sequences are optional and may appear in any order. If a control sequence appears multiple times, the last occurrence determines the setting of the initial text condition. Control sequences that are not listed in this table are treated as NOPs.

Table 20. PTOCA Initial Text Conditions in a MO:DCA Environment

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

Architecture Note: The Extended Text Color parameter is not supported as an initial text condition in IPDS environments when text is specified as *text-major*, that is, when the PTD for the text is specified in the AEG for the page. This parameter is supported in IPDS environments when the text is specified in a text object with OEG and the PTD is specified in the OEG.

Architecture Note: The following format 1 version of the Presentation Text Data Descriptor is a coexistence MO:DCA structured field and may only be used for migration purposes. The PTD format 1 structured field is not allowed in the OEG of a MO:DCA presentation text object.

Structured Field Introducer				
SF Length	SF Identifier **X'D3A69B'**	Flags	Reserved **X'0000'**	PTD Data

Offset	Type	Name	Range	Meaning	M/O	Def	Ind
0	CODE	XPBASE	X'00'	Unit base for X axis; ten inches	M	N	N
1	CODE	YPBASE	X'00'	Unit base for Y axis; ten inches	M	N	N
2–3	UBIN	XPUNITVL	X'0960' – X'3840'	X_p-units per unit base	M	N	N
4–5	UBIN	YPUNITVL	X'0960' – X'3840'	Y_p-units per unit base	M	N	N
6–7	UBIN	XPEXTENT	X'0001' – X'7FFF'	X_p-extent	M	N	N
8–9	UBIN	YPEXTENT	X'0001' – X'7FFF'	Y_p-extent	M	N	N
10–11	BITS	TEXTFLAGS		Reserved. Must be set to zeros	O	Y	N

Presentation Text Data (PTX)

Structured Field Introducer				
SF Length	SF Identifier **X'D3EE9B'**	Flags	Reserved **X'0000'**	PTD Data

The Presentation Text Data (PTX) structured field has a structured field identifier of X'D3EE9B'. It contains the graphic characters and the control sequences that position the graphic characters.

The contents of the PTX structured field, that is, graphic characters and control sequences, may be included directly into one or more IPDS Write Text (WT) commands by removing the MO:DCA structured field introducer and replacing it with the IPDS WT command syntax. The length information from the PTX structured field must be changed to reflect the correct length in the WT command.

Presentation text data can span multiple PTX structured fields and can be split on any byte boundary. Therefore, a control sequence or chain of control sequences, or a sequence of single- or multi-byte code points can start in one PTX and continue or terminate in a following PTX.

Presentation Exception Conditions

An implementation of the MO:DCA architecture may detect and report PTOCA exception conditions as required to perform or assure the function for which it was designed. The MO:DCA architecture expects a PTOCA processor to handle exception conditions, either by using PTOCA standard actions or by recovering in some other manner.

Presentation Suppression Handling

Suppression of graphic characters is activated by referencing the local ID from the Begin Suppression and End Suppression control sequences with a keyword in the MO:DCA Medium Modification Control (MMC) structured field. The local ID may also be mapped to a global name with a Map Suppression (MSU) structured field. For further information, please see the *Mixed Object Document Content Architecture Reference*, AFPC-0004.

Additional Related Structured Fields

Map Coded Font (MCF): Font information for FOCA-based fonts is provided by the Map Coded Font (MCF) structured field which maps the LID parameter of the Set Coded Font Local control sequence to a FOCA font.

Map Data Resource (MDR): Font information for TrueType/OpenType fonts is provided by the Map Data Resource (MDR) structured field which maps the LID parameter of the Set Coded Font Local control sequence to a TrueType/OpenType font.

For further information about these structured fields, please refer to *Mixed Object Document Content Architecture Reference*, AFPC-0004.

Appendix B. IPDS Environment

The Intelligent Printer Data Stream (IPDS) provides the printer subsystem environment for Presentation Text objects. This appendix describes:

- The context of Presentation Text objects in the IPDS environment; as either text-major text or as independent text objects
- A comparison of PTOCA and IPDS exception conditions
- IPDS commands specific to presentation text

For further information about the IPDS Architecture, refer to *Intelligent Printer Data Stream Reference,* AFPC-0001.

IPDS Presentation Text

Presentation Text objects are transmitted to print devices acting as receivers as part of an IPDS data stream so that a presentation on the print device may be made with visual fidelity. The IPDS Architecture is expressly designed to support all points addressable (APA) printing function that allows text, image, graphics, and bar code to be positioned and presented at any point on the printed page.

Text is described to the IPDS printers in terms of Presentation Text data within Write Text commands. The Presentation Text data is comprised of graphic characters and control sequences in the same format as carried in MO:DCA data streams except that the containing IPDS structure has a different syntax. The printers that receive the Presentation Text object are expected to process the object contents accurately according to the semantic definitions for the supported PTOCA subset.

IPDS Architecture provides the following for the object:

- Command structure and syntax
- Initial conditions for the object
- A means of reporting exception conditions
- A means of resolving fonts and suppression identifiers

IPDS Architecture provides positioning information for the object. The object does not carry the size and shape of the physical medium or information about object positioning. The object does not control what part of its object space is to be presented or what part will fit into the logical page specified by the data stream. The object expresses the exact boundaries that the graphic characters it positions in the object space will require if they are to be presented without error. The logical page may also be used to provide that boundary.

Text data can be positioned anywhere on the logical page in two different ways:

1. All IPDS printers allow text to be placed directly within a logical page using the Write Text command. The logical page can also contain other presentation objects specified with other IPDS commands either before or after a Write Text command. With this method, called “text major”, there is no text object area, and text may be printed anywhere within the valid printable area. For text-major text, the text presentation space is the logical page. Furthermore, object areas for other objects may be positioned with respect to the text.
2. Some IPDS printers support text objects (in addition to the text-major concept). In this case, the Write Text Control command defines a presentation space, object area, and mapping option. The text data is carried within one or more Write Text commands.

Architecture Note: The Extended Text Color parameter is not supported as an initial text condition in IPDS environments for text-major text. This parameter is supported in IPDS environments when the text is specified in a text object.

IPDS Text Command Set

Presentation Text Data Descriptor for Text-Major Text

The following table describes the contents of the PTOCA Presentation Text Data Descriptor and the corresponding parameter location within the IPDS Logical Page Descriptor (LPD) command.

The offset in the table indicates the beginning of the parameter data relative to the beginning of the data portion of the LPD command. This table reflects the descriptor subset syntax for the Presentation Text Data Descriptor.

PTOCA Parameter	IPDS LPD Offset		Notes
UNITBASE	0		
XPUNITVL	2-3		
YPUNITVL	4-5		
XPEXTENT	7-9		5
YPEXTENT	11-13		5
TEXTFLAGS			3
Initial Text Condition Parameters			
Text Orientation: IORNTION BORNTION	 24-25 26-27		 2 2
Initial Inline Coordinate	28-29		5
Initial Baseline Coordinate	30-31		5
Inline Margin	32-33		2
Intercharacter Adjustment: ADJSTMNT DIRCTION	34-35		 2 4
Baseline Increment	38-39		2
Coded Font Local ID	40		2
Text Color: FRGCOLOR	41-42		1,2

Notes:

1. If a receiver cannot provide the color specified by FRGCOLOR, it may substitute values from Table 14 on page 94 defined in the STC control sequence.
2. The default indicator is allowed, meaning use the printer default.
3. The TEXTFLAGS parameter is reserved. This parameter is not used in the IPDS environment.
4. DIRCTION is always defaulted to X'00', that is, the positive direction, so this parameter is not carried in the Logical Page Descriptor (LPD) command.
5. The default indicator is not allowed for this parameter in this subset.

IPDS Presentation Text Data Descriptor for Text Objects

When a text object is used within an IPDS data stream, the PTOCA Presentation Text Data Descriptor is carried within the Write Text Control (WTC) command. This portion of the WTC command is called the Text

Data Descriptor (TDD) and specifies parameters that define the text presentation space size and initial text default conditions. The format of the TDD is as follows:

Offset	Type	Name	Range	Meaning	Required
0–1	UBIN	Length	X'0014' – end of TDD	Length of TDD, including this field	X'0014' – end of TDD
2–3	CODE	SDF ID	X'A69B'	Self-defining-field ID	X'A69B'
4–5			X'0000'	Reserved	X'0000'
6	CODE	Unit base	X'00' X'01' X'02'	Ten inches Ten centimeters Retired item 54	X'00'
7			X'00'	Reserved	X'00'
8–9	UBIN	XUPUB	X'0001' – X'7FFF'	X_t units per unit base	X'3840'
10–11	UBIN	YUPUB	X'0001' – X'7FFF'	Y_t units per unit base	X'3840'
12–14	UBIN	X_t extent	X'000001' – X'007FFF'	X_t extent of the text presentation space in L-units	X'000001' – X'007FFF'
15–17	UBIN	Y_t extent	X'000001' – X'007FFF'	Y_t extent of the text presentation space in L-units	X'000001' – X'007FFF'
18–19	BITS	Text flags	B'00...00'	Reserved for text flags	B'00...00'
20 to end of TDD		Initial text conditions		Initial text conditions, in the same syntax as bytes 14 to end of the MO:DCA PTD found in the table under "Presentation Text Data Descriptor (PTD)" on page 156	

Presentation Text Data

Presentation Text data, which contains the graphic characters and the control sequences that position the graphic characters, is carried in the IPDS Write Text (WT) command. The subsets of PTOCA that are used by IPDS printers are the PT1, PT2, PT3, and PT4 subsets. Notice that each subset contains all of the lower number subsets; for example, PT3 contains all of PT2 (and therefore all of PT1).

The contents of the PTX structured field in a MO:DCA architecture data stream, that is, graphic characters and control sequences, may be included directly into one or more IPDS Write Text (WT) commands. Remove the MO:DCA structured field introducer, replace it with the IPDS WT command syntax, and correct the length information to reflect the length of the WT command.

Presentation Text data can span multiple WT commands. That is, a control sequence or chain of control sequences can be started in the data sent by one WT command and can be completed in the data sent by the WT commands that follow.

A WT command may end in the middle of an embedded control sequence or a double-byte code point. In this event, an exception results if any commands other than Execute Order Anystate, No Operation, Set Home State, or Sense Type and Model are received before the next WT command.

Presentation Exception Conditions

The IPDS Architecture defines its own exception condition codes, called *exception IDs*, which consist of three bytes. PTOCA exception condition codes are mapped to IPDS exception IDs by mapping the two-byte PTOCA code to the last two bytes of the IPDS exception code. In most cases, this mapping is one-to-one. Where it is

not, the IPDS exception ID overrides the PTOCA exception condition code. The IPDS Architecture also defines its own exception responses. In some cases, this exception response is the same as the standard exception action defined by PTOCA. Where it is not, the IPDS exception response overrides the PTOCA standard exception action. Table 21 shows the mapping of PTOCA exception condition codes to IPDS exception IDs.

Table 21. PTOCA Exception Conditions in an IPDS Environment

PTOCA Exception Condition Code	IPDS Exception ID
EC-0001	0200..01
EC-0103	08C1..00 for text major 0201..03 for text objects
EC-0201	0202..01
EC-0401	0204..01
EC-0505	0264..02
EC-0601	0206..01
EC-0605	0260..02
EC-0705	0261..02
EC-0C01	020C..01
EC-0E02	020E..02
EC-0E03	020E..03
EC-0E04	020E..04
EC-0E05	020E..05
EC-0F01	020F..01
EC-1001	0210..01
EC-1101	0211..01
EC-1201	0212..01
EC-1301	0213..01
EC-1401	0214..01
EC-1403	0214..03
EC-1501	0215..01
EC-1601	0216..01
EC-1701	0217..01
EC-1802	0218..02
EC-1901	0219..01
EC-1A01	021A..01
EC-1A03	021A..03
EC-1B01	021B..01
EC-1C01	021C..01
EC-1E01	021E..01
EC-1F01	021F..01
EC-2100	0821..00
EC-3F02	023F..02
EC-5803	0258..03

Table 21 PTOCA Exception Conditions in an IPDS Environment (cont'd.)

PTOCA Exception Condition Code	IPDS Exception ID
EC-6802	0268..02
EC-6902	0269..02
EC-6A02	026A..02
EC-6B02	026B..02
EC-8002	0280..02
EC-8202	0282..02
EC-9801	0298..01
EC-9803	0298..03
EC-9A01	029A..01
EC-9B01	029B..01

Additional Related Commands

The following description is an overview. For further information about these commands, please refer to *Intelligent Printer Data Stream Reference*, AFPC-0001.

Sense Type and Model (STM): Requests information from the printer that identifies the type and model of the device and the command sets supported. The information requested is returned in the Special Data Area of the Acknowledge Reply to the STM command. The command sets and data levels supported, such as PT1, are also returned as part of the acknowledgement data.

XOH Obtain Printer Characteristics (XOH OPC): Requests information from the printer that identifies characteristics of the device. The characteristics include information about the printable area currently available, coded font resolution, and color support.

XOA Request Resource List (XOA RRL): Requests the printer to return a specified list of available resources (coded fonts, overlays, and page segments) in the Acknowledge Reply to this command. This information is provided for use in resource management functions.

Load Font Equivalence (LFE): This command is sent to the printer to map the LID parameter of the Set Coded Font Local control sequence to the coded font's Host Assigned ID to select the font and font attributes to be used by the printer. The coded fonts do not have to exist in the printer when the printer receives this command.

The mapping function provided by this command is independent of the specific font technology employed by the printer. For example, the device may resolve the mapping to stored font patterns downloaded from the host or to permanently resident font patterns.

The coded font resource used for Presentation Text objects may be used by other data objects, such as graphics, as well.

Load Equivalence (LE) and Load Copy Control (LCC): The LE command permits physical values embedded in printer data to be referenced externally using different values, and the LCC command specifies external suppression values to be used in conjunction with the presentation text. These two functions are used to control the suppression function in PTOCA.

Suppression is started by external suppression values specified in the current Load Copy Control (LCC) Command. The external suppression value works with the LID parameter in two ways:

- The external suppression value maps to the LID parameter of the BSU control sequence. For example, if the external value is X'0A', the presentation text identified with the LID X'0A' is suppressed.
- The external suppression value maps to the LID through an equivalence table, using the current Load Equivalence (LE) command. For example, if the external value is X'0B', the LE table may define the following equivalences: X'0B'=X'01', X'0B'=X'02', X'0B'=X'03'. Thus, the external value X'0B' identifies presentation text marked with LID values X'01', X'02', and X'03' for suppression.

  If no external suppression values are started by the LCC command, the BSU and ESU control sequences are processed as No Operation control sequences. Exception conditions defined for BSU or ESU control sequences are detected by the printer whether or not an external suppression value is specified by the LCC command.

During suppression, the printer performs all composition actions, including field checking, and processes all control sequences, such as SCFL, RMI, AMI, DIR, and so on.

The printer can use a single suppression LID for more than one Begin and End Suppression pair if the LIDs are all mapped to the same external value. For example, if LID values of X'06', X'07', and X'09' from BSU, ESU control sequence pairs are mapped to an external value of X'02' from a Load Copy Control (LCC) command by an appropriate LE command received by the printer, the presentation text marked by the three LID values will be suppressed by the single value of X'02'.

BSU, ESU pairs outside an overlay have no effect within the overlay, and BSU, ESU pairs within an overlay have no effect outside the overlay.

Font Commands

The IPDS architecture provides the ability to manage and use a wide variety of font resources. These font resources can be classified into two major categories: coded font components and data-object-font components. Coded fonts and their component parts are defined within the Font Object Content Architecture (FOCA); data-object-font components include the FOCA code page, TrueType/OpenType fonts, and TrueType/OpenType collections. TrueType/OpenType fonts use an outline font technology as do FOCA outline fonts. The FOCA and IPDS architectures also include support for raster fonts and a very simple type of font called a symbol set.

The following IPDS commands are used with TrueType/OpenType fonts and collections:
- Activate Resource (AR)
- Deactivate Data-Object-Font Component (DDOFC)
- Deactivate Font (DF)
- Load Code Page (LCP)
- Load Code Page Control (LCPC)
- Write Object Container Control (WOCC)
- Write Object Container (WOC)
- XOH Erase Residual Font Data (ERFD)
- XOH Erase Residual Print Data (ERPD)

The following IPDS commands are used with FOCA outline fonts:
- Activate Resource (AR)
- Deactivate Font (DF)
- Load Code Page (LCP)
- Load Code Page Control (LCPC)
- Load Font (LF)
- Load Font Character Set Control (LFCSC)
- Load Font Equivalence (LFE)
- XOH Erase Residual Font Data (ERFD)

The following IPDS commands are used with FOCA raster fonts:

- Activate Resource (AR)
- Deactivate Font (DF)
- Load Font (LF)
- Load Font Control (LFC)
- Load Font Equivalence (LFE)
- Load Font Index (LFI)
- XOH Erase Residual Font Data (ERFD)

The following IPDS commands are used with symbol set fonts:

- Activate Resource (AR)
- Deactivate Font (DF)
- Load Font Equivalence (LFE)
- Load Symbol Set (LSS)

Appendix C. PTOCA Retired Functions

Introduction

This appendix:

- Describes retired functions that may occur in a PTOCA object

General

The objective in defining retired functions is twofold: (1) to allow existing applications to run unchanged, and (2) to provide a clear growth direction for future applications.

Retired Functions

Retired functions are control sequences and parameters whose use has been retired except for specific products. Only these specific products may use these functions. All other products should not use these functions, that is, generators should not generate these functions and receivers may ignore them.

Retired Parameters

STC Precision Parameter (Byte 6, Name PRECSION)

The use of this parameter is restricted to pre-year 2000 AFP applications and printers.

The PRECSION parameter is optional on the STC (byte 6), and specifies how the receiver should process colors that are syntactically valid but not supported by the receiver. If PRECSION is X'00', the receiver must support the color selected by FRGCOLOR as specified. If the color is not supported, exception condition EC-5803 exists. The standard action in this case is to use X'FF07'. If PRECSION is X'01', and if the FRGCOLOR value is not supported by the receiver, a default action is allowed. The receiver may use a substitute color or X'FF07'. If the FRGCOLOR value is not syntactically valid, exception condition EC-5803 exists, regardless of the value for the PRECSION parameter. The PRECSION parameter is modal, and X'00' is the default. This parameter supports the default indicator (X'FF'), which means its value is provided by the hierarchy, as follows:

1. Value set by Text Color initial text condition parameter in Descriptor
2. PTOCA default - X'00'

If the value of the PRECSION parameter is not valid, EC-5803 exists. The standard action is to ignore the parameter and continue presentation with the value determined according to the hierarchy.

The PRECSION parameter also defines the hierarchy for determining the presentation process default color attribute value (FRGCOLOR = X'FFFF'). If PRECSION is X'00', the following order applies:

1. Value previously set by Text Color initial text condition parameter in descriptor
2. Data stream specified value
3. Receiver's best possible value

If PRECSION is X'01', the following order applies:

1. Receiver's best possible value
2. Value previously set by Text Color initial text condition parameter in descriptor
3. Data stream specified value

Note that when a Color Mapping Table (CMT) is specified in the MO:DCA environment, the PRECSION parameter is processed for the target color values, not for the original (source) color values.

Notices

The AFP Consortium or consortium member companies might 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.

The following statement does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: AFP Consortium 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 might not apply to you.

This publication 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. The AFP Consortium might make improvements and/or changes in the architecture described in this publication at any time without notice.

Any references in this publication to 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 architecture and use of those Web sites is at your own risk.

The AFP Consortium may use or distribute any information you supply in any way it believes appropriate without incurring any obligation to you.

This information contains examples of data and reports used in daily business operations. To illustrate them in a complete manner, some examples include the names of individuals, companies, brands, or products. These names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Trademarks

These terms are trademarks or registered trademarks of Ricoh Co., Ltd., in the United States, other countries, or both:

ACMA
Advanced Function Presentation
AFP
AFPCC
AFP Color Consortium
AFP Color Management Architecture
Bar Code Object Content Architecture
BCOCA
CMOCA
Color Management Object Content Architecture
InfoPrint
Intelligent Printer Data Stream
IPDS
Mixed Object Document Content Architecture
MO:DCA
Ricoh

The following terms are trademarks of other companies:

Apple and the Apple logo are either registered trademarks or trademarks of Apple Incorporated in the United States and/or other countries.

AFPC and AFP Consortium are trademarks of the AFP Consortium.

International Business Machines Corporation in the United States, other countries, or both: IBM

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

UP^{3}I is a trademark of UP^{3}I Limited.

Other company, product, or service names might be trademarks or service marks of others.

Glossary

Some of the terms and definitions that appear in this glossary have been taken from other source documents.

If you do not find the term that you are looking for, please refer to the *IBM Dictionary of Computing*, document number ZC20-1699 or the *InfoPrint Dictionary of Printing*.

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 move. A method used to designate a new presentation position by specifying the distance from the designated axes to the new presentation position. The reference for locating the new presentation position is a fixed position as opposed to the current presentation position.

absolute positioning. The establishment of a position within a coordinate system as an offset from the coordinate system origin. Contrast with *relative positioning*.

active coded font. The coded font that is currently being used by a product to process text.

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*. Synonymous with *position*.

Advanced Function Presentation (AFP). An open architecture for the management of presentable information that is developed by the AFP Consortium (AFPC). AFP comprises a number of data stream and data object architectures:

- Mixed Object Document Content (MO:DCA) Architecture; formerly called AFPDS
- Intelligent Printer Data Stream (IPDS) Architecture
- AFP Line Data Architecture
- Bar Code Object Content Architecture (BCOCA)
- Color Management Object Content Architecture (CMOCA)
- Font Object Content Architecture (FOCA)
- Graphics Object Content Architecture for AFP (AFP GOCA)
- Image Object Content Architecture (IOCA)
- Metadata Object Content Architecture (MOCA)
- Presentation Text Object Content Architecture (PTOCA)

AFP. See *Advanced Function Presentation*.

AFP Consortium (AFPC). A formal open standards body that develops and maintains AFP architecture. Information about the consortium can be found at `http://www.afpcinc.org`.

AFP data stream. A presentation data stream that is processed in AFP environments. The MO:DCA architecture is the strategic AFP interchange data stream. The IPDS architecture 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* and *AFP data stream*.

AEA. See *alternate exception action*.

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*.

alternate exception action (AEA). In the IPDS architecture, a defined action that a printer can take when a clearly defined, but unsupported, request is received. Control over alternate exception actions is specified by an Execute Order Anystate Exception-Handling Control command.

American National Standards Institute (ANSI). An organization consisting of producers, consumers, and general interest groups. ANSI establishes the procedures by which accredited organizations create and maintain voluntary industry standards in the United States. It is the United States constituent body of the International Organization for Standardization (ISO).

anamorphic scaling. Scaling an object differently in the vertical and horizontal directions. See also *scaling*, *horizontal font size*, and *vertical font size*.

annotation. A process by which additional data or attributes, such as highlighting, are associated with a page

or a position on a page. Application of this data or attributes to the page is typically under the control of the user. Common functions such as applying adhesive removable notes to paper documents or using a transparent highlighter are emulated electronically by the annotation process.

ANSI. See *American National Standards Institute*.

APA. See *all points addressable*.

application. (1) The use to which an information system is put. (2) A collection of software components used to perform specific types of work on a computer.

application program. A program written for or by a user that applies to the user's work.

architected. Identifies data that is defined and controlled by an architecture. Contrast with *unarchitected*.

ascender. The parts of certain lowercase letters, such as *b*, *d*, or *f*, which at zero-degree character rotation rise above the top edge of other lowercase letters such as *a*, *c*, and *e*. Contrast with *descender*.

ascender height. The character shape's most positive character coordinate system Y-axis value.

A-space. The distance from the character reference point to the least positive character coordinate system X-axis value of the character shape. A-space can be positive, zero, or negative. See also *B-space* and *C-space*.

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. See also *character attribute* and *color attribute*.

B

B. See *baseline direction*.

+B. Positive baseline direction.

B_c. See *current baseline presentation coordinate*

B_o. See *baseline presentation origin*

background. The part of a presentation space that is not occupied with object data. Contrast with *foreground*.

background mix. An attribute that determines how the points in overlapping presentation space backgrounds are combined. Contrast with *foreground mix*.

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.

base-and-towers concept. A conceptual illustration of an architecture that shows the architecture as a base with optional tower(s). The base and the towers represent different degrees of function achieved by the architecture.

base support level. Within the base-and-towers concept, the smallest portion of architected function that is allowed to be implemented. This is represented by a base with no towers. Synonymous with *mandatory support level*.

baseline. A conceptual line with respect to which successive characters are aligned. See also *character baseline*. Synonymous with *printing baseline* and *sequential baseline*.

baseline coordinate. One of a pair of values that identify the position of an addressable position with respect to the origin of a specified I,B coordinate system. This value is specified as a distance in addressable positions from the I-axis of an I,B coordinate system. Synonymous with *B-coordinate*.

baseline direction (B). The direction in which successive lines of text appear on a logical page. Synonymous with *baseline progression* and *B-direction*.

baseline extent. A rectangular space oriented around the character baseline and having one dimension parallel to the character baseline. The space is measured along the Y-axis of the character coordinate system. For characters with bounded boxes, the baseline extent at any rotation is its character coordinate system Y-axis extent. Baseline extent varies with character rotation. See also *maximum baseline extent*.

baseline increment. The distance between successive baselines.

baseline offset. The perpendicular distance from the character baseline to the character box edge that is parallel to the baseline and has the more positive character coordinate system Y-axis value. For characters entirely within the negative Y-axis region, the baseline offset can be zero or negative. An example is a subscript character. Baseline offset can vary with character rotation.

baseline presentation origin (B_o). The point on the B axis where the value of the baseline coordinate is zero.

baseline progression (B). The direction in which successive lines of text appear on a logical page. Synonymous with *baseline direction* and *B-direction*.

B-axis. The axis of the I,B coordinate system that extends in the baseline or B-direction. The B-axis does not have to

be parallel to the Y_p-axis of its bounding X_p,Y_p coordinate space.

BCOCA. See *Bar Code Object Content Architecture*.

B-coordinate. One of a pair of values that identify the position of an addressable position with respect to the origin of a specified I,B coordinate system. This value is specified as a distance in addressable positions from the I-axis of an I,B coordinate system. Synonymous with *baseline coordinate*.

B-direction (B). The direction in which successive lines of text appear on a logical page. Synonymous with *baseline direction* and *baseline progression*.

between-the-pels. The concept of pel positioning that establishes the location of a pel's reference point at the edge of the pel nearest to the preceding pel rather than through the center of the pel.

B-extent. The extent in the B-axis direction of an I,B coordinate system. The B-extent must be parallel to one of the axes of the coordinate system that contains the I,B coordinate system. The B-extent is parallel to the Y_p-extent when the B-axis is parallel to the Y_p-axis or to the X_p-extent when the B-axis is parallel to the X_p-axis.

big endian. A format for storage or transmission of binary data in which the most significant bit (or byte) is placed first. Contrast with *little endian*.

BITS. A data type for architecture syntax, indicating one or more bytes to be interpreted as bit string information.

boldface. A heavy-faced type. Printing in a heavy-faced type.

bounded character box. A conceptual rectangular box, with two sides parallel to the character baseline, that circumscribes a character and is just large enough to contain the character, that is, just touching the shape on all four sides.

B-space. The distance between the character coordinate system X-axis values of the two extremities of a character shape.

C

cap-M height. The average height of the uppercase characters in a font. This value is specified by the designer of a font and is usually the height of the uppercase *M*.

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. See also *graphic character* and *control character*.

character angle. The angle that is between the baseline of a character string and the horizontal axis of a presentation space or physical medium.

character attribute. A characteristic that controls the appearance of a character or character string.

character baseline. A conceptual reference line that is coincident with the X axis of the character coordinate system.

character box. A conceptual rectangular box with two sides parallel to the character baseline. A character's shape is formed within a character box by a presentation process, and the character box is then positioned in a presentation space or on a physical medium. The character box can be rotated before it is positioned.

character-box reference edges. The four edges of a character box.

character cell size. The size of a rectangle in a drawing space used to scale font symbols into the drawing space.

character code. An element of a code page or a cell in a code table to which a character can be assigned. The element is associated with a binary value. The assignment of a character to an element of a code page determines the binary value that will be used to represent each occurrence of the character in a character string.

character coordinate system. An orthogonal coordinate system that defines font and character measurement distances. The origin is the character reference point. The X axis coincides with the character baseline.

character identifier. The unique name for a *graphic character*.

character escapement point. The point where the next character reference point is usually positioned. See also *character increment* and *presentation position*.

character identifier. The unique name for a graphic character.

character increment. The distance from a character reference point to a character escapement point. For each character, the increment is the sum of a character's A-space, B-space, and C-space. A character's character increment is the distance the inline coordinate is incremented when that character is placed in a presentation space or on a physical medium. Character increment is a property of each graphic character in a font and of the font's character rotation.

character increment adjustment. In a scaled font, an adjustment to character increment values. The adjustment value is derived from the kerning track values for the font used to present the characters.

character metrics. Measurement information that defines individual character values such as height, width, and space. Character 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. Often included as part of the more general term “font metrics”. See also *character set metrics* and *font metrics*.

character pattern. The scan pattern for a graphic character of a particular size, style, and weight.

character-pattern descriptor. Information that the printer needs to separate font raster patterns. Each character pattern descriptor is eight bytes long and specifies both the character box size and an offset value; the offset value permits the printer to find the beginning of the character raster pattern within the character raster pattern data for the complete font.

character origin. The point within the graphic pattern which is to be aligned with the presentation position. See also *character reference point*.

character positioning. A method used to determine where a character is to appear in a presentation space or on a physical medium.

character precision. The acceptable amount of variation in the appearance of a character on a physical medium from a specified ideal appearance, including no acceptable variation. Examples of appearance characteristics that can vary for a character are shape and position.

character reference point. The origin of a character coordinate system. The X axis is the character baseline.

character rotation. The alignment of a character with respect to its character baseline, measured in degrees in a clockwise direction. Examples are 0°, 90°, 180°, and 270°. Zero-degree character rotation exists when a character is in its customary alignment with the baseline. Contrast with *rotation*.

character set. A finite set of different graphic or control characters that is complete for a given purpose. For example, the character set in ISO Standard 646, *7-bit Coded Character Set for Information Processing Interchange*.

character set attribute. An attribute used to specify a font.

character set metrics. The measurements used in a font. Examples are height, width, and character increment for each character of the font. See also *character metrics* and *font metrics*.

character shape. The visual representation of a graphic character.

character shape presentation. A method used to form a character shape on a physical medium at an addressable position.

character shear. The angle of slant of a character cell that is not perpendicular to a baseline. Synonymous with *shear*.

character string. A sequence of characters.

CJK fonts. Fonts that contain a set of unified ideographic characters used in the written Chinese, Japanese, and Korean languages. The character encoding is the same for each language, but there might be glyph variants between languages.

clipping. Eliminating those parts of a picture that are outside of a clipping boundary such as a presentation space. Synonymous with *trimming*.

CMOCA. See *Color Management Object Content Architecture*.

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 consisting of an encoding scheme identifier, character set identifiers, code page identifiers, and other relevant information that uniquely identifies the coded graphic character representation used.

coded font. (1) 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*. (2) In FOCA, a resource containing the resource names of a valid pair of font character set and code page resources. The graphic character set of the font character set must match the graphic character set of the code page for the coded font resource pair to be valid. (3) In the IPDS architecture, a raster font resource containing code points directly paired to font metrics and the raster representation of character shapes, for a specific graphic character set. (4) In the IPDS architecture, a font resource containing descriptive information, a code page, font metrics, and a digital-technology representation of character shapes for a specific graphic character set.

coded font local identifier. A binary identifier that is mapped by the environment to a named resource to identify a coded font. See also *local identifier*.

coded graphic character. A graphic character that has been assigned one or more code points within a code page.

coded graphic character set. A set of graphic characters with their assigned code points.

Coded Graphic Character Set Global Identifier (CGCSGID). A four-byte binary or a ten-digit decimal identifier consisting of the concatenation of a GCSGID and a CPGID. The CGCSGID identifies the code point assignments in the code page for a specific graphic character set, from among all the graphic characters that are assigned in the code page.

code page. (1) A resource object containing descriptive information, graphic character identifiers, and code points corresponding to a coded graphic character set. 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 *code page*, *code table* and *section*.

code table. A table showing the character allocated to each code point in a code. See also *code page* and *code point*.

color. A visual attribute of things that results from the light they emit, transmit, or reflect.

color attribute. An attribute that affects the color values provided in a graphics primitive, a text control sequence, or an IPDS command. Examples of color attributes are foreground color and background color.

Color Management Object Content Architecture (CMOCA). An architected collection of constructs used for the interchange and presentation of the color management information required to render a print file, document, group of pages or sheets, page, overlay, or data object with color fidelity.

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 called *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 space*.

command. (1) In the IPDS architecture, a structured field sent from a host to a printer. (2) A request for system action.

complex text layout. The typesetting of writing systems that require complex transformations between text input and text display for proper rendering on the screen or the printed page.

compression algorithm. An algorithm used to compress image data. Compression of image data can decrease the volume of data required to represent an image.

construct. An architected set of data such as a structured field or a triplet.

continuous-form media. Connected sheets. An example of connected sheets is sheets of paper connected by a perforated tear strip. Contrast with *cut-sheet media*.

control character. A character that denotes the start, modification, or end of a control function. A control character can be recorded for use in a subsequent action, and it can have a graphic representation. See also *character*.

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.

control sequence chaining. A method used to identify a sequential string of control sequences so they can be processed efficiently.

control sequence class. An assigned coded character that identifies a control sequence's syntax and how that syntax is to be interpreted. An example of a control sequence class is X'D3', which identifies presentation text object control sequences.

control sequence function type. The coded character occupying the fourth byte of an unchained control sequence introducer. This code defines the function whose semantics can be prescribed by succeeding control sequence parameters.

control sequence introducer. The information at the beginning of a control sequence. An unchained control sequence introducer consists of a control sequence prefix, a class, a length, and a function type. A chained control sequence introducer consists of a length and a function type.

control sequence length. The number of bytes used to encode a control sequence excluding the control sequence prefix and class.

control sequence prefix. The escape character used to identify a control sequence. The control sequence prefix is the first byte of a control sequence. An example of a control sequence prefix is X'2B'.

controlled white space. White space caused by execution of a control sequence. See also *white space*.

controlling environment. The environment in which an object is embedded, for example, the IPDS and MO:DCA data streams.

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. See also *character coordinate system*.

coordinates. A pair of values that specify a position in a coordinate space. See also *absolute coordinate* and *relative coordinate*.

CPGID. See *Code Page Global Identifier*.

C-space. The distance from the most positive character coordinate system X-axis value of a character shape to the character's escapement point. C-space can be positive, zero, or negative. See also *A-space* and *B-space*.

current baseline coordinate. The baseline presentation position at the present time. The baseline presentation position is the summation of the increments of all baseline controls since the baseline was established in the presentation space. The baseline presentation position is established in a presentation space either as part of the initialization procedures for processing an object or by an Absolute Move Baseline control sequence. Synonymous with *current baseline presentation coordinate*.

current baseline presentation coordinate (B_c).. Synonymous with *current baseline coordinate*

current inline coordinate. The inline presentation position at the present time. This inline presentation position is the summation of the increments of all inline controls since the inline coordinate was established in the presentation space. An inline presentation position is established in a presentation space either as part of the initialization procedures for processing an object or by an Absolute Move Inline control sequence. Synonymous with *current inline presentation coordinate*.

current inline presentation coordinate (I_c). Synonymous with *current inline coordinate*.

current position. The position identified by the current presentation space coordinates. For example, the coordinate position reached after the execution of a drawing order. See also *current baseline presentation coordinate* and *current inline presentation coordinate*. Contrast with *given position*.

cut-sheet media. Unconnected sheets. Contrast with *continuous-form media*.

D

data element. A unit of data that is considered indivisible.

data mask. A sequence of bits that can be used to identify boundary alignment bits in image data.

data-object font. (1) In the IPDS architecture, a complete-font resource that is a combination of font components at a particular size, character rotation, and encoding. A data-object font can be used in a manner analogous to a coded font. The following useful combinations can be activated into a data-object font:

- A TrueType/OpenType font, an optional code page, and optional linked TrueType/OpenType objects; activated at a particular size, character rotation, and encoding
- A TrueType/OpenType collection, either an index value or a full font name to identify the desired font within the collection, an optional code page, and optional linked TrueType/OpenType objects; activated at a particular size, character rotation, and encoding

See also data-object-font component. (2) 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-object-font component. In the IPDS architecture, a font resource that is either printer resident or is downloaded using object container commands. Data-object-font components are used as components of a data-object font. Examples of data-object-font components include TrueType/OpenType fonts and TrueType/OpenType collections. See also data-object font.

data stream. A continuous stream of data that has a defined format. An example of a defined format is a structured field.

DBCS. See *double-byte character set*.

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 can be specified by an external parameter. See also *external parameter*.

descender. The part of the character that extends into the character coordinate system negative Y-axis region. Examples of letters with descenders at zero-degree character rotation are *g*, *j*, *p*, *q*, *y*, and *Q*. Contrast with *ascender*.

descender depth. The character shape's most negative character coordinate system Y-axis value.

design metrics. A set of quantitative values, recommended by a font designer, to describe the characters in a font.

design size. The size of the unit em for a font. All relative font measurement values are expressed as a proportion of the design size. For example, the width of the letter “I” can be specified as one fourth of the design size.

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. See also *system-level font resource*.

device level font resource. A device-specific font object from which a presentation device can obtain the font information required to present character images.

digital half-toning. A method used to simulate gray levels on a bilevel device.

digital image. An image whose image data was sampled at regular intervals to produce a digital representation of the image. The digital representation is usually restricted to a specified set of values.

document. (1) A machine-readable collection of one or more objects that represents a composition, a work, or a collection of data. (2) A publication or other written material.

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 the document component. See also *document component* and *structured field*.

document editing. A method used to create or modify a document.

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 fidelity. The degree to which a document presentation preserves the creator’s intent.

document formatting. A method used to determine where information is positioned in presentation spaces or on physical media.

document presentation. A method used to produce a visible copy of formatted information on physical media.

double-byte character set (DBCS). A character set that can contain up to 65,536 characters.

double-byte coded font. A coded font in which the code points are two bytes long.

draw rule. A method used to construct a line, called a rule, between two specified presentation positions. The line that is constructed is either parallel to the inline I axis or baseline B axis.

duplex. A method used to print 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. In printing, a unit of linear measure referring to the baseline-to-baseline distance of a font, in the absence of any external leading.

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*.

escape sequence. A string of bit combinations that is used for control in code extension procedures. The first of these bit combinations represents the control function Escape.

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. See also *character direction* and *inline direction*.

ESID. See *Encoding Scheme Identifier*.

established baseline coordinate. The current baseline presentation coordinate when no temporary baseline exists or the last current baseline presentation coordinate that existed before the first active temporary baseline was created. If temporary baselines are created, the current baseline presentation coordinate coincides with the presentation coordinate of the most recently created temporary baseline.

exception. One of the following:
1. An invalid or unsupported data-stream construct.
2. In the IPDS architecture, a condition requiring host notification.
3. In the IPDS architecture, a condition that requires the host to resend data.

exception action. Action taken when an exception is detected.

exception condition. The condition that exists when a product finds 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*.

expanded. A type width that widens all characters of a typeface.

Extended Binary-Coded Decimal Interchange Code (EBCDIC). A coded character set that consists of eight-bit coded characters.

external leading. The amount of white space, in addition to the internal leading, that can be added to interline spacing without degrading the aesthetic appearance of a font. This value is usually specified by a font designer. Contrast with *internal leading*.

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. Contrast with *internal parameter*.

F

FGID. See *Font Global Identifier*.

final form data. Data that has been formatted for presentation.

fixed metrics. Graphic character measurements in physical units such as pels, inches, or centimeters.

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 baseline extent. In the IPDS architecture, the sum of the uniform or maximum baseline offset and the maximum baseline descender of all characters in the font.

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 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 inline sequence. The clockwise rotation of the inline direction relative to a character pattern.

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. See also *character metrics* and *character set metrics*.

font modification parameters. Parameters that alter the appearance of a typeface.

font object. A resource object that 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 production. A method used to create a font. This method includes designing each character image, converting the character images to a digital-technology format, defining parameter values for each character, assigning appropriate descriptive and identifying information, and creating a font resource that contains the required information in a format that can be used by a text processing system. Digital-technology formats include bit image, vector drawing orders, and outline algorithms. Parameter values include such attributes as height, width, and escapement.

font referencing. A method used to identify or characterize a font. Examples of processes that use font referencing are document editing, formatting, and presentation.

font width (FW). Synonymous with *horizontal font size*.

foreground. The part of a presentation space that is occupied with object data.

foreground color. An attribute used to specify the color of the foreground of a primitive. Contrast with *background color*.

foreground mix. A method used to determine how the foreground color of data is combined with the existing color of a graphics presentation space. An example of data is a graphics primitive. Contrast with *background mix*.

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.

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*.

GCUID. See *Graphic Character UCS Identifier*.

GID. See *global identifier*.

given position. The coordinate position at which drawing is to begin. A given position is specified in a drawing order. Contrast with *current position*.

GLC chain. The set of glyph layout control sequences used to present a set of glyphs. It consists of a GLC control sequence followed by one or more GIR/GAR/GOR control sequence groupings, wherein the GOR is always optional. These control sequences must be chained together using PTOCA chaining rules. No other control sequences can be interspersed within the GIR/GAR/GOR groupings or between the groupings. The GLC chain may be terminated by an optional UCT control sequence that carries the code points of the glyphs rendered by the GLC chain.

Global Identifier (GID). One of the following:

- Code Page Global ID (CPGID)
- Graphic Character Global Identifier (GCGID)
- *Graphic Character UCS Identifier (GCUID)*
- Font Global Identifier (FGID)
- Graphic Character Set Global Identifier (GCSGID)
- In the MO:DCA architecture, an encoded graphic character string which, when qualified by the associated CGCSGID, provides a reference name for a document element.

global resource identifier (GRID). An eight-byte identifier that identifies a 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 1,440ths of an inch

glyph. (1) A member of a set of symbols that represent data. Glyphs can be letters, digits, punctuation marks, or other symbols. Synonymous with *graphic character*. See also *character*. (2) In typography, a glyph is a particular graphical representation of a grapheme, or sometimes several graphemes in combination (a composed glyph), or only a part of a grapheme. In computing as well as typography, the term character refers to a grapheme or grapheme-like unit of text, as found in natural language writing systems (scripts). A character or grapheme is a unit of text, whereas a glyph is a graphical unit. TrueType/OpenType fonts describe glyphs as a set of paths.

glyph advance. A glyph advance is the absolute displacement of a glyph's origin on the baseline in the inline direction from a specific point. In the context of complex text rendering using GLC chains, the specific point is the current text position at the beginning of the GLC chain.

glyph ID. A glyph ID is an index to a table entry in a TrueType/OpenType font that allows an application to retrieve the glyph's shape data.

glyph offset. A glyph offset is the offset of the glyph's origin from the current baseline in the baseline direction. In the context of complex text rendering using GLC chains, the current baseline is the baseline defined at the beginning of the GLC chain.

GOCA. See *Graphics Object Content Architecture*.

grapheme. (1) A minimally distinctive unit of writing in the context of a particular writing system. For example, å ("a + Combining Ring Above" or "Latin Small Letter A with Ring Above") is a grapheme in the Danish writing system. (2) What an end-user might think of as a character. (3) In typography, a grapheme is the fundamental unit in written language. Graphemes include alphabetic letters, Chinese characters, numerals, punctuation marks, and all the individual symbols of any of

the world's writing systems. In a typeface each character typically corresponds to a single glyph, but there are exceptions, such as a font used for a language with a large alphabet or complex writing system, where one character may correspond to several glyphs, or several characters to one glyph.

graphic character. A member of a set of symbols that represent data. Graphic characters can be letters, digits, punctuation marks, or other symbols. Synonymous with *glyph*. See also *character*.

Graphic Character Global Identifier (GCGID). An alphanumeric character string used to identify a specific graphic character. A GCGID can be from four bytes to eight bytes long.

graphic character identifier. The unique name for a graphic character in a font or in a graphic character set. See also *character identifier*.

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.

Graphic Character UCS Identifier (GCUID). An alphanumeric character string used to identify a specific graphic character. The GCUID naming scheme is used for additional characters and sets of characters that exist in UNICODE; each GCUID begins with the letter *U* and ends with a UNICODE code point. The Unicode Standard is fully compatible with the earlier Universal Character Set (UCS) Standard.

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.

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*.

horizontal font size. (1) A characteristic value, parallel to the character baseline, that represents the size of all graphic characters in a font. Synonymous with *font width*. (2) In a font character set, nominal horizontal font size is a font-designer defined value corresponding to the nominal character increment for a font character set. The value is generally the width of the space character, and is defined differently for fonts with different spacing characteristics.

1. For fixed-pitch, uniform character increment fonts: the fixed character increment, that is also the space character increment.
2. For PSM fonts: the width of the space character.
3. For typographic, proportionally-spaced fonts: one third of the vertical font size, that 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, character shapes and character metrics might need to be scaled prior to presentation.

I

I. See *inline direction*.

+I. Positive inline direction.

I_c. See *current inline presentation coordinate*

I_o. See *inline presentation origin*

I-axis. The axis of an I,B coordinate system that extends in the inline direction. The I-axis does not have to be parallel to the X_p-axis of its bounding X_p,Y_p coordinate space.

I,B coordinate system. The coordinate system used to present graphic characters. This coordinate system is used to establish the inline and baseline directions for the placement of successive graphic characters within a presentation space. See also *X_p,Y_p coordinate system*.

ID. Identifier.

IDE. See *image data element*.

IEEE. Institute of Electrical and Electronics Engineers.

I-direction. The direction in which successive characters appear in a line of text. Synonymous with *inline direction*.

I-extent. The X_p-extent when the I-axis is parallel to the X_p-axis or Y_p-extent when the I-axis is parallel to the Y_p-axis. The definition of the I-extent depends on the X_p- or Y_p-extent because the I,B coordinate system is contained within a X_p,Y_p coordinate system.

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 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 object. An object that contains image data. See also *object*.

Image Object Content Architecture (IOCA). An architected collection of constructs used to interchange and present images.

initial addressable position. The values assigned to I_c and B_c by the data stream at the start of object state. The standard action values are I_o and B_o.

inline-baseline coordinate system. See *I,B coordinate system*.

inline coordinate. The first of a pair of values that identifies the position of an addressable position with respect to the origin of a specified I,B coordinate system. This value is specified as a distance in addressable positions from the B-axis of an I,B coordinate system.

inline direction (I). The direction in which successive characters appear in a line of text. Synonymous with *I-direction*.

inline margin. The inline coordinate that identifies the initial addressable position for a line of text.

inline presentation origin (I_o). The point on the I-axis where the value of the inline coordinate is zero.

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*.

intercharacter adjustment. Additional distance applied to a character increment that increases or decreases the distance between presentation positions, effectively modifying the amount of white space between graphic characters. The amount of white space between graphic characters is changed to spread the characters of a word for emphasis, distribute excess white space on a line among the words of that line to achieve right justification, or move the characters on the line closer together as in kerning. Examples of intercharacter adjustment are intercharacter increment and intercharacter decrement.

intercharacter decrement. Intercharacter adjustment applied in the negative I-direction from the current presentation position. See also *intercharacter adjustment*.

intercharacter increment. Intercharacter adjustment applied in the positive I-direction from the current presentation position. See also *intercharacter adjustment*.

internal leading. A font design parameter referring to the space provided between lines of type to keep ascenders separated from descenders and to provide an aesthetically pleasing interline spacing. The value of this parameter usually equals the difference between the vertical font size and the font baseline extent. Contrast with *external leading*.

internal parameter. In PTOCA, a parameter whose current value is contained within the object. Contrast with *external parameter*.

interoperability. The capability to communicate, execute programs, or transfer data among various functional units in a way that requires the user to have little or no knowledge of the unique characteristics of those units.

IOCA. See *Image Object Content Architecture*.

IPDS. See *Intelligent Printer Data Stream*.

italics. A typeface with characters that slant upward to the right. In FOCA, italics is the common name for the defined inclined typeface posture attribute or parameter.

K

Kanji. A graphic character set for symbols used in Japanese ideographic alphabets.

kerning. The design of graphic characters so that their character boxes overlap, resulting in the reduction of space between characters. This allows characters to be designed for cursive languages, ligatures, and proportionally-spaced fonts. An example of kerning is the printing of adjacent graphic characters so they overlap on the left or right side.

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*.

leading. A printer's term for the amount of space between lines of a printed page. Leading refers to the lead slug

placed between lines of type in traditional typesetting. See also *internal leading* and *external leading*.

leading edge. (1) The edge of a character box that in the inline direction precedes the graphic character. (2) The front edge of a sheet as it moves through a printer.

legibility. Characteristics of presented characters that affect how rapidly, easily, and accurately one character can be distinguished from another. The greater the speed, ease, and accuracy of perception, the more legible the presented characters. Examples of characteristics that affect legibility are shape, spacing, and composition.

LID. See *local identifier*.

ligature. A single glyph representing two or more characters. Examples of characters that can be presented as ligatures are *ff* and *ffi*.

little endian. A bit or byte ordering where the right-most 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.

logical page. A presentation space. One or more object areas or data blocks can be mapped to a logical page. A logical page has specifiable characteristics, such as size, shape, orientation, and offset. The shape of a 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 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 = 14,400)
- 1 logical unit = 1/240 inch
  (unit base = 10 inches,
  units per unit base = 2,400)

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*.

lowercase. Pertaining to small letters as distinguished from capital letters. Examples of small letters are: *a*, *b*, and *g*. Contrast with *uppercase*.

L-unit. Synonymous with *logical unit*.

LUT. See *look-up table*.

M

magnetic ink character recognition (MICR). Recognition of characters printed with ink that contains particles of a magnetic material.

mandatory support level. Within the base-and-towers concept, the smallest portion of architected function that is allowed to be implemented. This is represented by a base with no towers. Synonymous with *base support level*.

maximum ascender height. The maximum of the individual character ascender heights. A value for maximum ascender height is specified for each supported rotation of a character. Contrast with *maximum descender depth*.

maximum baseline extent. In FOCA, the sum of the maximum of the individual character baseline offsets and the maximum of the individual character descender depths, for a given font.

maximum descender depth. The maximum of the individual character descender depths. A value for maximum descender depth is specified for each supported rotation of a character. Contrast with *maximum ascender height*.

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 statement of function.

measurement base. A base unit of measure from which other units of measure are derived.

media. Plural of medium. See also *medium*.

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 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*.

Metadata Object Content Architecture (MOCA). A resource object architecture to carry metadata that serves to provide context or additional information about an AFP object or other AFP data.

MICR. See *magnetic ink character recognition*.

mil. 1/1000 inch.

mix. A method used to determine how the color of a graphics primitive is combined with the existing color of a graphics presentation space. See also *foreground mix* and *background mix*.

Mixed Object Document Content Architecture (MO:DCA). An architected, device-independent data stream for interchanging documents.

MO:DCA. See *Mixed Object Document Content Architecture*.

MO:DCA-L. The subset of the MO:DCA architecture that defines an interchange format for resource documents.

MO:DCA-P. The subset of the MO:DCA architecture that defines presentation documents. This term is now synonymous with the term MO:DCA.

MOCA. See *Metadata Object Content Architecture*.

monospaced 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. Synonymous with *uniformly spaced font*. Contrast with *proportionally spaced font* and *typographic font*.

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”.

nesting coordinate space. A coordinate space that contains another coordinate space. Examples of coordinate spaces are medium, overlay, page and object area.

neutral white. A color attribute that gives a device-dependent default color, typically white on a screen, and black on a printer.

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*.

normal-duplex printing. Duplex printing that simulates the effect of physically turning the sheet around the Y_m-axis.

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, that can be used to reference the object. Examples of objects are text, graphics, and image data objects.

object area. In the MO:DCA architecture, 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 bundled together. Examples of object data include graphic characters, image data elements, and drawing orders.

offset. A table heading for architecture syntax. The entries under this heading indicate the numeric displacement into a construct. The offset is measured in bytes and starts with byte zero. Individual bits can be expressed as displacements within bytes.

orientation. The angular distance a presentation space or data block 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. See also *presentation space orientation* and *text orientation*.

origin. The point in a coordinate system where the axes intersect. Examples of origins are the addressable position in an X_m,Y_m coordinate system where both coordinate values are zero and the character reference point in a character coordinate system.

orthogonal. Intersecting at right angles. An example of orthogonal is the positional relationship between the axes of a Cartesian coordinate system.

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*.

overscore. A line parallel to the baseline and placed above the character.

overstrike. In PTOCA, the presentation of a designated character as a string of characters in a specified text field. The intended effect is to make the resulting presentation appear as though the text field, whether filled with

characters or blank, has been marked out with the overstriking character.

overstriking. The method used to merge two or more graphic characters at the same addressable position in a presentation space or on a physical medium.

P

page. (1) A data stream object delimited by a Begin Page structured field and an End Page structured field. A page can contain text, image, graphics, and bar code data. In the IPDS architecture, a page can be copied a specified number of times with or without modification. (2) The final representation of such an object on a physical medium.

page segment. (1) In the IPDS architecture, a resource object that can contain text, image, graphics, and bar code data. Page segments do not define their own environment, but are processed in the existing environment. (2) 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. (3) 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.

pel. The smallest printable or displayable unit on a physical medium. In computer graphics, the smallest element of a physical medium that can be independently assigned color and intensity. Pels per inch is often used as a measurement of presentation granularity. Synonymous with *picture element* and *pixel*.

physical medium. A physical entity on which information is presented. Examples of a physical medium are a sheet of paper and a display screen. See also *medium* and *medium presentation space*.

picture element. Synonymous with *pel*.

pixel. Synonymous with *pel*.

point. A unit of measure used mainly for measuring typographical material. There are seventy-two points to an inch.

portrait. A presentation orientation in which the X_m axis is parallel to the short sides of a rectangular physical medium. Contrast with *landscape*.

position. A position in a presentation space or on a physical medium that can be identified by a coordinate from the coordinate system of the presentation space or physical medium. See also *picture element*. Synonymous with *addressable position*.

posture. Inclination of a letter with respect to a vertical axis. Examples of inclination are upright and inclined. An example of upright is Roman. An example of inclined is italics.

pragmatics. Information related to the usage of a construct.

presentation device. A device that produces character shapes, graphics pictures, images, or bar code symbols on a physical medium. Examples of a physical medium are a display screen and a sheet of paper.

presentation position. An addressable position that is coincident with a character reference point. See also *addressable position* and *character reference point*.

presentation services. In printing, a software component that communicates with a printer using a printer data stream, such as the IPDS data stream, to print pages, download and manage print resources, and handle exceptions.

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 presentation spaces are medium, logical page, and object area. See also *text presentation space*.

presentation space orientation. The number of degrees and minutes a presentation space is rotated in a specified coordinate system. For example, the orientation of printing on a physical medium, relative to the X_m axis of the X_m,Y_m coordinate system. See also *orientation* and *text orientation*.

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 direction. In FOCA, the direction in which successive characters appear in a line of text.

printing baseline. Synonymous with *baseline*.

process color. A color that is specified as a combination of the components, or primaries, of a color space. A process color is rendered by mixing the specified amounts of the primaries. An example of a process color is C=.1, M=.8, Y=.2, K=.1 in the cyan/magenta/yellow/black (CMYK) color space. Contrast with *spot color*.

proportion. Relationship of the width of a letter to its height.

proportional spacing. The spacing of characters in a printed line so that each character is allotted a space based on the character's width.

proportionally spaced 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*. Synonymous with *typographic font*. Contrast with *monospaced font* and *uniformly spaced font*.

Proportional Spacing Machine font (PSM font). A font originating with the electric typewriter and having character increment values in proportion to the relative character shape width.

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 can be expressed in binary, decimal, or hexadecimal. The range can consist of a single value.

readability. The characteristics of visual material that determine the degree of comfort with which it can be read over a sustained period of time. Examples of characteristics that influence readability are type quality, spacing, and composition.

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 line. A straight line developed from a specified point by a given displacement.

relative metrics. Graphic character measurements expressed as fractions of a square, called the *em-square*, whose sides correspond to the vertical size of the font. Because the measurements are relative to the size of the em-square, the same metrics can be used for different point sizes and different raster pattern resolutions. Relative metrics require defining the unit of measure for the Em-square, the point size of the font, and, if applicable, the resolution of the raster pattern.

relative move. A method used to establish a new current position. Distance and direction from the current position are used to establish the new current position. The direction of displacement is inline along the I axis in the I-direction, or baseline along the B axis in the B-direction, or both.

relative positioning. The establishment of a position within a coordinate system as an offset from the current position. Contrast with *absolute positioning*.

repeat string. A method used to repeat the character content of text data until a given number of characters has been processed. Any control sequences in the text data are ignored. This method provides the functional equivalence of a Transparent Data control sequence when the given number of repeated characters is equal to the number of characters in the text data.

repeating group. A group of parameter specifications that can 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 can 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.

Roman. Relating to a type style with upright letters.

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*.

rule. A solid line of any line width.

S

sans serif. A type style characterized by strokes that end with no flaring or crossing of lines at the stroke-ends. Contrast with *serif*.

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 two's-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. See also *anamorphic scaling*.

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. 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.

sequential baseline. Synonymous with *baseline*.

sequential baseline position. The current addressable position for a baseline in a presentation space or on a physical medium. See also *baseline coordinate* and *current baseline presentation coordinate*.

serif. A short line angling from or crossing the free end of a stroke. Examples are horizontal lines at the tops and bottoms of vertical strokes on capital letters, for example, *I* and *H*, and the decorative strokes at the ends of the horizontal members of a capital *E*. Contrast with *sans serif*.

shape technology. A method used to encode character shapes digitally using a specified algorithm.

shear. The angle of slant of a character cell that is not perpendicular to a baseline. Synonymous with *character shear*.

shearline direction. In GOCA, the direction specified by the character shear and character angle attributes.

sheet. In the IPDS architecture, a physical entity on which information is printed. An example of a sheet is one piece of paper.

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. 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.

slope. The posture, or incline, of the main strokes in the graphic characters of a font. Slope is specified in degrees by a font designer.

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*.

standard action. The architecture-defined action to be taken on detecting an exception condition, when the environment specifies that processing should continue.

stroke. A straight or curved line used to create the shape of a letter.

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*.

subset. Within the base-and-towers concept, a portion of architecture represented by a particular level in a tower or by a base. See also *subsetting tower*.

subsetting tower. Within the base-and-towers concept, a tower representing an aspect of function achieved by an architecture. A tower is independent of any other towers. A tower can be subdivided into subsets. A subset contains all the function of any subsets below it in the tower. See also *subset*.

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.

surrogate pair. A sequence of two *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 coded 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*.

system-level font resource. A common-source font from which:

- Document-processing applications can obtain resolution-independent formatting information.
- Device-service applications can obtain device-specific presentation information.

T

temporary baseline. The shifted baseline used for subscript and superscript.

temporary baseline coordinate. The B-value of the I,B coordinate pair of an addressable position on the temporary baseline.

temporary baseline increment. A positive or negative value that is added to the current baseline presentation coordinate to specify the position of a temporary baseline in a presentation space or on a physical medium. Several increments might have been used to place a temporary baseline at the current baseline presentation coordinate.

text. A graphic representation of information. Text can consist of alphanumeric characters and symbols arranged in paragraphs, tables, columns, and other shapes.

text major. A description for text where the Presentation Text Data Descriptor (PTD) is specified in page controls. In MO:DCA, the PTD is in the Active Environment Group (AEG) for the page; in IPDS, the PTD is specified as initial text-major conditions in the Logical Page Descriptor command.

text object. (1) An object that contains text data. (2) A device-independent, self-defining representation of a two-dimensional presentation space, called the text object space, which contains presentation text data.

text object space. Synonymous with *text presentation space*.

text orientation. A description of the appearance of text as a combination of inline direction and baseline direction. See also *baseline direction*, *inline direction*, *orientation*, and *presentation space orientation*.

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. Synonymous with *text object space*.

throughscore. A line parallel to the baseline and placed through the character.

toned. Containing marking agents such as toner or ink. Contrast with *untoned*.

translating. In computer graphics, moving all or part of a picture in the presentation space from one location to another without rotating.

transparent data. A method used to indicate that any control sequences occurring in a specified portion of data can be ignored.

trimming. Eliminating those parts of a picture that are outside of a clipping boundary such as a presentation space. Synonymous with *clipping*.

triplet. A three-part self-defining variable-length parameter consisting of a length byte, an identifier byte, and parameter-value bytes.

triplet identifier. A one-byte type identifier for a triplet.

truncation. Planned or unplanned end of a presentation space or data presentation. This can occur when the presentation space extends beyond one or more boundaries of its containing presentation space or when there is more data than can be contained in the presentation space.

tumble-duplex printing. Duplex printing that simulates the effect of physically turning a sheet around the X_m axis.

type. A table heading for architecture syntax. The entries under this heading indicate the types of data present in a construct. Examples include: BITS, CHAR, CODE, SBIN, UBIN, UNDF.

typeface. All characters of a single type family or style, weight class, width class, and posture, regardless of size. For example, Sonoran Serif, Bold, Normal, Italics, in any point size.

type family. All characters of a single design, regardless of attributes such as width, weight, posture, and size. Examples are Courier and Gothic.

type structure. Attributes of characters other than type family or typeface. Examples are solid shape, hollow shape, and overstruck.

type style. The form of characters within the same font, for example, Courier or Gothic.

type weight. A parameter indicating the degree of boldness of a typeface. A character's stroke thickness determines its type weight. 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*. Synonymous with *proportionally spaced font*. Contrast with *monospaced font* and *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*.

unbounded character box. A character box that can have blank space on any sides of the character shape.

underscore. A method used to create an underline beneath the characters in a specified text field. An example of underscore is the line presented under one or more characters. Also a special graphic character used to implement the underscoring function.

UNDF. A data type for architecture syntax, indicating one or more bytes that are undefined by the architecture.

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. Surrogates are pairs of Unicode code points that allow for the encoding of as many as 1 million additional characters without any use of escape codes.

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).

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. Synonymous with *monospaced font*. Contrast with *proportionally spaced font* and *typographic font*.

unit base. A one-byte code that represents the length of the measurement base. For example, X'00' might specify that the measurement base is ten inches.

untoned. Unmarked portion of a physical medium. Contrast with *toned*.

uppercase. Pertaining to capital letters. Examples of capital letters are *A*, *B*, and *C*. Contrast with *lowercase*.

V

variable space. A method used to assign a character increment dimension of varying size to space characters. The space characters are used to distribute white space within a text line. The white space is distributed by expanding or contracting the dimension of the variable space character's increment dependent upon the amount

of white space to be distributed. See also *variable space character* and *variable space increment*.

variable space character. The code point assigned by the data stream for which the character increment varies according to the semantics and pragmatics of the variable space function. This code point is not presented, but its character increment parameter is used to provide spacing. See also *variable space character increment*.

variable space character increment. The variable value associated with a variable space character. The variable space character increment is used to calculate the dimension from the current presentation position to a new presentation position when a variable space character is found. See also *variable space character*.

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 baselines when character rotation is zero degrees and no external leading is used. This distance represents the baseline-to-baseline increment that includes the font's maximum baseline extent and the designer's recommendation for internal leading. The font designer can also define a minimum and a 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.

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*.

white space. The portion of a line that is not occupied by characters when the characters of all the words that can be placed on a line and spaces between those words are assembled or formatted on a line. When a line is justified, the white space is distributed among the words, characters, or both on the line in some specified manner. See also *controlled white space*.

width class. A parameter indicating a relative change from the font's normal width-to-height ratio. Examples are normal, condensed, and expanded. Synonymous with *type width*.

X

X-height. The nominal height above the baseline, ignoring the ascender, of the lowercase characters in a font. X-height is usually the height of the lowercase letter *x*. See also *lowercase* and *ascender*.

X_m,Y_m coordinate system. (1) In the IPDS architecture, the medium presentation space coordinate system. (2) In the MO:DCA architecture, the medium coordinate system.

X_p extent. The size of a presentation space or logical page in the X_p dimension. See also *presentation space* and *logical page*.

X_p,Y_p coordinate system. The coordinate system of a presentation space or a logical page. This coordinate system describes the size, position, and orientation of a presentation space or a logical page. Orientation of an X_p, Y_p 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. See also *logical page*, *medium presentation space*, and *presentation space*.

Y

Y_p extent. The size of a presentation space or logical page in the Y_p dimension. See also *presentation space* and *logical page*.

Index

A

B

C

D

E

R

S

T

U

Advanced Function Presentation Consortium

Presentation Text Object Content Architecture Reference

AFPC-0009-03